Old Dominion University

ODU Digital Commons

Computational Modeling & Simulation Engineering Theses & Dissertations

Computational Modeling & Simulation Engineering

Summer 2007

# Distributed Knowledge Discovery in Large Scale Peer-to-Peer Networks

Sachin Shetty
*Old Dominion University*

Follow this and additional works at: https://digitalcommons.odu.edu/msve_etds

 Part of the Computer Sciences Commons, and the Electrical and Computer Engineering Commons

## Recommended Citation

Shetty, Sachin. "Distributed Knowledge Discovery in Large Scale Peer-to-Peer Networks" (2007). Doctor of Philosophy (PhD), Dissertation, Computational Modeling & Simulation Engineering, Old Dominion University, DOI: 10.25777/1jjq-0r58
https://digitalcommons.odu.edu/msve_etds/19

This Dissertation is brought to you for free and open access by the Computational Modeling & Simulation Engineering at ODU Digital Commons. It has been accepted for inclusion in Computational Modeling & Simulation Engineering Theses & Dissertations by an authorized administrator of ODU Digital Commons. For more information, please contact digitalcommons@odu.edu.

# Distributed Knowledge Discovery in Large Scale Peer-to-Peer Networks

by

Sachin Shetty
B.E. Computer Engineering 1998
M.S. Computer Science 2002

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirement for the Degree of

Doctor of Philosophy

Modeling and Simulation

**OLD DOMINION UNIVERSITY**

August 2007

Approved by:

_____
Dr. Min Song (Director)

_____
Dr. Mark Pullen

_____
Dr. John Sokolowski

_____
Dr. Bowen Loftin

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# ABSTRACT

Distributed Knowledge Discovery in Large Scale Peer-to-Peer Networks

Sachin Shetty
Old Dominion University, 2007
Director: Dr. Min Song

Explosive growth in the availability of various kinds of data in distributed locations has resulted in unprecedented opportunity to develop distributed knowledge discovery (DKD) techniques. DKD embraces the growing trend of merging computation with communication by performing distributed data analysis and modeling with minimal communication of data. Most of the current state-of-the-art DKD systems suffer from the lack of scalability, robustness and adaptability due to their dependence on a centralized model for building the knowledge discovery model. Peer-to-Peer networks offer a better scalable and fault-tolerant computing platform for building distributed knowledge discovery models than client-server based platforms. Algorithms and communication protocols have been developed for file search and discovery services in peer-to-peer networks. The file search algorithms are concerned with identification of a peer and discovery of a file on that specified peer, so most of the current peer-to-peer networks for file search act as directory services. The problem of distributed knowledge discovery is different from file search services, however new issues and challenges have to be addressed. The algorithms and communication protocols for knowledge discovery deal with implementing algorithms by which every peer in the network discovers the correct knowledge discovery model, as if it were given the combined database. Therefore, algorithms and communication protocols for DKD mainly deal with distributed computing. The distributed computations are entirely asynchronous, impose very little communication overhead, transparently tolerate network topology changes and peer failures and quickly adjust to changes in the data as they occur. Another important aspect of the distributed computations in a peer-to-peer network is that most of the communication between peer nodes is local i.e. the knowledge discovery model is learned at each peer using information gathered from a very small neighborhood, whose

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

size is independent of the size of the peer-to-peer network. The peer-to-peer constraints on data and/or computing are the hard ones, so the challenge is to show that it is still possible to extract useful information from the distributed data effectively and dependably. The implementation of a distributed algorithm in an asynchronous and decentralized environment is the hardest challenge. DKD in a peer-to-peer network raises issues related to impracticality of global communications and global synchronization, on-the-fly data updates, lack of control, accuracy of computation, the need to share resources with other applications, and frequent failure and recovery of resources. We propose a methodology based on novel distributed algorithms and communication protocols to perform DKD in a peer-to-peer network. We investigate the performance of our algorithms and communication protocols by means of analysis and simulations.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

This dissertation is dedicated to the proposition that

*Success is not final, failure is not fatal: it is the courage to continue that counts.*

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# ACKNOWLEDGMENTS

Though the following dissertation is an individual work, it would not be complete without the help, support, guidance and efforts of a lot of people. I am thankful to Dr. Min Song, my advisor and friend, for giving me guidance and counsel, and for having faith and confidence in me. His patience in reading draft after draft of every paper, proposal and ideas I wrote up continues to amaze me. I thank him for always being willing to help me whenever I approached him.

I am grateful to Dr. Mark Pullen, Dr. John Sokolowski, and Dr. Bowen Loftin for their comments and suggestions. I am thankful to the Department of Electrical and Computer Engineering and Virginia Modeling Analysis and Simulation Center for making it possible for me to do research.

I thank my parents for inculcating in me the dedication and discipline to do whatever I undertake well.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# TABLE OF CONTENTS

LIST OF TABLES ..... 1

LIST OF FIGURES ..... 2

CHAPTER I
INTRODUCTION ..... 3

1.1 Problem Statement ..... 4

1.2 Dissertation Statement ..... 4

1.3 Outline ..... 6

CHAPTER II
BACKGROUND AND RELATED WORK ..... 7

2.1 Structure Learning in Bayesian Networks for knowledge discovery ..... 7

2.1.1 Bayesian networks for Knowledge Discovery ..... 8

2.1.2 Structured Bayesian Learning ..... 9

2.1.3 BN Selection Methods ..... 10

2.2 Distributed Knowledge Discovery ..... 11

2.2.1 Architecture ..... 11

2.3 Epidemic Protocols ..... 15

2.3.1 Background ..... 15

2.3.2 Related Work ..... 16

2.4 Distributed Knowledge Discovery Algorithm ..... 17

2.4.1 Related Work ..... 18

2.5 Distributed Knowledge Discovery in Peer-to-Peer Networks ..... 19

CHAPTER III

Data Mining of Bayesian Network Structure Using a Semantic Genetic Algorithm-Based Approach ..... 22

3.1. Introduction ..... 22

3.2 SEMANTIC GENETIC ALGORITHM-BASED APPROACH ..... 23

3.2.1 Structure Learning of Bayesian Networks ..... 23

3.2.2 Representative Function and Fitness Function ..... 24

3.2.3 Mutation and Crossover Operators ..... 25

Pseudo code for Semantic Crossover ..... 26

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

3.3 Simulations ..... 27
3.3.1 SGA Implementation ..... 27
3.3.2 Simulation Methodology ..... 27
3.3.3 Simulations and Analysis ..... 28
3.4. Conclusions ..... 33
CHAPTER IV
Controlling Information Dissemination Process Using An Epidemic Protocol with Dynamic Fanout ..... 34
4.1. Introduction ..... 34
4.2. ROUND BASED DYNAMIC FANOUT ..... 37
4.2.1 The Round Based Dynamic Fanout Approach ..... 37
4.2.2 Mathematical Analysis ..... 40
4.3.3 Message Overhead ..... 42
4.3. Cluster Based Dynamic Fanout ..... 43
4.3.1 Network Topology ..... 43
4.3.2 Analytical Model ..... 44
4.4. Simulations and Analysis ..... 47
4.4.1 Round Based Dynamic Fanout ..... 47
4.4.2 Cluster Based Dynamic Fanout ..... 50
4.5. Conclusions ..... 55
CHAPTER V
A Majority Based Consensus Methodology For Learning Bayesian Network Structure From Distributed Databases ..... 56
5.1. Introduction ..... 56
5.2. Problem Definition ..... 58
5.3. Majority Consensus based Methodology ..... 60
5.3.1 Majority Consensus Protocol ..... 61
5.3.2 Majority Bayesian Network Learning Algorithm ..... 65
5.4 Simulations and Analysis ..... 69
5.4.1 Effect of Majority Selection on Local Communication ..... 70
5.4.2 Convergence of Majority Bayesian Network Learning Algorithm ..... 73

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

5.5 Conclusion ..... 77
CHAPTER VI
CONCLUSIONS AND FUTURE DIRECTIONS ..... 79
6.1 Conclusions ..... 80
6.2 Future Directions ..... 82
References ..... 84
VITA ..... 90

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# LIST OF TABLES

Table 3.1 Average number of generations.....31
Table 3.2 Average graph errors for 8-node.....31
Table 3.3 Average graph errors for 12-node.....32
Table 3.4 Average graph errors for 18-node.....32
Table 3.5 Average graph errors for 24-node.....32
Table 3.6 Average graph errors for 30-node.....32
Table 3.7 Average graph errors for 36-node.....32
Table 4.1 Round based dynamic fanout for 1,000 and 2,000 nodes.....48
Table 4.2 Round based dynamic fanout for different network sizes.....49
Table 4.3 Message Overhead in the Round based Dynamic Fanout approach .....49
Table 4.4 Chain- Binomial Probabilities for cluster size of 4 .....51
Table 4.5 Simulation and Analytical Values of infected clusters for cluster size of 4.....52
Table 4.6 Chain- binomial Probabilities for cluster size of 5.....53
Table 4.7 Simulation and Analytical Values of the infected clusters for cluster size of 5
.....54

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# LIST OF FIGURES

Fig. 2.1 A Bayesian Network Representation.....9
Fig. 2.2 Distributed Knowledge discovery architectures..... 12
Fig. 2.3 Data Based approach for DKD..... 13
Fig. 2.4 Model Based approach for DKD .....14
Fig. 2.5 Architecture Based approach for DKD.....15
Fig. 3.1 Simulation setup for learning Bayesian network structure..... 27
Fig. 3.2 The structure of the ASIA network. .....28
Fig. 3.3 Plot of generations versus average fitness values (10000 Records)..... 30
Fig. 3.4 Learned BN after 100 generations for 5,000 records - graph errors = 3. .....30
Fig. 3.5 Learned BN after 100 generations for 10,000 records - graph errors = 2. .....31
Fig. 4.1 Round based Dynamic Fanout pseudo-code .....39
Fig. 4.2 Cluster membership depicting intra-cluster fanout and inter-cluster fanout.....44
Fig. 4.3 Beta-Distribution for cluster of size 4..... 52
Fig. 5.1 Local Bayesian Network models present in three peer nodes .....59
Fig. 5.2 Global Bayesian Network discovered at each node in the peer-to-peer network 59
Fig. 5.3 A visual illustration of the Majority Selection process from the perspective of node *i*..... 65
Fig. 5.4 A visual illustration of the Majority Bayesian Network Algorithm from the perspective of node *i*. .....69
Fig. 5.5 Locality for network size of 5000 nodes and majority value of 50 %..... 71
Fig. 5.6 Locality for network size of 10000 nodes and majority value of 50 %..... 71
Fig. 5.7 Percentage of new edges and correct edges discovered for a network of 5000 nodes on a ASIA dataset. .....74
Fig. 5.8 Percentage of nodes which compute the right decision for confidence levels lesser than or equal to 40 % . .....74
Fig. 5.9 Percentage of nodes which compute the right decision for confidence levels greater than or equal to 60 % . .....75
Fig. 5.10 Percentage of nodes which compute the right decision for confidence levels near the 50 % majority ratio..... 76

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# CHAPTER I

# INTRODUCTION

Integrating multiple, independently developed local knowledge discovery models distributed in a peer-to-peer network into a global knowledge discovery model is the crux of this dissertation. Experience shows that this integration is a difficult problem because of increased communication overhead and delayed convergence during the discovery of the integrated model. Most of the current approaches make sub-optimal tradeoffs between run-time performance and convergence of the models. Current practice involves employing one of the two basic approaches: large-scale distributed discovery, which may compromise convergence for performance, and collective integration, which may compromise run-time performance for convergence.

The work proposed in this dissertation attempts to fill the void in a very important research area in modeling and simulation: Learning a descriptive knowledge discovery model from distributed data. As distributed databases are prevalent in most commercial/military organizations, there is a growing need to have an architecture to learn the underlying descriptive knowledge discovery model. Due to logistical and security reasons, organizations expect the architecture to learn the model without exchanging databases among the participating data sites. In absence of data exchange, the architecture is expected to impose minimal communication overhead on the underlying network, without adversely compromising the convergence. The capabilities for bi-directional inferences (e.g., prediction and diagnosis), quick debugging and reconfiguring, combined with a rigorous probabilistic foundation, has led to the rapid emergence of Bayesian Networks as the model of choice for knowledge discovery. This dissertation addresses the integration of independently developed Bayesian Networks in a manner that reconciles demands for lower communication overhead and faster convergence.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## 1.1 Problem Statement

Learning the structure of a distributed Bayesian network from local Bayesian networks learnt in nodes in a peer-to-peer network is an unsolved problem. This dissertation proposes a methodology based on novel distributed algorithms and communication protocols to learn Bayesian networks from distributed data. Our methodology will address challenges related to global synchronization, global communication and dynamism in discovering a distributed Bayesian Network model in a decentralized environment. Our methodology is an emerging technique to address the aforementioned challenges.

The objectives of the work presented in this dissertation are as follows:

- Improving the convergence rate of Bayesian Network structure learning algorithm.
- Better control over the information dissemination between the nodes in a peer-to-peer network.
- Asynchronous computation of distributed Bayesian Network structure learning algorithms.
- Scalability and fault-tolerance in distributed Bayesian Network structure learning algorithms.
- Reducing global communication overhead in distributed computations.

## 1.2 Dissertation Statement

The main contributions of this work are three-fold. The first contribution is a *semantic genetic algorithm* (SGA) to learn the best Bayesian network structure from a database. SGA introduces semantic crossover and mutation operators to aid in obtaining accurate solutions. The crossover and mutation operators incorporate the semantics of Bayesian network structures to learn the structure with very minimal errors. SGA has been proven to discover Bayesian networks with greater accuracy than existing classical genetic algorithms.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The second contribution of this work proposes an adaptive fanout based epidemic protocol to allow the computation of the global Bayesian network to be completed within a pre-determined response time. The protocol is based on controlling the information dissemination process between nodes in a peer-to-peer network. The fanout property of the epidemic protocol has been modified. Two approaches have been implemented to modify the fanout: Round Based Dynamic Fanout and Cluster Based Dynamic Fanout. In the first approach, the network topology is flat and each node transmits a message with a varied fanout every round. In the second approach, the network topology is hierarchical and the fanout values in every cluster differ within the same round. The main objectives are to ensure that peers receive messages within a bounded latency and that the system message overhead is a bounded value.

The third contribution of this dissertation is a majority based consensus methodology to learn a Bayesian network structure from databases distributed in peer-to-peer networks. The methodology consists of a majority consensus protocol and a majority Bayesian network algorithm which are executed in tandem on each node such that every node can learn the Bayesian network structure as if it were given the combined database. The protocol and the algorithm perform their operations asynchronously. Thus it imposes very little communication overhead and transparently tolerates network topology changes and node failures.

To implement DKD in a peer-to-peer network using our methodology, semantic genetic algorithms, adaptive fanout based epidemic protocol, majority consensus protocol, and the majority Bayesian network algorithm should be executed in tandem at each and every node in a peer-to-peer network. The semantic genetic algorithm would be responsible to accurately learn the structure of the local Bayesian network model at each and every node. The majority Bayesian network algorithm would be responsible for updating the local Bayesian network based on the information received from the neighboring peers. The majority Bayesian network algorithm would interact with the majority consensus protocol while sending local Bayesian networks and receiving neighboring Bayesian networks. The majority consensus protocol is responsible for the discovery of the exact

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

distributed Bayesian network at every peer node and ensuring that the communication is local. The majority consensus protocol would entrust the task of choosing neighbors, populating the neighbor list, and controlling the information dissemination to the adaptive fanout based epidemic protocol.

## 1.3 Outline

The remainder of this dissertation follows a traditional format. Chapter 2 discusses the background of Bayesian network structure learning, epidemic protocols and distributed knowledge discovery algorithms. This chapter also presents related work in the above fields as well as limitations of approaches mentioned in the literature and the approach taken by this dissertation. Chapter 3 presents the first contribution, a Semantic Genetic Algorithm, which allows for a faster convergence in learning the structure of a Bayesian Network. The chapter discusses the details of the modified crossover and mutation operators used in the algorithm. In Chapter 4, the information dissemination problem in peer-to-peer network is addressed. This chapter presents an adaptive fanout based epidemic protocol which provides better control over the overall information dissemination process during communication between peers. Chapter 5 presents a majority based consensus methodology to learn a distributed Bayesian network from data distributed among nodes in a peer-to-peer network. Chapter 6 concludes by discussing the contributions of this dissertation to the practice of distributed knowledge discovery and presenting some areas for future work.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# CHAPTER II

# BACKGROUND AND RELATED WORK

This chapter discusses the background of structure learning in Bayesian Networks for knowledge discovery, epidemic protocols and distributed knowledge discovery. This chapter also reviews a sampling of relevant research in the above fields, discusses the limitations of previous approaches, and the approach taken in this dissertation.

## 2.1 Structure Learning in Bayesian Networks for Knowledge Discovery

Knowledge discovery (KD) refers to extracting knowledge from large amounts of data. The KD process is comprised of a set of techniques and algorithms taken from different fields such as statistics, the social sciences, and artificial intelligence. The KD process can be subdivided into different sub-processes depending on the kind of information to be searched and the input data.

- **Clustering** – Process of partitioning a given set of data points into distinct groups or clusters such that the similarity between the data points in one cluster is maximized and the similarity between data points in different clusters is minimized.
- **Classification** –Process of assigning objects to predefined categories or classes.
- **Sequential Patterns** – Process of determining strong sequential dependencies among different events.
- **Association Rules** – Process of finding all rules that correlate the presence of one set of items with that of another set of items from a transaction of item sets.

To implement each of the above processes, a very rich set of algorithms and techniques have been taken from the fields of statistics and machine learning [26]. This dissertation

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

focuses on an algorithm which is based on a probabilistic framework called Bayesian networks. Next, background information related to Bayesian networks and its relevance to KD is provided.

### 2.1.1 Bayesian networks for Knowledge Discovery

Modeling the human learning process has been one of the most challenging tasks faced by researchers in the artificial intelligence field. Machine learning is an area of artificial intelligence concerned with the development of techniques which allow computers to learn. More specifically, machine learning is a method for creating computer programs by the analysis of data sets. A common approach to machine learning is that of discovering causal connections and probabilities between events [9]. Conditional dependencies are a major aspect of learning as they give us the power to infer future events and make intelligent decisions. These conditional dependencies can be shown visually in a graphical model. A Bayesian network (BN) is a special type of graphical model which utilizes Bayes' rule for inference.

Most simply, Bayes' rule can be expressed as:

$$P(b \mid a) = \frac{P(a \mid b)P(b)}{P(a)}$$

where $P(a)$ is the probability of $a$, and $P(a \mid b)$ is the probability of $a$ given that $b$ has occurred.

BNs represent probabilistic relationships among a set of variables. Recently researchers have developed methods for learning BNs from a combination of expert knowledge and data. These methods are fairly new and still evolving, but are found to be very effective in the banking, military and medicine domains [8, 18]. A typical KD process using BNs proceeds as follows [16]. The existing knowledge of a domain is encoded in a BN by an expert. A database is used to update this knowledge to create more BNs. The final result is a refinement of the original expert knowledge and sometimes the identification of new distinctions and relationships. The two most important advantages of using BNs are the capability of encoding expert knowledge in a BN and the nodes and arcs in learned BNs correspond to recognizable distinctions and causal relationships.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

As mentioned earlier, BNs are a special type of graphical model, represented as directed acyclic graph of dependencies between nodes. An edge from node $n_1$ to $n_2$ means that node $n_1$ directly influences node $n_2$. A BN describes a collection of joint probabilities between several nodes. The chain rule allows us to define joint probabilities in terms of conditional probabilities. We determine $P(x_1,....,x_n)$ by computing $\prod_{i=1}^{n} P(x_i \mid pa(x_i))$, where $pa(x_i)$ refers to the parent of variable $x_i$. Fig. 2.1 shows a hypothetical example of a BN. The BN represents three variables ($X=\{x,\overline{x}\}, Y=\{y,\overline{y}\}$ and $Z=\{z,\overline{z}\}$). X and Y are parents of variable Z. The joint probabilities can be determined as $p(X,Y,Z)=p(X).p(Y).p(Z \mid X,Y)$.

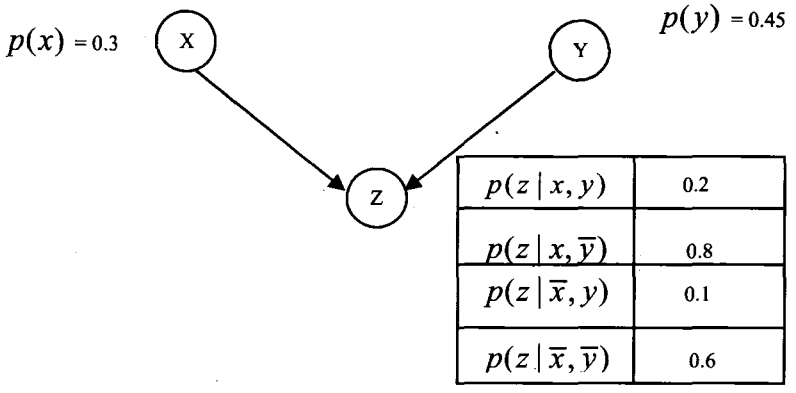


Fig. 2.1 A Bayesian Network Representation.

### 2.1.2 Structured Bayesian Learning

Once a BN is constructed it constitutes an efficient device to perform probabilistic inference, but the problem of building such a network still remains. The structure and conditional probabilities necessary for characterizing the network can be provided either externally by experts or from direct empirical observations. The learning task in a BN can be separated into two subtasks, structure learning, that is to identify the topology of the network, and parameter learning, the numerical parameters (conditional probabilities) for a given network topology. This dissertation focuses on structure learning, rather than

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

parameter learning. However, for the construction of a complete BN, it is also necessary to estimate parameters.

Structure learning of a BN consists of two tasks: one to identify the nodes or variables, and two to fill in the directed edges. Given a list of data items, finding the representative variables or nodes is a trivial task. The real challenge, then, is to fill in the directed edges. We want to know which nodes are parents of which other nodes. The set of parents and the range of each node tells us what the target conditional probabilities are. This problem has been shown to be NP-hard [18].

### 2.1.3 BN Selection Methods

In order to learn a BN from data, one of the most common criteria is the Bayesian Descriptor Estimation (BDE) score, which is equivalent to the Minimum Descriptor Length (MDL) score. The main idea behind the MDL score is to maximize the final score output while minimizing the complexity of the structure. To reduce the search space during learning BN, many heuristic approaches have been taken: greedy search, best-first search, and Monte-Carlo methods [21]. Several algorithms based on the aforementioned approaches have been proposed. The important ones are K2 [8], and Genetic Algorithms [42]. Larranaga *et al.* proposed a genetic algorithm based on the score-based greedy algorithm[40]. In their algorithm, a directed acyclic graph (DAG) is represented by a connectivity matrix that is stored as a string. The recombination is implemented as one-point crossover on these strings, while mutation is implemented as random bit flipping. In a related work, Larranaga *et al.* [41] employed a wrapper approach by implementing a genetic algorithm that searches for an ordering that is passed on to K2 [8], a score-based greedy learning algorithm. The results of the wrapper approach were comparable to those of their previous genetic algorithms. Different crossover operators have been implemented in a genetic algorithm to increase the adaptiveness of the learning problem with good results [9]. Lam and Bacchus [39] proposed a hybrid evolutionary programming (HEP) algorithm that combines the use of independence tests with a quality-based search. In the HEP algorithm, the search space of DAG is constrained in the sense that each possible DAG only connects two nodes if they

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

show a strong dependence in the available data. The HEP algorithm evolves a population of DAG to find a solution that minimizes the minimal MDL score. A common feature of the aforementioned algorithms is that the mutation and crossover operators were classical in nature. These operators do not help the evolution process to reach the best solution.

Wong *et al.* [59] developed an approach based on MDL score and evolutionary programming. They have integrated a knowledge-guided genetic operator for the optimization in the search process. However, the fitness function is not taken into account to guide the search process. Myers and Levitt [46] have proposed an adaptive mutation operator for the learning structure of BN from incomplete data. It is a generalized approach to influence the current recombination process based on previous population; it also does not take into account the fitness of a population. Blanco *et al.* [4] have adopted the estimation of distribution algorithms method for learning BN without the use of crossover and mutation operators. This is not in accordance with the classical genetic algorithm due to the lack of recombination operators. Recently, Dijk *et al.* [12] built another generalized genetic algorithm to improve the search process without taking into account the specific characteristics of the population. As we see, most of the genetic algorithm-based approaches mentioned above adopt a generalized approach to improve the search process.

## 2.2 Distributed Knowledge Discovery

Distributed Knowledge Discovery (DKD) can be described as a process of discovering knowledge from a database geographically distributed among a group of sites connected by a high latency network. The goal of a DKD system is to build a global model representing the data distributed at more than one site. This section presents architectural, algorithm, and communication challenges in implementing DKD.

### 2.2.1 Architecture

Fig. 2.2 shows a classification of current DKD system architectures. Each of the architectures poses different challenges and provides different benefits.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

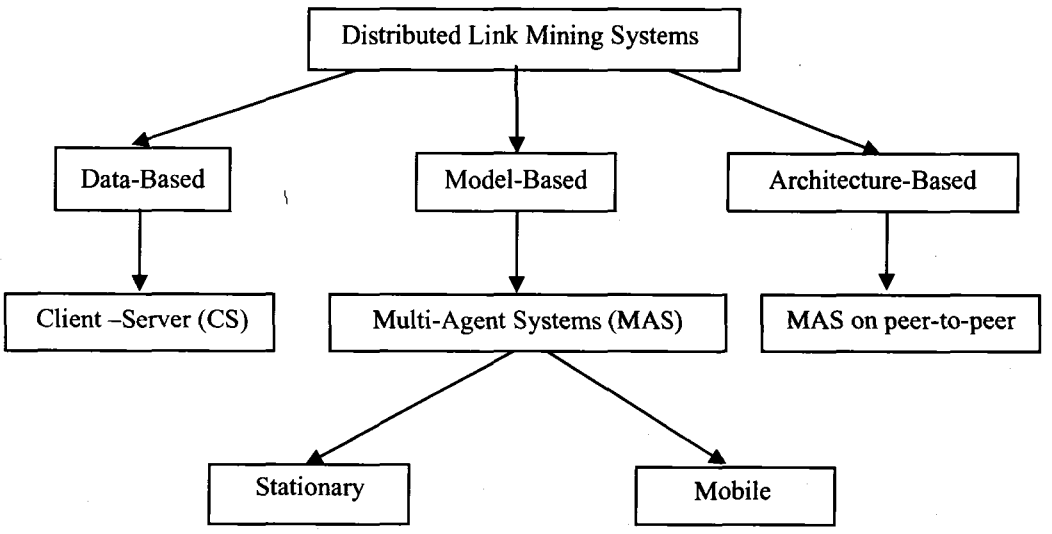


Fig. 2.2 Distributed Knowledge discovery architectures.

**Data-based.** This is a *centralized learning* strategy which refers to a common approach to DKD, as the focus is on the location of the data. Fig. 2.3 describes the architecture. In the figure, the data sites refer to the distributed databases, and the KD process is responsible for building a complete KD model (global model) based on the data from all the distributed locations. Centralized learning involves data from all data sites moved to a single central server for analysis and predictive modeling. Client-Server technologies are ideal for implementing this strategy. However, this strategy for DKD has a high communication overhead as it involves transfer of huge quantities of data. The advantage of this strategy is that the DKD server has well defined computational resources which have the ability to handle resource intensive KD tasks.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

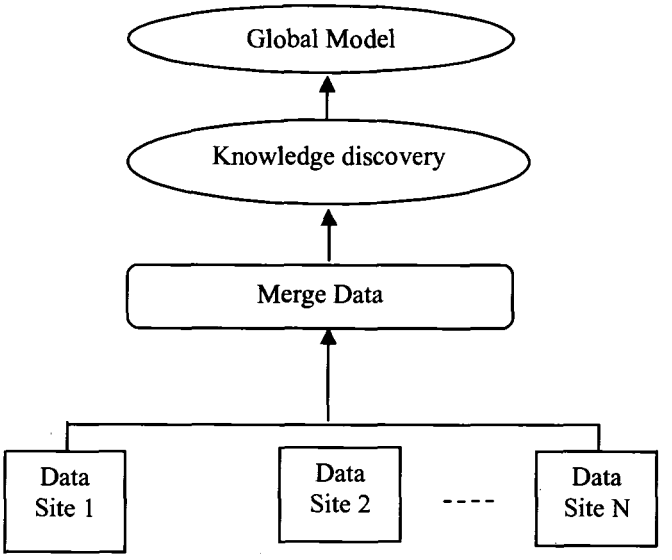


Fig.2.3 Data Based approach for DKD

**Model-Based.** This is a *local learning* approach, where in-place strategies are employed by building models at each data site and then moved to the server where they are combined. Fig. 2.4 sketches the architecture. In the figure, the global model is created by the merging of the local models. Multi-agent systems have been used to implement the local learning strategies. Agent technology is seen as being able to address the specific concern of increasing scalability and enhancing performance by moving code instead of data and thereby reducing the communication overhead incurred in the centralized strategy. However, the absence of dedicated KD servers and the lack of control over available computational resources at remote sites are limitations.

In this approach, the local models can be implemented by sequential knowledge discovery algorithms without any modification. The challenge is to combine the partial results coming from the local models. Different techniques can be adopted, based on voting strategies or collective operations [28]. Multi-agent systems may apply meta-learning to combine partial results of distributed local classifiers [52]. The drawback of the model-driven approach is that it is not always possible to obtain an exact final result; i.e., the global knowledge model obtained may be different from the one obtained by

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

applying the data-driven approach (if possible) to the same data. Moreover, in this model, hardware resource usage is not optimized. If the heavy computational part is always executed locally to data, when the same data is accessed concurrently, the benefits coming from the distributed environment might vanish due to the possible strong performance degradation.

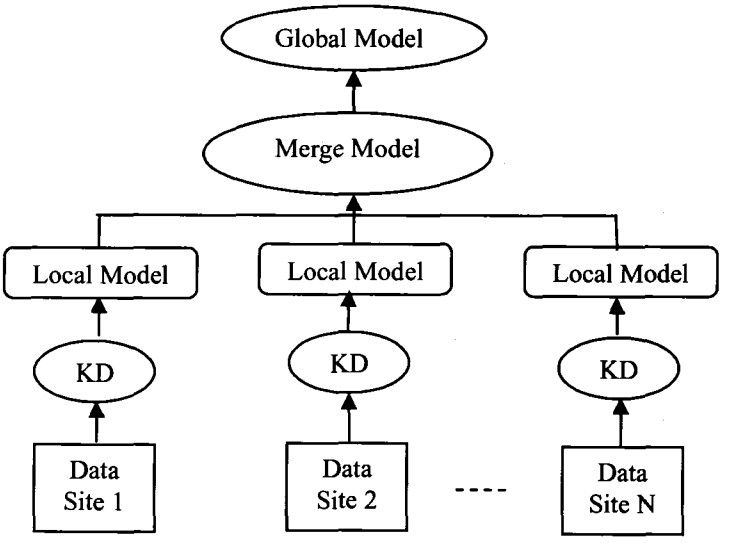


Fig. 2.4 Model Based approach for DKD

**Architecture-Based**. This is a layered approach to DKD. Fig. 2.5 shows the schematic diagram for this approach. This approach provides a control over the performance of DKD. It is a relatively newer approach to distributed learning. The two main layers in this approach are: resource selection and communications. The resource selection layer moves data to different sites with respect to their original location for better performances. The communications layer avoids sending the local models to a central location and allows the global model to be built during the local computation. This allows for arbitrary precision to be achieved, at the price of a higher communication overhead. Since in this approach for DKD the focus is on optimized resource usage, we refer to this approach as architecture-driven.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

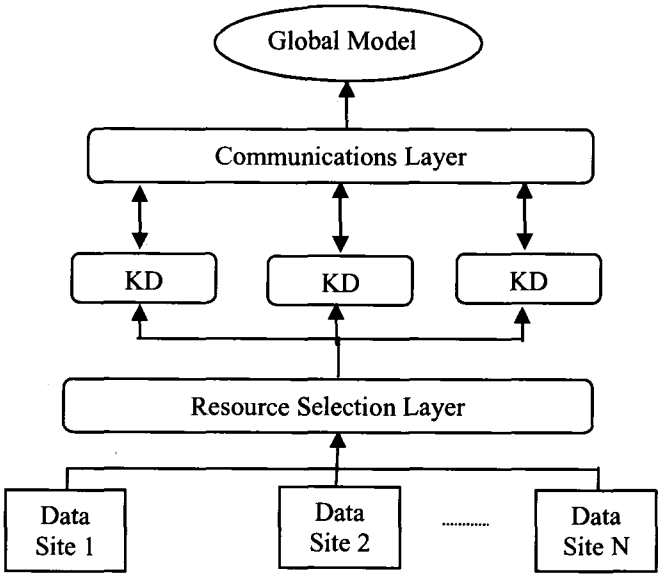


Fig. 2.5 Architecture Based approach for DKD.

## 2.3 Epidemic Protocols

Having discussed the architecture models for the distributed knowledge discovery model in the previous section, the next step is to describe protocols and algorithms to learn the structure of Bayesian networks from distributed data. This section presents details of epidemic protocols: communication protocols which provide a scalable, robust, and probabilistic reliable mechanism for information dissemination in large-scale peer-to-peer communication systems

### 2.3.1 Background

Epidemic algorithms have gained popularity as a robust and scalable way of propagating information in distributed systems [3]. A process that wishes to disseminate information to the system does not send it to a server, but rather to a set of other peer processes, chosen at random. In turn, each of these processes does the same and also forwards the information to randomly selected processes, and so forth. The principle underlying this information dissemination technique is similar to the spread of epidemics. Once started, epidemics are hard to eradicate: a few people infected by a contagious disease are able to spread it, directly or indirectly, to a large population. Epidemics are resilient to *failures* in

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

the infection process. That is, even if many infected people die before being able to transmit the disease, or are immunized, the epidemic is still *reliably* propagated over populations. Epidemic dissemination algorithms are simple and easy to deploy. In addition to scalability, epidemic algorithms exhibit a very stable behavior even in the presence of link and node failures. There is no single point of failure and the reliability degrades gracefully with the number of failures. A large amount of research has been devoted to observing, analyzing, and devising mathematical theories for epidemics. Applying the epidemic idea to dissemination of information among a large number of nodes with a dynamic connection topology is thus very appealing [20], [29]. The use of epidemic algorithms has been explored in applications such as failure detection [58], routing in adhoc networks [44], data aggregation [15], resource discovery and monitoring [57] and database replication [11].

### 2.3.2 Related Work

As mentioned in the introduction section, most of the epidemic protocols in the literature usually adopt a constant fanout. The first attempt at implementing an epidemic protocol for information dissemination resulted in the *Bimodal Multicast* [3]. This protocol works on two phases. In the first phase, a best-effort multicast protocol (e.g., IP multicast) is used for a rough dissemination of messages. In the second phase, a gossip based peer retransmission protocol is used to ensure reliability with a certain probability. In particular, every process in the system periodically gossips a digest of its received messages; receivers can solicit such messages from the sender. The fanout used for the second phase is a constant of one. The gossip based probabilistic multicast protocol proposed for failure detection also assumes a constant fanout of one [58]. The failure detection service has been performed for larger network sizes but the fanout has been kept to a constant of one irrespective of network size. More recently, there have been efforts to implement reliable group communication protocols for distributed systems and applications such as publish/subscribe systems [13]. These protocols deploy a scalable, peer-to-peer membership service, but the protocol was never designed to compute an optimal value of fanout. Instead, the fanout values were set to be a constant for a given network. In particular the fanout values are increased to higher values as the network size

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

increases. Higher fanout degrades performance due to increased message overhead. For networks with dynamic behavior where information is changing continuously, for example, a network of sensors or a cluster of distributed computing hosts, spatial epidemic protocol can be used to guarantee that the new information reaches all the nodes. In [30], the spatial epidemic protocol bounds propagation time by a poly-logarithmic function in distance by choosing epidemic targets with a probability which is an inverse polynomial function of distance. The fanout is assumed to be a constant of one. In [29], it is discussed that a generic gossip protocol needs $O(n \log n)$ messages to spread a rumor. For example, the gossip protocol which forwards each rumor from the sender to the receiving nodes for $O(\log n)$ rounds needs to transmit the rumor $O(n \log n)$ times in order to ensure that every player finally receives the rumor with a high probability.

The very first attempt to compute an optimal value for the fanout in a probabilistic reliable information dissemination process was performed by [32]. The authors computed the fanout needed to deliver information to all nodes with a high probability by using random graphs. They showed that a fanout in the order of $\log(n) + c + o(1)$ gives a success probability of $e^{e^{-c}}$, where $c$ is a design parameter whose value ranges from 0 to 1. The reliability of this gossip-based protocol is related to key system parameters (system size, failure rates, and the number of gossip targets). Reliability can be maintained by ensuring that each peer provides only a small subset of the total membership information, and by organizing peer members into a hierarchical structure that reflects their proximity according to some network-related metric. The network topology is hierarchical, which leads to separate fanout values for inter-cluster and intra-cluster peers.

## 2.4 Distributed Knowledge Discovery Algorithm

A distributed knowledge discovery architecture that employs epidemic protocols for communication provides a better run-time performance, but we still have to construct a consistent global knowledge discovery model from local knowledge discovery models distributed in peer-to-peer networks. This section talks about algorithms in the literature,

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

that discuss different approaches to discovering a global knowledge discovery model.

### 2.4.1 Related Work

There have been numerous approaches to perform structure learning of a BN from complete data [36, 37, 51,53]. Learning the structure of BN from incomplete data has also been covered in [38, 44, 45,46,55]. The structure and parameter learning methodologies for complete data presented above have been tested on local datasets only. The performance of these methodologies on distributed datasets has not been experimented. For all the above methods to be effective, all data must be available at a central location. The learning methodologies for incomplete data are a good fit for distributed Bayesian network learning. But none of the above methodologies have been experimented in an environment where additional data is available for improving the learning performance.

Though there has been a lot of research performed on learning the structure and parameters of a BN on a local dataset, research efforts on structure learning on distributed datasets have been minimal. The available research on distributed Bayesian network learning can be classified as structure and parameter learning on homogeneous and heterogeneous data sets. To the best of our knowledge, there have been only two models presented for distributed Bayesian network learning. Both models propose a centralized approach. The centralized solution to this problem is to download all datasets from distributed nodes. Kenji has worked on the homogeneous distributed learning scenario [31]. In this case, every distributed node has the same feature but different observations. A collective approach to implementing a Distributed Bayesian Network (DBN) has been proposed by Chen *et. al* [6] but this approach is implemented in a client-server framework for heterogeneous datasets. To identify cross links, every node has to submit relevant datasets to the central server. The central server is responsible for the construction of the global BN model. Each local node can create its own BN model based on its local datasets only.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The distributed algorithm proposed in this chapter deviates from the client-server model proposed by Chen et. al [6] and Johnson et. al. [27]. The distributed algorithm is based on a majority voting protocol. Majority voting protocols have been proposed in research areas wherein a decision is made by participating nodes. There are few majority vote protocols which are based on local communication. Local communication is key to the success of our distributed algorithm. The majority vote problem is similar to the persistent bit problem, for which local protocols were given [34, 35,5], and the two problems are reducible to one another. The main drawback of the aforementioned persistent bit protocols is that each of them assumes some form of synchronization: In [35], nodes query groups of other nodes and must await a reply before they proceed, while [34] works in locked-step, assuming a global clock pulse. In contrast, our majority vote protocol requires no synchronization at all. There are also more subtle differences which make these protocols impractical for majority vote. For instance, the former only works when the majority is very evident while the latter, because it allows any intermediate result to be corrupted, requires $O(n)$ memory at each node, where $n$ is the network size.

There have been few approaches to implementing distributed data mining algorithms in peer-to-peer networks [32]. In this approach, a local communication based model is chosen to learn association rules in a distributed database. The messages exchanged between nodes include the data items and a confidence level for the rules. The efficiency of the local communication model is dependent on the size of the data items exchanged and the neighbor list. There is a significant message overhead due to the presence of data items in each message exchange between two peers. Also the validity of the protocol depends on user supplied frequency and confidence parameters. This makes the protocol less robust to different kinds of datasets.

## 2.5 Distributed Knowledge Discovery in Peer-to-Peer Networks

This dissertation proposes a methodology to learn a distributed Bayesian Network from local Bayesian networks learnt at individual nodes which are part of a peer-to-peer network. Our methodology consists of a *semantic genetic algorithm* to learn a local

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Bayesian network structure, an *adaptive fanout based epidemic protocol* to provide a scalable, robust and probabilistic reliable controlled communication capability, and a *majority based consensus methodology* to facilitate learning the accurate structure of a global Bayesian Network at each and every node in the network.

The methodology is similar in principle to the architectural model described in section 2.2.1. The focal point of this dissertation is the implementation of the communication layer in the architecture model described in Fig. 2.5. The main responsibility of the communication layer is to ensure the quicker convergence of learning a global Bayesian network from distributed data in a scalable and robust network at the expense of lower communication overhead. Epidemic protocols are potentially effective solutions for disseminating information in large scale and dynamic systems. They are easy to deploy, robust and provide high resilience to failures. They proactively fight random processes and network failures and do not need any reconfiguration when failures occur. This characteristic is particularly useful in peer-to-peer systems deployed on Internet or ad-hoc networks.

One can adjust the parameters of an epidemic information dissemination protocol in such a way that high reliability of dissemination is achieved despite process crashes and disconnections, packet losses and dynamic network topology. This provides the illusion of a global and virtual information system that every client can access and also takes part in implementing. The constant fanout parameter in an epidemic protocol is changed to an adaptive fanout to provide better control over the information dissemination process. An adaptive fanout based epidemic protocol provides a mechanism to compute the global Bayesian Network model within a pre-determined response time, by controlling the information dissemination process.

As the methodology is implemented in a peer-to-peer network, the computation of the global Bayesian network occurs in-network. A majority based consensus methodology ensures that every node learns an exact global Bayesian Network, as if it were given the combined database. The methodology consists of a majority consensus protocol and

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

majority Bayesian network algorithm. The protocol and algorithm work in tandem at each node. The protocol and algorithm are entirely asynchronous, impose very little communication overhead, transparently tolerate network topology changes and node failures, and quickly adjust to changes in the data as they occur.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# CHAPTER III

# Data Mining of Bayesian Network Structure Using a Semantic Genetic Algorithm-Based Approach

## 3.1. Introduction

One of the most important steps in data mining is to build a descriptive model of the database being mined. To do so, probability-based approaches have been considered an effective tool because of the uncertain nature of descriptive models. Unfortunately, high computational requirements and the lack of proper representation have hindered the building of probabilistic models. To alleviate the above problems, probabilistic graphical models have been proposed. In the past decade, many variants of probabilistic graphical models have been developed, with the simplest variant being Bayesian networks (BN) [52]. BN is a popular descriptive modeling technique to see relationships between attributes of a set of records for available data. It has been employed to reason under uncertainty, with wide varying applications in the field of medicine, finance, and military planning [52] [27]. Computationally, BN provides an efficient way to represent relationships between attributes and allow reasonably fast inference of probabilities. Learning BN from raw data can be viewed as an optimization problem where a BN has to be found that best represents the probability distribution that has generated the data in a given database [18]. This has lately been the subject of considerable research since the traditional designer of a BN may not be able to see all of the relationships between the attributes. In this chapter, we focus on the structure learning of a BN from a complete database. The database stores the statistical values of the variables as well as the conditional dependence relationship among the variables. We employ a genetic algorithm technique to learn the structure of BN.

A typical genetic algorithm works with a population of individuals each of which needs to be coded using a *representative function* and be evaluated using a *fitness function* to

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

measure the adaptiveness of each individual. These two functions are the basic building blocks of a genetic algorithm. To actually perform the algorithm, three genetic operators are used to explore the set of solutions: *reproduction*, *mutation*, and *crossover*. The reproduction operator promotes the best individual structures to the next generation. That is, the individual with the highest fitness in a population will reproduce with a higher probability than the one with the lowest fitness. The mutation operator toggles a position in the symbolic representation of the potential solutions. Mutation avoids local optima by exploring new solutions by introducing a variation in the population. The crossover operator exchanges genetic material to generate new individuals by selecting a point where pieces of parents are swapped. The main parameters, that influence the genetic algorithm search process, are initial population, population size, mutation, and crossover operators.

The rest of the chapter is organized as follows. Section 3.2 presents the details of our approach for structure learning in a BN structure using a modified genetic algorithm. In Section 3.3, we experiment with two different genetic algorithms. The first one is the genetic algorithm with classical genetic operators. In the second algorithm, we extend the standard mutation and crossover operators to incorporate the semantic of the BN structures. Finally, Section 3.4 concludes the chapter and proposes some thoughts for further research.

## 3.2 SEMANTIC GENETIC ALGORITHM-BASED APPROACH

### 3.2.1 Structure Learning of Bayesian Networks

Formally, a BN consists of a set of nodes that represent variables, and a set of directed edges among the nodes. Each node is represented by a finite set of mutually exclusive states. The directed edges between nodes represent the dependence between the linked variables. The strengths of the relationships between the variables are expressed as conditional probability tables (CPT). Thus a BN efficiently encodes the joint probability

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

distribution of its variables. For $n$-dimensional random variable $(X_1,....,X_n)$, the joint probability distribution is determined as follows,

$$P(x_1,....,x_n)=\prod_{i=1}^{n}P(x_i\,|pa(x_i)) \tag{3.1}$$

where $x_i$ represents the value of the random variable $X_i$ and $pa(x_i)$ represents the value of the parents of $X_i$. Thus, the structure learning problem of a BN is equivalent to the problem of searching the optimum in the space of all DAG. During the search process, a trade-off between the structural network complexity and the network accuracy has to be made. The trade-off is necessary as complex networks suffer from over fitting, making the run time of inference very long. A popular measure to balance complexity and accuracy is based on the principle of MDL from information theory (Lam and Bacchus, 1994). In this chapter, the BN structure learning problem is solved by searching for a DAG that minimizes the MDL score.

### 3.2.2 Representative Function and Fitness Function

The first task in a genetic algorithm is the representation of an initial population. To represent a BN as a genetic algorithm individual, an edge matrix or adjacency matrix is needed. The set of network structures for a specific database characterized by $n$ variables can be represented by an $n\times n$ connectivity matrix $C$. Each bit represents the edge between two nodes where

$$C_{ij}=\begin{cases}1, \text{ if } j \text{ is a parent of } i\\ 0, \text{ otherwise}\end{cases}$$

The two-dimensional array of bits can be represented as an individual of the population by the following string $C_{11}C_{12}...C_{1n}C_{21}C_{22}...C_{2n}...C_{n1}C_{n2}...C_{nn}$, where the first $n$ bits represent the edges to the first node of the network and so on. It can be easily found that $C_{kk}$ are the irrelevant bits which represent an edge from node $k$ to itself which can be ignored by the search process.

With the representative function decided, we need to devise the generation of the initial population. There are several approaches to generate initial population. We implemented

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

the Box-Muller random number generator to select how many parents would be chosen for each individual node. The parameters for the Box-Muller algorithm are the desired average and standard deviation. Based on these two input parameters, the algorithm generates a number that fits the distribution. For our implementation, the average corresponds to the average number of parents for each node in the resultant BN. After considerable experimentation with databases whose Bayesian structure is similar to the ASIA network [43], we found that the best average was 1.0 with a standard deviation of 0.5. Though this approach is simple, it creates numerous illegal DAG due to cyclic subnetworks. An algorithm to remove or fix these cyclic structures has to be designed. The basic operation of the algorithm is to remove a random edge of a cycle until cycles are not found in a DAG individual.

Now that the representative function and the population generation have been decided, we need to find a good fitness function. Most of the *state-of-the-art* implementations use the fitness function proposed in the algorithm K2 [8]. The K2 algorithm assumes an ordered list of variables as its input. It maximizes the following function by searching for every node from the ordered list of a set of parent nodes:

$$g(x_i, pa(x_i)) = \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}! \tag{3.2}$$

where $r_i$ represents the possible value assignments $(v_{i1}, ..., v_{ir_i})$ for the variable with index $i$. $N_{ijk}$ represents the number of instances in a database in which a variable $X_i$ has value $v_{ik}$, and $q_i$ represents the number of unique instantiations of $pa(x_i)$.

### 3.2.3 Mutation and Crossover Operators

We introduce two new operators, semantic mutation (SM) and single point semantic crossover (SPSC), to the existing standard mutation and crossover operators. The SM operator is a heuristic operator that toggles the bit value of a position in the edge matrix to ensure that the fitness function $g(x_i, pa(x_i))$ is maximized. The SPSC operator is specific to our representation function. As the function is a two-dimensional edge matrix consisting of columns and rows, our new crossover operator operates on either columns

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

or rows. Thus the crossover operator generates two offsprings by either manipulating columns or rows. The SPSC crosses two parents by manipulating columns or parents and maximizing the function $g(x_i, pa(x_i))$, and also manipulating rows or children and maximizing the function $\prod_i g(x_i, pa(x_i))$. By combining SM and SPSC, we implement our new genetic algorithm called the semantic genetic algorithm (SGA). The following is the pseudocode for the semantic crossover operation. The algorithm expects an individual as input and returns the modified individual after applying semantic crossover operations.

---

Pseudo code for Semantic Crossover

Step 1. Initialization

Read the input individual and populate a parent table for each node

Step 2. Generate new individual

For each node in the individual do the following *n* times

2.1 Execute the Box Mueller algorithm to find how many parents need to be altered.

2.2 Ensure that the nodes selected as parents do not form cycles. If cycles are formed repeat step 2.1

2.3 Evaluate the network score of the resultant structure.

2.4 If current score is higher than previous score, then the chosen parents are the new parents of the selected node

Repeat steps 2.1 through 2.4.

Step 3. Return the final modified individual.

---

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## 3.3 Simulations

### 3.3.1 SGA Implementation

The SGA algorithm has been implemented and incorporated into the Bayesian network tools in Java (BNJ) [2]. BNJ is an open-source suite of software tools for research and development using graphical models of probability. Specifically, SGA is implemented as a separate module using the BNJ API. To depict the Bayesian network, BNJ visually provides a visualization tool to create and edit Bayesian networks.

### 3.3.2 Simulation Methodology

Fig. 3.1 shows the overall simulation setup to evaluate our genetic algorithm. Following are the main steps of the algorithm:

*Step 1.* Determine a BN and simulate it using a probabilistic logic sampling technique [21] to obtain a database $D$, which reflects the conditional relations between the variables;

*Step 2.* Apply our SGA approach to obtain the BN structure $B_s$, which maximizes the probability $P(D|B_s)$;

*Step 3.* Evaluate the fitness of the solutions.

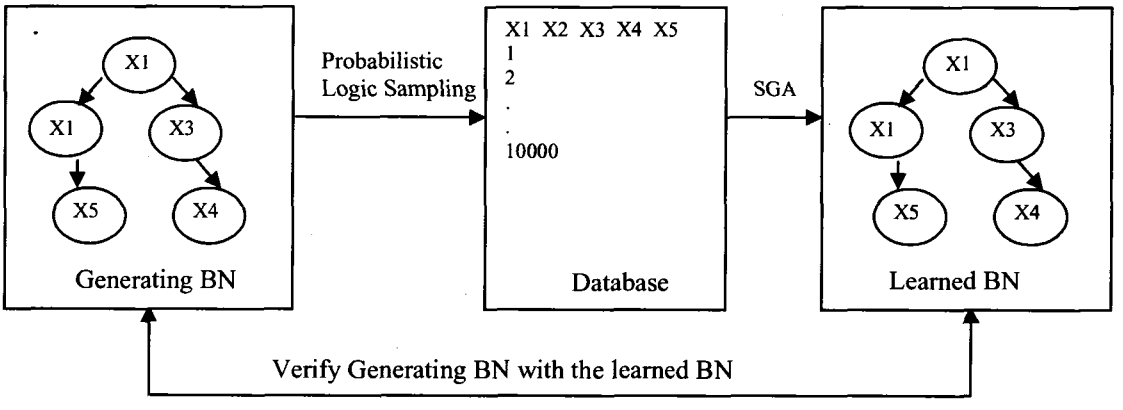


Fig. 3.1 Simulation setup for learning Bayesian network structure.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

### 3.3.3 Simulations and Analysis

The BN sizes used in our simulations are 8, 12, 18, 24, 30, and 36. The 8-node BN used in the simulations is from the ASIA networks as shown in Fig. 3.2. The ASIA network illustrates a method of propagation of evidence and considers a small amount of fictitious qualitative medical knowledge. The remaining networks were created by adding extra nodes to the basic ASIA network.

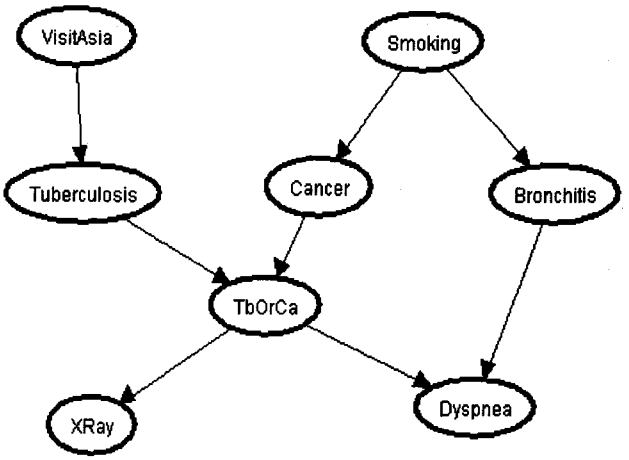


Fig. 3.2 The structure of the ASIA network.

There are several techniques for simulating BN. For our experiments we have adopted the probabilistic logic sampling technique. In this technique, the data generator generates random samples based on the ASIA network's joint probability distribution table. The data generator sorts nodes topologically and picks a value for each root node using the probability distribution, and then generates values for each child node according to its parent's values in the joint probability table. The root mean square error (RMSE) of the data generated compared to the ASIA network is approximately zero. This indicates that the data was generated correctly. We have populated the database with 2000, 3000, 5000, and 10,000 records. This was done to measure the effectiveness of the learning algorithm for larger records. The following inputs are used in the simulations:

- Population size $\lambda$. The experiments have been carried out with $\lambda = 100$.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

- Crossover probability $p_c$, we chose $p_c$=0.9.
- Mutation rate $p_m$, we considered $p_m$=0.1.

The fitness function used by our algorithm is based on the formula proposed by Cooper and Herskovits [8]. For each of the samples (2000, 3000, 5000, 10000), we executed 10 runs with each of the above parameter combinations. We considered the following four metrics to evaluate the behavior of our algorithm.

- *Average fitness value* – This is an average of fitness function values over 10 runs.
- *Best fitness value* – This value corresponds to the best fitness value throughout the evolution of the genetic algorithm.
- *Average graph errors* – This represents the average of the graph errors between the best BN structure found in each search and the initial BN structure. Graph errors are defined to be an addition, a deletion or a reversal of an edge.
- *Average number of generations* – This represents the number of generations taken to find the best fitness function.

For comparison purpose, we also implemented the classical genetic algorithm (CGA) with classical mutation (CM) and single point cyclic crossover (SPCC) operators. Fig. 3.3 plots the average fitness values for the following parameter combination. The average and best fitness values are expressed in terms of $\log P(D|B_s)$. The number of records is 10,000. The figure also shows the best fitness value for the whole evolution process. One can see that SGA performs better than CGA in the initial 15 - 20 generations. After 15 - 20 generations, the genetic algorithm using both operators stabilizes to a common fitness value. The final fitness value is very close to the best fitness value. An important observation is that the average fitness value does not deviate by any significant amount even after 100 generations. The best fitness value is carried over to every generation and is not affected.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

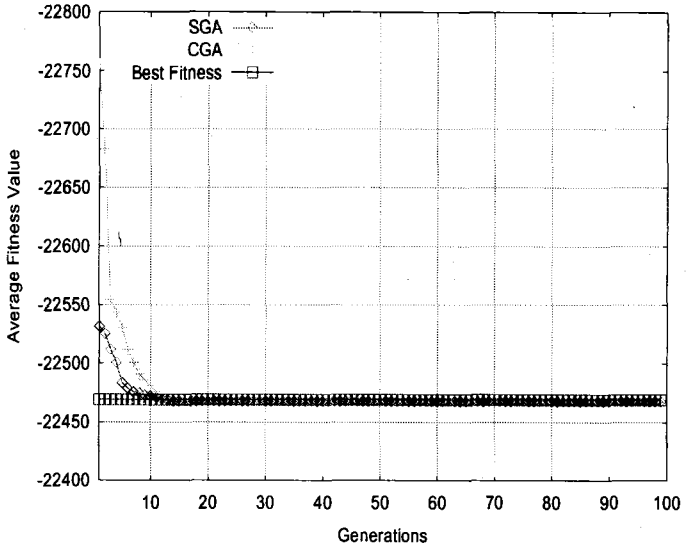


Fig. 3.3 Plot of generations versus average fitness values (10000 Records).

The final learned BN was constructed from the final individual generated after 100 generations. Figs. 3.4 and 3.5 plot the final learned BN for 5,000 records and 10,000 records, respectively. It can be observed that for both scenarios, the learned BN differs from the actual generating BN shown in Fig. 3.2 by a small number of graph errors. It is also worth noting that the number of graph errors reduces when the total number of records increases. This could mean that to reduce the total number of graph errors, a large number of records needs to be provided.

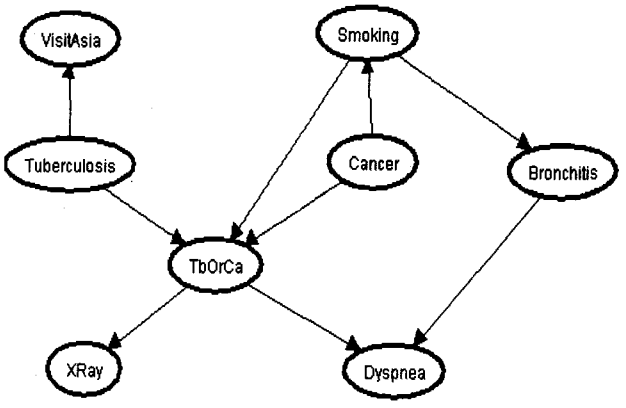


Fig. 3.4 Learned BN after 100 generations for 5,000 records - graph errors = 3.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

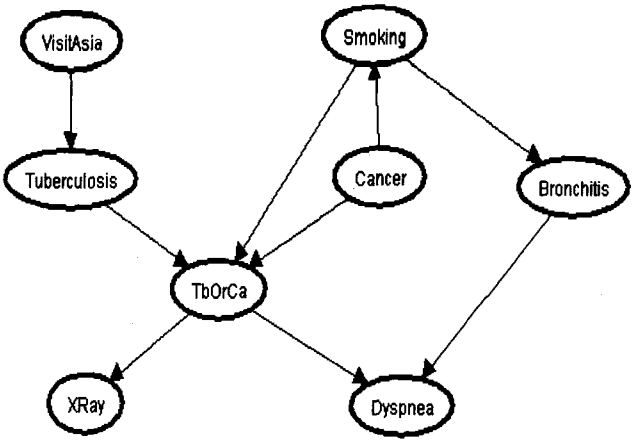


Fig. 3.5 Learned BN after 100 generations for 10,000 records - graph errors = 2.

Tables 3.1 and 3.2 provide the average number of generations and the average graph errors for different numbers of records. It is obvious that for 3000 records the total number of generations taken to achieve the stabilized fitness value is very high. Also the average number of graph errors is too high. For 5,000, and 10,000 records the values for these metrics are reasonable and acceptable.

Table 3.1 Average number of generations.

| Records | SGA | CGA |
|---|---|---|
| 3000 | 25 | 30 |
| 5000 | 20 | 15 |
| 10000 | 20 | 15 |

Table 3.2 Average graph errors for 8-node

| Records | SGA | CGA |
|---|---|---|
| 3000 | 3 | 4 |
| 5000 | 2 | 3 |
| 10000 | 2 | 3 |

To compare the performance of SGA with CGA in the presence of larger BN structures, we modified the 8-node ASIA network and generated five additional BN with node sizes 12, 18, 24, 30, and 36. Tables 3.3-3.7 show results for simulations carried out on these additional BNs. The tables compare the average graph errors in both approaches. The accuracy of SGA does not deteriorate under increased network sizes.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Table 3.3 Average graph errors for 12-node

| Records | SGA | CGA |
|---|---|---|
| 3000 | 21 | 28 |
| 5000 | 24 | 29 |
| 10000 | 20 | 25 |

Table 3.4 Average graph errors for 18-node

| Records | SGA | CGA |
|---|---|---|
| 3000 | 19 | 24 |
| 5000 | 19.5 | 25.5 |
| 10000 | 20.7 | 26 |

Table 3.5 Average graph errors for 24-node

| Records | SGA | CGA |
|---|---|---|
| 3000 | 14.8 | 22.2 |
| 5000 | 15.3 | 23.1 |
| 10000 | 10.9 | 13.3 |

Table 3.6 Average graph errors for 30-node

| Records | SGA | CGA |
|---|---|---|
| 3000 | 15.7 | 21.5 |
| 5000 | 14.9 | 20.3 |
| 10000 | 14 | 18.9 |

Table 3.7 Average graph errors for 36-node

| Records | SGA | CGA |
|---|---|---|
| 3000 | 15.1 | 19.4 |
| 5000 | 15.6 | 20.5 |
| 10000 | 13.6 | 22.5 |

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## 3.4. Conclusions

In this chapter, we have presented a new semantic genetic algorithm (SGA) for BN structure learning. This algorithm is another effective contribution to the list of structure learning algorithms. Our results show that SGA discovers BN structures with a greater accuracy than existing classical genetic algorithms. Moreover, for large network sizes, simulation results show that the accuracy of SGA does not degrade and this accuracy improvement does not come with an increase of search space. In all our simulations, 100 to 150 individuals are used in each of the 100 generations. Thus 10,000 to 15,000 networks are completely searched to learn the BN structure. Considering that the exhaustive search space is of $2^{n^2}$ networks, only a small percentage of the entire search space is needed by our algorithm to learn the BN structure.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# CHAPTER IV

# Controlling the Information Dissemination Process Using An Epidemic Protocol with Dynamic Fanout

## 4.1. Introduction

*Using epidemic protocols for information dissemination in large-scale systems.* Epidemic protocols adopt a scalable, robust, and probabilistic reliable paradigm for information dissemination in large-scale peer-to-peer communication systems [11], [20], [24]. An epidemic protocol proceeds through asynchronous rounds before the information is reliably disseminated to every node. A round is defined as the time taken for all nodes to disseminate a message to their neighboring nodes. In the basic epidemic protocol, every node within the system is potentially involved in the information dissemination process. A peer node that has delivered a given message will be termed *infective*, or *susceptible*. Basically, every node buffers every message it receives up to a certain buffer capacity and forwards that message a limited number of rounds. The node forwards the message each time to a randomly selected set of nodes. The size of this set is called *fanout*. Many variants of the basic epidemic protocol exist and are typically distinguished by the value of the various parameters: buffer capacity, the number of rounds, and the fanout. The reliability of information delivery depends both on these parameters as well as on the size of the system. The epidemic protocol can also be perceived as a *proactive* mechanism where redundancy and randomization are deployed to prevent potential process and network link failures [14].

The information dissemination process in a peer-to-peer communication system takes place in the same way as an epidemic protocol. The underlying peer-to-peer communication paradigm is the key to the scalability of the dissemination scheme. This paradigm makes the epidemic protocols a robust and scalable means for dissemination of

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

information. To successfully implement an epidemic protocol in a peer-to-peer system, three design issues need to be carefully addressed:

- Membership—how do processes get to know each other, and how many do they need to know?
- Topology Awareness—how do we make the connections between processes reflect the actual network topology such that the performance is acceptable?
- Message Filtering—how do we take into account the actual interest of processes and decrease the probability that they receive and store information of no interest to them?

*The need for epidemic protocols with dynamic fanouts.* The use of epidemic algorithms has been explored in applications such as reliable multicast [31], failure detection [58], data aggregation [15], [23], resource discovery and monitoring [57], and database replication [11]. Each of these experimentations implements different variations of the basic epidemic protocol. For instance, in [23], the aggregation protocol is based on the simple "push-pull gossiping" scheme. In this scheme, every node executes two different threads. The active thread periodically initiates an *information exchange* with one random neighbor. The passive thread waits for messages sent by one sender. We observe that in most of the epidemic protocols, fanout is kept a constant that is based on the assumption that every peer possesses uniform and independent membership. This assumption is not valid in practical applications where every peer cannot be expected to be infected independently of other peers. Moreover, a constant fanout is only applicable in scenarios where the total number of infected nodes in every round is immaterial, but this may not be the case when a controlled infection pattern is expected. The infection pattern is a frequency distribution specifying the number of nodes that are expected to be infected at the end of a round. By controlling the infection pattern, the end user has a better control over the overall latency and message overhead of the information dissemination process.

*Contributions.* In this chapter, we introduce a model for dynamic fanout based epidemic protocol to aid in fine-grained control of the information dissemination process. We present two approaches to quantify the fanout and thereby ensure all nodes receive a specific message within a bounded time. The first approach is called *Round Based*

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

*Dynamic Fanout*, in which each node transmits a message with a varied fanout from round to round. In this approach, the network topology is flat with no hierarchy. The fanout values are quantified based on the infection pattern and redundancy message pattern over rounds. In the second approach, *Cluster Based Dynamic Fanout*, we shift from a flat membership network to a cluster membership network. In the first approach, the fanout remains constant during a round. In the second approach, nodes are clustered based on geographical proximity criterion and we vary the fanout between clusters of nodes. This implies that during each round, nodes in different clusters disseminate information using different fanout values. In both approaches, the number of messages generated is bounded by $O(n \log n)$, where $n$ represents the total number of nodes in the system. In spite of ensuring users controlled information dissemination, the lower bound on the message overhead for both the approaches is the same as the one achieved in epidemic protocols with a constant fanout.

*Applications of the proposed epidemic protocols*. Our dynamic fanout model is best suited for distributed computing applications in which a global function within a pre-determined response time needs to be computed. One such application is distributed knowledge discovery where the global knowledge discovery model is constructed from local knowledge discovery models learnt at each peer node. The accuracy and speed of the global knowledge discovery model construction process is dependent on the underlying controlled information dissemination process. In the presence of a dynamic data environment, the knowledge discovery process must proceed according to the availability of data in peer nodes. An epidemic protocol with constant fanout will not guarantee the completion of the process according to pre-determined response time requirements. Also, in a dynamic data environment the constant fanout would hinder the accuracy of the model. To provide a reliable guarantee, a dynamic fanout based epidemic protocol is required. The pre-determined response time requirements can be fitted into an exponential distribution based on the knowledge of availability of relevant data and computing power in each peer node. The values from the exponential distribution can be used to compute the infection pattern. This infection pattern can then be provided as an input to the underlying epidemic protocol. Thus, in every round of the knowledge

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

discovery process only peer nodes with relevant data items will participate in the information dissemination process

*Roadmap.* Section 4.2 presents the first approach, Round Based Dynamic Fanout. Section 4.3 presents the second approach, Cluster Based Dynamic Fanout. For both approaches, we provide the theoretical analysis to the model. In the analysis of the first approach, we present the dynamic fanout equations and the lower bound on the overall message overhead. In the analysis of the second approach, we show how the nodes with dynamic fanout get infected based on a beta-distribution model. Section 4.4 gives some simulation results supporting the analysis. Section 4.5 concludes the chapter.

## 4.2. ROUND BASED DYNAMIC FANOUT

In this section, we present the first approach for implementing dynamic fanout based epidemic protocol, followed by a detailed mathematical analysis. Among the three design issues, we address the membership issue by adopting Newscast [24], an epidemic-based membership manager protocol. The main goal of Newscast is to continuously reconnect the (logical) connections between peers. The reconnection process is designed in such a way that the resulting peer-to-peer network is very close to a random graph. As we adopt a flat-membership approach, the network topology is flat in nature without any hierarchy. Message filtering is not performed at every node and thereby even redundant and duplicate messages are processed during every epidemic instance.

### 4.2.1 The Round Based Dynamic Fanout Approach

The basic idea underlying our approach, inspired by the work presented in [13], is as follows. Each peer maintains a fixed size view of member peers. The view is sorted according to network distance estimates. Therefore, the first position in the view holds the closest peer known so far. At protocol initialization phase, views need to be initialized with a random sample of nodes taken from the whole peer-to-peer network. For this purpose, we use Newscast to build and maintain an approximately random-graph overlay topology. In order to evolve the topology, peers exchange views in an epidemic fashion.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Periodically, each peer actively selects a neighbor from its view and starts a view exchange process (see pseudo-code in Fig. 4.1). Each peer node execution implements two threads. The active thread is shown in Fig. 4.1a and the passive thread is shown in Fig. 4.1b. In the active thread, every peer node picks the set of random neighbors based on the round-based fanout. The round-based fanout is computed based on user specifications, which are specified as an infection pattern over rounds. The peer node sends the message and its local view to this set of random neighbors. Once the remote peer's view has been received, it is merged with the local one. This merge operation preserves the ordering of the local view. Therefore, newly received member peers are sorted in accordance with their network distance estimates.

---

**do forever**

*// Wait for finite interval of Δt, which is equal to the time taken for a round to be completed*

wait(Δt);

// Roundfanout retrieves the fanout to be used in the current round. The round based fanout is //computed based on the user specified infection pattern.

fanout= Roundfanout(current round)

//Using the SELECTPEERS method, the neighbor list is populated with neighbors. The size of the neighbor list is equal to the current round-based fanout.

Neighbors[fanout] = SELECTPEERS();

//Using the SENDMESSAGE method, message is transmitted to the neighbor list.

SENDMESSAGE(Neighbors[fanout]);

// The view of the current node is sent to the neighbors list

SENDSTATE(myview, Neighbors[fanout]);

// Receive the view from the node who sent its view

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
n_state = RECEIVESTATE();

//Merge the current view wit the received view in a temporary list.
my_state.UPDATE(n_state);
```

(a) Active Thread

---

**do forever**

```
n_state = RECEIVESTATE();
SENDSTATE(n_state.sender);
my_state.UPDATE(n_state);
```

(b) Passive Thread

---

Fig.4.1 Round based Dynamic Fanout pseudo-code

In the following analysis, we assume that the composition of the network does not vary during the run and we observe the transmission of a single message from a peer. Each peer participates in the gossip process via synchronous rounds. During each round, each peer has an independent, uniformly distributed, random view of known peers. Thus, peers have a global membership view and epidemic targets are picked from this global view uniformly and randomly. We also assume two kinds of failures affecting our system. They are *message loss* and *peer crash*. Both failures are assumed to be stochastically independent. All nodes are assumed to have the same failure probabilities. The values of $\tau$ and $\epsilon$ are the same as the corresponding values in [23]. Neither the recovery of crashed peers nor Byzantine failures are taken into consideration,. We assume that redundant messages are generated every round. We estimate the redundant message factor $\beta_r$ from our simulations discussed in the next section.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## Notations

$n$ Number of peers in a peer-to-peer network

$r$ A single epidemic round

$R_{\max}$ The maximum number of rounds

$\epsilon$ Probability of a message loss

$k$ Number of peer crashes in a round

$\tau$ Probability of a peer crash during a single round. $\tau = k/n$

$\beta_r$ Probability of redundant messages during a single round

$I_r$ Number of susceptible peers that are infected by a message sent from an infective during round $r$. $I_0 = 1$, indicating at round 0 there is only one peer with a message

$S_r$ Number of peers that are not infected in the network after the end of round $r$. At the beginning of an epidemic period, $S_0 = n-1$ and we expect the epidemic period to end with a high probability of $S_{R_{\max}} = 0$.

$F_r$ The fanout associated with every peer, i.e., a peer that receives a message during round $r$ will transmit the message to $F_{r+1}$ peers in round $r+1$.

### 4.2.2 Mathematical Analysis

The main objective of our analysis is to measure the adaptive fanout value in every round and the number of messages generated after the epidemic protocol successfully terminates. The first step of our analysis is to estimate the distribution of $S_r$ and $\beta_r$ over the $R_{\max}$ rounds. We plan to associate values for both distributions with an exponential rule. The constraints of the exponential rule are $S_0 = n-1, S_{R_{\max}} = 0$, and $\beta_0 = 0, \beta_{R_{\max}} = thresh$, where *thresh* is the maximum probability of redundancy allowed in the network. After estimating $S_r$ and $\beta_r$, the corresponding fanout values for each round $r$ can be determined.

Our analysis is based on the chain-binomial based recurrence relation [1] [10], which has been derived from epidemic models and successfully applied to epidemic

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

protocols in peer-to-peer networks. According to Eugster et.al.[13], the lower bound on the probability that a given susceptible peer is infected by a message is given by

$$p = \left(\frac{F}{n-1}\right)(1-\epsilon)(1-\tau)$$

where $F$ is the constant fanout size. In our approach, the fanout varies from round to round. We conjecture that the equation for probability of infection $p_r$ during the individual round remains the same. In the analysis provided in [23] redundant or duplicate messages were assumed to be discarded by the peer and their impact on the overall message overhead was not considered. After incorporating the variable fanout $F_r$ and the probability of redundancy in the above equation, the probability that a given susceptible peer is infected by a given message in round $r$ is given by:

$$p_r = \left(\frac{F_r}{n-1}\right)(1-\epsilon)(1-\tau)(1-\beta_r) \tag{4.1}$$

Let $q_r = 1 - p_r$ be the probability that a given susceptible peer is not infected by a given gossip message in round $r$. Also the probability that a given susceptible peer is not affected by the presence of $I_r$ infected peers is ${q_r}^{I_r}$. Along the same lines, we can derive that the number of peers which would not be infected by the message in the round $r$+1 is:

$$S_{r+1} = S_r {q_r}^{I_r} \tag{4.2}$$

Substituting (1) into (2) we get

$$S_{r+1} = S_r\left[1-\left(\frac{F_r}{n-1}(1-\epsilon)(1-\tau)(1-\beta_r)\right)\right]^{I_r} \tag{4.3}$$

After some algebraic manipulations, the fanout value for round $r$ can be computed by

$$F_r = \left[\frac{n-1}{(1-\epsilon)(1-\tau)(1-\beta_r)}\right]\left[1-\exp\left(\frac{\ln\left(\frac{S_{r+1}}{S_r}\right)}{I_r}\right)\right] \tag{4.4}$$

To compute the value of $I_r$, we use the following simple relationship

$$I_r = S_{r-1} - S_r \text{ , for } 1 \le r \le R_{\max} \tag{4.5}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The fanout values computed for every round will ensure that the message will reach every peer. Next .we compute the lower bound on the message overhead of our adaptive fanout approach. The delay involved in the fanout calculation is only in the initial setup of the values for $S_r$ by the user. Once the user specifies the expected values for $S_r$, there is no additional involvement of the user in the fanout calculation.

### 4.3.3 Message Overhead

In our approach, the number of messages generated by a peer during each round is not constant and is determined by the value of $F_r$. At the beginning of round $r$, each peer transmits $F_r$ new messages into the network and $I_r$ peers are participating in the message transmission process. So the number of new messages generated in round $r$ is $F_r \times I_r$.

Considering $n >> F_r$, in practical peer-to-peer systems, Eq. (4.3) thus becomes:

$$S_{r+1} = S_r \exp\left[-\left(\frac{F_r}{n-1}(1-\epsilon)(1-\tau)(1-\beta_r)\right)\times I_r\right] \tag{4.6}$$

Applying logarithms and algebraic manipulations we get an expression for $F_r \times I_r$ as follows,

$$F_r I_r = \left[\frac{n-1}{(1-\epsilon)(1-\tau)(1-\beta_r)}\right]\log\left(\frac{S_r}{S_{r+1}}\right) \tag{4.7}$$

Summing up Eq. (4.7) for all permissible rounds, the total number of messages generated in the network is given as:

$$M_{R_{\max}} = \left[\frac{n-1}{(1-\epsilon)(1-\tau)(1-\beta_r)}\right]\sum_{r=0}^{R_{\max}-1}\log\left(\frac{S_r}{S_{r+1}}\right) \tag{4.8}$$

After the summation process, Eq. (4.8) reduces to

$$M_{R_{\max}} = \left[\frac{n-1}{(1-\epsilon)(1-\tau)(1-\beta_r)}\right]\log\left(\frac{S_0}{S_{R_{\max}}}\right) \tag{4.9}$$

Eq. (4.9) can be approximated as

$$M_{R_{\max}} = \left[\frac{n-1}{(1-\epsilon)(1-\tau)(1-\beta_r)}\right]\log(n-1) \tag{4.10}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Thus, the message overhead is bound by $O(n \log n)$, which is similar to other constant fanout based epidemic protocols.

## 4.3. Cluster Based Dynamic Fanout

In the first approach the nodes were connected by a flat membership, where the local subscription list is composed of nodes located all over the network. In our second approach, we adopt a cluster model where nodes are clustered according to a geographical proximity criterion. In the first approach, the fanout for a given node changes from round to round but remains constant within a round. We relax this constant round-based fanout assumption in our second approach. The three design issues are handled in the same fashion as the first approach. The pseudo-code for the information dissemination process in this approach is similar to Fig. 4.1. The only difference is the fanout values and the view list for each node.

In the cluster based dynamic fanout approach, the probability $p$ that a given susceptible peer is infected by a message follows a frequency distribution during every round. There are two ways in which $p$ can be varied: vary $p$ among each node in the network or vary $p$ between clusters. We chose to vary $p$ between the clusters because our network topology consists of heterogeneous clusters, that may include workstation clusters composed of machines with different processor architectures, data formats, and operating system environments. It is arguable that the probability of infection should vary between the nodes within a cluster too, but such a level of detail may make the model intractable. In the following analysis, we present an agreement between our hypothesis and observed data.

### 4.3.1 Network Topology

The topology of the network studied is shown in Fig. 4.2. The figure shows a two level hierarchy for ease of analysis, and can be easily extended to a hierarchy of more levels. The figure shows two types of nodes; empty circles represent internal nodes, while solid circles represent external nodes. Internal nodes have a local subscription list composed

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

exclusively of nodes belonging to the same cluster; external nodes are provided with remote subscription list consisting of nodes in other clusters. The presence of these two types of nodes in our network topology leads to two kinds of fanout: intra-cluster fanout and inter-cluster fanout. The intra-cluster fanout denotes the constant number of links each internal node has with member nodes in the same cluster. In our topology, this fanout remains constant within each cluster, but the intra-cluster fanouts vary between any given pair of clusters. In Fig. 4.2, clusters $C_1$ and $C_2$ have a fanout of 2, while clusters $C_3$, $C_4$, and $C_5$ have a fanout of 3. The inter-cluster fanout denotes the number of remote links that each external node must maintain with other external nodes. In our topology, this fanout is a constant of 1.

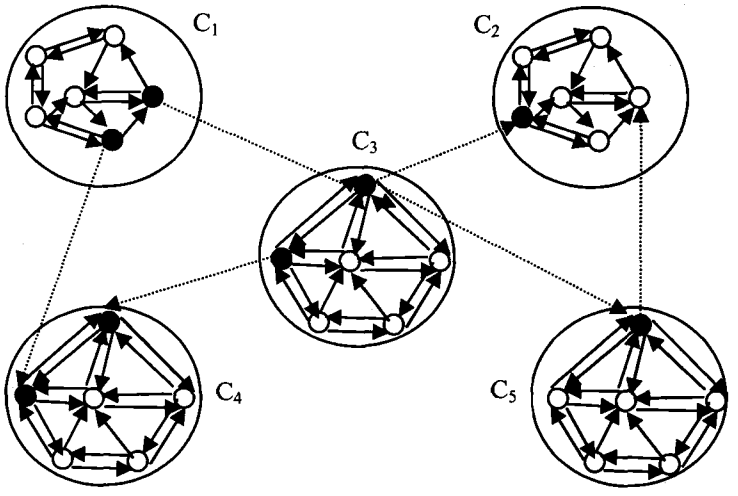


Fig. 4.2 Cluster membership depicting intra-cluster fanout and inter-cluster fanout.

### 4.3.2 Analytical Model

The basis for our analytical model is the variation of the intra-cluster fanout. To model this variation we focus on varying the value of $p$. There are different ways to model the variation of $p$. We chose to adopt a simple way to take into account variation of $p$, by introducing a suitable distribution for $p$ and then estimating the parameters in the distribution and testing goodness-of-fit. We assume $p$ is a random variable following a

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

beta distribution. We prefer to use a beta distribution because an exhaustive analysis would lead to a very complex Markov Chain with an impractical size.

As our analyses are based on the *chain-binomial model*, we define the term "chain" before we proceed with our analysis. At each stage of an epidemic, there are certain numbers of infective and susceptible nodes, with the latter yielding new infective nodes at next stage distributed in a binomial series. Thus we have a chain of binomial distributions. An instance of a chain in a cluster is denoted by:

$(S,I)_{[0,R_{\max}]} \equiv \{(S,I)_r : r = 0,1,...,R_{\max}\}$.

For example. if all the nodes in a 5-node cluster have been infected in 3 rounds then a possible chain sequence would be $(4,1)_{[0,3]} \equiv \{(4,1)_0,(2,3)_1,(1,4)_2,(0,5)_3\}$.

The probability of occurrence of a chain in a cluster can be expressed as follows:

$$P(p) = P((S,I)_{[0,R_{\max}]} \mid p). \tag{4.11}$$

Since we assume that $p$ varies between clusters only, we compute the required expectations by averaging the probability of frequencies of every kind of chain $P(p)$ over all possible values of $p$ in the limit (0, 1), which gives the following,

$$\int_{p=0}^{1} P(p)f(p)dp. \tag{4.12}$$

The density function is given by:

$$f(p) = \frac{p^{a-1}(1-p)^{b-1}}{B(a,b)}, 0 \le p \le 1, \tag{4.13}$$

where $B(a,b)$ is the beta function and $a > 0$, and $b > 0$.

The probability $P(p)$ is derived from the chain-binomial model [5] as:

$$P(p) = P(S_{r+1} = j \mid S_r = i) = \binom{n-i}{j-i}(1-q^i)^{j-i}q^{i(n-j)}, j \ge i. \tag{4.14}$$

In Eq. (4.14), $q = 1-p$ represents the probability that a susceptible peer is *not* infected by a given message. By integrating over all values of $p$ where $p$ has the beta density in Eq. (4.13), the expectations can be shown as functions of parameters *a, b, i ,j*.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

$$\begin{aligned}\int_0^1 P(p)f(p)dp &= \int_0^1 \binom{n-i}{j-i}\frac{(1-q^i)^{a+j-i-1}q^{b+i(n-j)-1}}{B(a,b)}dp \\ &= \binom{n-i}{j-i}\frac{B(a+j-i,b+i(n-j))}{B(a,b)} \\ &= \frac{(n-i)!}{(j-i)!(n-j)!}\cdot\frac{(a+j-i-1)...a(b+i(n-j)-1)...b}{(a+b+j-1)(a+b)}.\end{aligned} \tag{4.15}$$

Thus in our chain we are required to record only the values of susceptibles and infectives $(S_r, I_r) = (s_r, i_r)$ the probability $P(p)$ of this chain equals

$$\begin{aligned}P(p) &= \binom{n-s_0}{s_1-s_0}(1-q^{s_0})^{s_1-s_0}q^{s_0(n-s_1)}...\binom{n-s_{R_{\max}-1}}{s_{R_{\max}}-s_{R_{\max}-1}}(1-q^{s_{R_{\max}-1}})^{s_{R_{\max}}-s_{R_{\max}-1}}q^{s_{R_{\max}-1}(n-s_{R_{\max}})} \\ &= \prod_{r=1}^{R_{\max}}\binom{n-s_r}{s_{r+1}-s_r}(1-q^{s_r})^{s_{r+1}-s_r}(q^{s_r})^{n-s_{r+1}} \\ &= \left[\prod_{r=1}^{R_{\max}}\binom{n-s_r}{s_{r+1}-s_r}(1-q^{s_r})^{s_{r+1}-s_r}\right]q^{\sum_{r=1}^{R_{\max}}s_r(n-s_{r+1})} \\ &= \left[\prod_{r=1}^{R_{\max}}\binom{n-s_r}{s_{r+1}-s_r}\sum_{m=0}^{n-s_{r+1}}\binom{n-s_{r+1}}{m}(-1)^m q^{ms_r}\right]q^{\sum_{r=1}^{R_{\max}}s_r(n-s_{r+1})}.\end{aligned} \tag{4.16}$$

Combining Eq. (4.15) and Eq. (4.16), we integrate $P(p)$ over all values of $p$, which gives

$$\begin{aligned}E[P(p)] &= \int_0^1\binom{n-s_r}{s_{r+1}-s_r}(1-q^{s_r})^{s_{r+1}-s_r}(q^{s_r})^{n-s_{r+1}}\frac{(1-p)^{a-1}p^{b-1}}{B(a,b)}dp \\ &= \binom{n-s_r}{s_{r+1}-s_r}\sum_{k=0}^{n-s_{r+1}}\binom{n-s_r}{k}(-1)^k\int_0^1\frac{p^{s_r[k+s_r]+b-1}(1-p)^{a-1}}{B(a,b)}dp \\ &= \binom{n-s_r}{s_{r+1}-s_r}\sum_{k=0}^{n-s_{r+1}}\binom{n-s_r}{k}(-1)^k\frac{B(s_r[k+s_r]+b,a)}{B(a,b)}\end{aligned} \tag{4.17}$$

Eq. (4.17) provides a mechanism to measure the expected value of the probability of a particular chain. The next step is to estimate the parameter values of *a,* and *b*. We estimate these values based on the data collected from simulation experiments. For example, in one sample simulation scenario, we setup a network topology of 100 clusters, each with a membership of 5 nodes. The initial sequence in each cluster would comprise of the following chain $S_0 = 4$ and $I_0 = 1$. In our example, the expected number of clusters with the chain (1, 3, 0) equals

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

$$100\int_0^1 4p^4(1-p)^3\frac{(1-p)^{a-1}p^{b-1}}{B(a,b)}dp$$
$$=400[\frac{B(a+3,b+4)}{B(a,b)}] \quad (4.18)$$

Similar integrations for other more probable chains based on the simulation data will be carried out and the estimation of the parameters $a$, and $b$ will be done by applying maximum likelihood methods.

## 4.4. Simulations and Analysis

In this section, we compare the analytical results obtained for both approaches with simulation results. For the round based dynamic fanout approach, our results highlight the impact on infection pattern and message overhead in presence of variable fanouts from round to round. For the cluster based dynamic fanout approach, our results show the relation between the infection pattern of clusters and the beta-distribution characterized probability of infection model. We conducted the simulations using peersim, an open source peer-to-peer simulator developed at the University of Bologna [22][49].

### 4.4.1 Round Based Dynamic Fanout

In our simulations, we have used Newscast as the underlying overlay network membership protocol. The reason for this choice is twofold: First, we want to show empirical results in a realistic overlay network that can actually be built in a decentralized way. Second, Newscast is known to be robust and capable of maintaining a sufficiently random network in the failure scenarios. In Fig. 4.1a, Newscast has been used to implement the SELECTPEERS function. We have performed our simulations on network sizes ranging from 500 nodes to 2,500 nodes. The size of the local neighbor sets at each peer node, that are maintained and exchanged by the NEWSCAST protocol is set to 1 % of the overall network size. This value is large enough that a given message is able to reach all nodes in the network. For our approach to be effective we rely on the user for the infection pattern. In our simulations, we have tackled the issue of designing good user inputs by adopting an exponential distribution rule. Using this rule the distribution of the non-infection pattern $S_r$ and the redundancy probability $\beta_r$ was computed. In computing

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

the distribution for $\beta_r$, the *thresh* value is set to 0.3. The values for $\tau$ and $\epsilon$ are set to 0.05 and 0.01, respectively. We have conducted 20 simulation runs Using equation (4), the fanout values were computed for each round. To incorporate fractional values in our simulation runs, we either converted the real number to a lower or higher integer value randomly in different runs such that the average value is equal to the real number. Table 1 presents the analytical and simulation values for non-infection pattern $S_r$ with network sizes of 1,000 and 2,000 nodes. As is evident from Table 1, the option of variable fanouts from round to round allows users to control the infection pattern. In addition, the user can specify the total number of rounds for a specific infection pattern and the fanouts are computed accordingly. Though the $S_r$ values for individual runs deviated from their corresponding analytical values, but the average over 20 runs converged to the theoretical values. Table 4.1 summarizes the average values for $S_r$ over a set of runs and the standard deviation of the observed simulation results. The simulation values for $S_r$ do confirm to our analytical model.

Table 4.1 Round based dynamic fanout for 1,000 and 2,000 nodes

| Rounds | 1000 | | | 2000 | | |
|---|---|---|---|---|---|---|
| | $S_r$ (Analysis) | $S_r$ (avg). (Simulation) | Std Dev | $S_r$ (Analysis) | $S_r$ (avg.) (Simulation) | Std Dev |
| 1 | 999 | 999 | 0 | 1999 | 1999 | 0 |
| 2 | 994 | 992.4 | 0.7 | 1994 | 1991.5 | 0.7 |
| 3 | 961 | 959.3 | 0.5 | 1961 | 1959.4 | 0.5 |
| 4 | 771 | 770.5 | 0.75 | 1728 | 1725.3 | 0.75 |
| 5 | 210 | 209.5 | 2.6 | 969 | 965.5 | 2.6 |
| 6 | 1 | 0.5 | 0 | 52 | 50.3 | 0.55 |
| 7 | - | - | | 1 | 0.5 | 0 |

Table 4.2 presents the computed dynamic fanout values for network sizes of 1,000 and 2,000 nodes. The fanouts are computed for each round.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Table 4.2 Round based dynamic fanout for different network sizes

| Rounds | Nodes | | | | |
|---|---|---|---|---|---|
| | 500 | 1000 | 1500 | 2000 | 2500 |
| 1 | - | - | - | - | - |
| 2 | 5 | 5 | 5 | 5 | 3 |
| 3 | 4.089 | 4.46 | 4.44 | 4.27 | 3.5 |
| 4 | 4.47 | 5.79 | 13.68 | 10.848 | 3.785 |
| 5 | 7.76 | 6.32 | 2.33 | 3.08 | 7.33 |
| 6 | 10.86 | 8.37 | 4.94 | 5.56 | 4.19 |
| 7 | - | - | - | - | 3.4016 |
| 8 | - | - | - | - | 4.99 |
| 9 | - | - | - | - | 15.875 |

Table 4.3 highlights the message overhead in the round-based dynamic fanout approach under different network sizes. The third column shows the cumulative message overhead at the end of all rounds generated by the simulations. Column 4 shows the theoretical values for message overhead computed based on the analytical results from Eq. (4.10). We observed that the overall message overhead is independent of the variations in fanout, but is dependent on the percentage of redundant messages in the network. As we can observe from Table 4.3, as the number of nodes increases, the message overhead deviate from the analytical values. This deviation is largely due to the fact that as network size gets larger, the percentage of redundancy messages also increases.

Table 4.3 Message Overhead in the Round based Dynamic Fanout approach

| Number of Nodes | Maximum Rounds | Message Overhead (simulations) | Message Overhead (Analysis) |
|---|---|---|---|
| 1,000 | 6 | 3540 | 3000 |
| 1,500 | 7 | 5624 | 4764 |
| 2,000 | 7 | 7795 | 6602 |
| 2,500 | 9 | 10031 | 8494 |

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

### 4.4.2 Cluster Based Dynamic Fanout

For this approach, we have run experiments on two scenarios. In the first scenario, our network topology was comprised of 100 clusters, with each cluster consisting of four nodes. In the second scenario, the total number of clusters remained the same, but the number of nodes in each cluster was increased to five. As the infection probability varies for each cluster, we observe the infection pattern over rounds. The infection pattern for each cluster is represented as a binomial chain. For example, in the first scenario, a cluster with four nodes, where one node is infected ($I_0 = 1$) and three nodes are not yet infected ($S_0 = 3$), we record the number of new infected nodes $\{I_r\}$ at $r = 0,1,2,...$ rounds as the chain 1-1-2, where all nodes are eventually infected. In Table 4.4, the first column shows all the possible infection pattern chains for a cluster of size four. The chain-binomial probabilities for all of these chains $\{I_0 = 1, I_1, ..., I_k\}$ are given in the second column, where $n$ is the total number of clusters. Our goal is to find out the total number of clusters which have the same infection pattern. Each entry in column two shows the total number of clusters for the corresponding chain in the first column. We need estimates of $p$ and $z$, to find out the total number of clusters for each infection pattern chain. The third column shows the expected probabilities of the chain-binomial model. The expected probabilities are presented by simpler notations as follows:

$$
\begin{aligned}
z &= (a+b)^{-1} \\
p &= a/(a+b) \\
z_1(n) &= \prod_{i=0}^{n}(1+iz) = z(n) \\
z_q(n) &= \prod_{i=0}^{n}(q+iz) \\
z_p(n) &= \prod_{i=0}^{n}(p+iz)
\end{aligned}
\tag{4.19}
$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Table 4.4 Chain- Binomial Probabilities for cluster size of 4

| Infection Pattern $\{I_r\}$ $r=0\ 1\ 2\ 3$ | Probabilities | Expected Values of Probabilities |
|---|---|---|
| 1-1-1-1 | $6np^3q^3$ | $6z_q(2)z_p(2)/z(5)$ |
| 1-1-2 | $3np^3q^2$ | $3z_q(1)z_p(2)/z(4)$ |
| 1-2-1 | $3np^3q$ | $3z_q(0)z_p(2)/z(3)$ |
| 1-3 | $np^3$ | $z_p(2)/z(2)$ |

In Table 4.5, the first column presents the binomial chains and the second column contains the total number of clusters from simulation results. The third column contains the total number of clusters which need to be computed based on the estimates of $p$ and $z$. To find the estimates of $p$ and $z$, we employ a log-likelihood function. The log-likelihood function is constructed based on the expected probabilities in the third column of Table 4 and the simulation generated total number of clusters in the second column of Table 5. The log-likelihood function $Log\ L$ is given as follows,

$$\begin{aligned} LogL = 5*\log(6z_q(2)z_p(2)/z(5)) + 6*\log(3z_q(1)z_p(2)/z(4)) + \\ 19*\log(3z_q(0)z_p(2)/z(3)) + 70*\log(z_p(2)/z(2)) \end{aligned} \tag{4.20}$$

Substituting the values of $z$, $z_q$, and $z_p$ from (4.19), we get

$$\begin{aligned} LogL = C + 30\log q + 11\log(q+z) + 100\log p \\ + 100\log(p+z) + 100\log(p+2z) - 100\log(1+z) \\ - 100\log(1+2z) - 20\log(1+3z) - 11\log(1+4z) \\ - 11\log(1+5z) \end{aligned} \tag{4.21}$$

where $C$ is a constant.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Table 4.5 Simulation and Analytical Values of infected clusters for cluster size of 4

| Infection Pattern $\{i_r\}$ $r=0\ 1\ 2\ 3$ | Observed number of clusters (Simulation) | Fitted values from the analytical model |
|---|---|---|
| 1-1-1-1 | 5 | 3.1 |
| 1-1-2 | 6 | 4.1 |
| 1-2-1 | 19 | 14.8 |
| 1-3 | 70 | 67.3 |

Using MATLAB, the minimization of $LogL$ given in (4.21) is carried out. The value of $C$ has no effect on the minimization process since it is a constant. The maximum likelihood estimates of $p$ and $z$ are found to be

$$\hat{p} = 0.822 \pm 0.028$$

$$\hat{z} = 0.521 \pm 0.178$$

The corresponding estimates of $a$ and $b$ were obtained as

$$\hat{a} = 1.29,$$

$$\hat{b} = 0.3$$

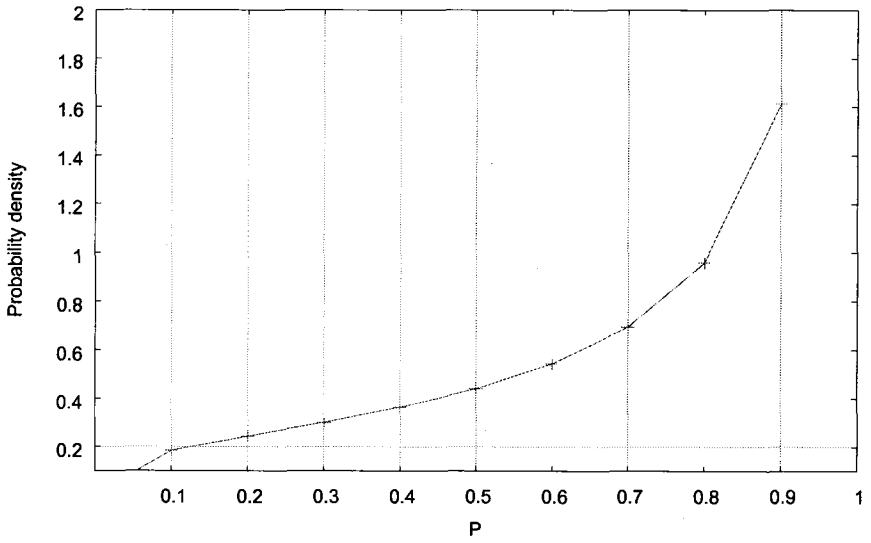


Fig 4.3 Beta-Distribution function for cluster of size 4

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Since $b$ is less than unity, the beta-distribution is $J$-shaped with an infinite ordinate at $p$=1. Substituting the values of $p$, $q$, $z$ into the third column of Table 4.4, the fitted values for our chain-binomial model with variable $p$ are calculated as seen in the third column of Table 5. The fitted values are the total number of clusters for the corresponding infection pattern chain given in the first column.

In the second scenario, we observe the scalability of the cluster based dynamic fanout scheme. We increase the cluster size to 5, which increases the total number of nodes in the system to 500. As in Table 4.4, the first column of Table 4.6 shows all the possible infection pattern chains for a cluster of size 5. The chain-binomial probabilities for all these chains are given in the second column of Table 4.6. The third column shows the expected probabilities of the chain-binomial model.

Table 4.6 Chain- binomial Probabilities for cluster size of 5

| Infection Pattern $\{i_r\}$ $r=0\ 1\ 2\ 3\ 4$ | Expected number of Clusters (analysis) | Expected Values of Probabilities |
|---|---|---|
| 1-1-1-1-1 | $24nq^6p^4$ | $24z_q(5)z_p(3)/z(9)$ |
| 1-1-1-2 | $12nq^5p^4$ | $12z_q(4)z_p(3)/z(8)$ |
| 1-1-2-1 | $12nq^4p^3(1-q^2)$ | $12z_q(3)z_p(3)(1+q+12z)/z(8)$ |
| 1-1-3 | $4q^3p^4$ | $4z_q(2)z_p(3)/z(6)$ |
| 1-2-1-1 | $12nq^4p^3(1-q^2)$ | $12z_q(3)z_p(3)(1+q+12z)/z(8)$ |
| 1-2-2 | $6q^2p^2(1-q^2)^2$ | $6z_q(1)z_p(3)[76z^2+(17+19q)z+(1+q)^2]/z(7)$ |
| 1-3-1 | $4nqp^3(1-q^3)$ | $4z_q(0)z_p(3)[38z^2+(12+9q)z+(1+q)^2]/z(7)$ |
| 1-4 | $p^4$ | $z_p(3)/z(3)$ |

From the third column of Table 4.6 and the simulation data in second column of Table 4.7, the log-likelihood function is given as follows:

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

$$
\begin{aligned}
Log\ L = C &+ 52\log q + 11\log(q+z) + 5\log(q+2z) \\
&+ 4\log(q+3z) + 3\log(q+4z) + \log(q+5z) \\
&+ 100\log p + 100\log(p+z) + 100\log(p+2z) \\
&+ 100\log(p+3z) + 11\log(1+q+12z) \\
&+ 11\log(76z^2 + (17+19q)z + (1+q)^2) \\
&+ 41\log(38z^2 + (12+9q)z + (1+q)^2) \\
&- 100\log(1+z) - 100\log(1+2z) - 100\log(1+3z) \\
&- 52\log(1+4z) - 52\log(1+5z) - 52\log(1+6z) \\
&- 51\log(1+7z) - 4\log(1+8z) - \log(1+9z)
\end{aligned}
\tag{4.22}
$$

where $C$ is a constant.

Using MATLAB, the minimization of *LogL* was carried out. The maximum likelihood estimates of $p$ and $z$ are found to be

$$\hat{p} = 0.8444 \pm 0.0113$$

$$\hat{z} = 0.6122 \pm 0.0221$$

The corresponding estimates of $a$ and $b$ were obtained as

$$\hat{a} = 1.377,$$

$$\hat{b} = 0.253$$

The shape of the beta-distribution is identical to Fig. 4.3, as the value of $\hat{b}$ is less than 1.

Table 4 7 Simulation and Analytical Values of the infected clusters for cluster size of 5

| Infection Pattern $\{i_r\}$ r= 0 1 2 3 4 | Observed number of clusters (Simulation) | Fitted values from the analytical model |
|---|---|---|
| 1-1-1-1-1 | 1 | 0 |
| 1-1-1-2 | 1 | 0 |
| 1-1-2-1 | 1 | 1.5 |
| 1-1-3 | 1 | 0.65 |
| 1-2-1-1 | 1 | 1.5 |
| 1-2-2 | 6 | 10.3 |
| 1-3-1 | 41 | 37.72 |
| 1-4 | 48 | 51.24 |

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## 4.5. Conclusions

We have described two approaches for implementing a dynamic fanout based epidemic protocol: Round based dynamic fanout and Cluster Based Dynamic Fanout. We have derived fanout equations for the round-based dynamic fanout approach and computed the overall message overhead. We used a flat membership scheme to implement the first approach and a cluster based network topology to implement the second approach. In the first approach, the number of infected processes per round is based on the frequency distribution provided by a system user. This helps users quantify fanout to control the epidemic infection and allows users to fine tune this parameter according to their requirements. In the Cluster Based Dynamic Fanout approach, the fanout is not constant among the various clusters unlike the round-based dynamic fanout approach. We have observed that the infection pattern among the various clusters during every round. Simulation results for cluster size four and five have proved that the infection pattern closely follows a beta distribution.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# CHAPTER V

# A Majority Based Consensus Methodology For Learning Bayesian Network Structure From Distributed Databases

## 5.1. Introduction

Distributed data mining deals with the problem of data analysis in environments with distributed data, computing nodes, and users. Bayesian network (BN) is a popular and effective model for performing data mining. A BN is a probabilistic graphical model to represent uncertain knowledge. To perform distributed data mining, a BN model has to be constructed from data distributed among a network of computing nodes. There are numerous domains in wired and wireless networks where a BN is a more natural and scalable solution. Consider an ad hoc wireless sensor network where the numerous sensor nodes are monitoring some time-critical environmental data. Central collection of data from every sensor node may place a heavy burden over the limited bandwidth wireless channels. Heavy traffic may also be the cause for power drainage in these devices. A distributed architecture for learning BN is an appropriate solution for reducing the communication load and also distributing the battery power more evenly across the different nodes in the sensor network. Similar requirements for distributed computation of data mining can be found in ad hoc wireless networks of mobile devices like PDAs, cell phones, and wearable computers. Potential applications include personalization, collaborative process monitoring, and intrusion detection over ad hoc wireless networks. Thus there is a need for distributed architectures that pay careful attention to the distributed resources of data, computing, and communication in order to consume them in a near optimal fashion.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

In this chapter, we address the problem of learning a BN from a homogeneous database distributed among a peer-to-peer network of computing nodes. The distributed architecture proposed in this chapter is called Majority Based Consensus Methodology. This methodology is analogous to the majority consensus problem in distributed computing [5]. According to this problem, each node has an input, and it is required that the output at each node stabilizes to the majority of these inputs. In our methodology, each node executes a BN learning algorithm, which discovers a *local* BN based on the local database. In addition to this local learning algorithm, each node implements a majority consensus protocol to discover all the links in the *global* BN, that exist in the global database. The two main issues addressed by the protocol are reduction of communication cost by local negotiation, and computation of the global Bayesian network within an appreciable time limit, which is lesser or equal to the time taken to construct a global Bayesian network within a centralized framework.

The two main contributions of this chapter are the *Majority Consensus Protocol* and the *Majority Bayesian Network Learning Algorithm*. The protocol and algorithm work in tandem to discover the global BN at each node in the network. The key strength of our Majority Consensus Protocol is faster convergence with significantly lower message overhead. Also the performance of our majority consensus protocol is not dependent on user configurable parameters. The parameter independence makes the protocol more predictable and robust. The majority consensus protocol is implemented locally and requires no synchronization among the nodes. The impact of the locality of communication leads to faster convergence of the computation of the global Bayesian network model at the expense of lower message overhead. By locality, we imply that each node updates it BN based on the link information provided by a small set of neighbors. The key strength of the Majority Bayesian Network Learning Algorithm is the ability to correctly ascertain the confidence in the cross links discovered by the Majority Consensus Protocol. By using the algorithm the time taken by each node to discover the global BN does not exceed the time taken by a centralized BN learning approach.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The following is the outline of this chapter. Section 5.2 formulates the problem. Section 5.3 describes the majority consensus protocol and majority Bayesian network-learning algorithm in detail. Section 5.4 presents the simulation results and analysis. Finally we conclude in Section 5.5.

## 5.2. Problem Definition

Notations

| | |
|---|---|
| $N$ | Set of nodes in the peer-to-peer network |
| $V$ | Set of variables in the database; $v \in V$ |
| $DB$ | The global homogenous database |
| $DB_i$ | Local database at node $i$ |
| $E$ | Set of edges representing probabilistic relationships among the variables in $DB$ |
| $E_i$ | Set of edges representing probabilistic relationships among the variables in $DB_i$. |
| $BN$ | The global Bayesian network, which can be defined as $(V, E)$ |
| $BN_i$ | The local Bayesian network at node $i$ |

The problem of structure learning of $BN$ is defined as follows. For a complete $DB$ (no missing values) distributed in a peer-to-peer network of $N$ nodes, and a known variable order, the problem is to identify the structure of the $BN$, $(V, E)$, at each node that best matches $DB$. To accomplish this task, we assume that the $DB$ is distributed leading to $DB_i$ present at each node in the peer-to-peer network. The underlying distribution of data in $DB_i$ may or may not be identical across different nodes in the network. Each node has all the variables in the same order.

In Fig. 5.1, the $DB$ is distributed across three peer nodes as $DB_1$, $DB_2$, and $DB_3$, where $v_j$, $j \in [1,9]$, denote the variables in the $DB$. The observations for these variables are available at the three peer nodes. The $BN_1$ is comprised of variables $(v_1, v_2, v_3)$ at node 1, $(v_3, v_4, v_5, v_6, v_9)$ form $BN_2$ at node 2, and $(v_7, v_8, v_9)$ form $BN_3$ at node 3. The solution to the problem is depicted in Fig. 5.2. The figure shows that every node in the network discovers the structure of the $BN$.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

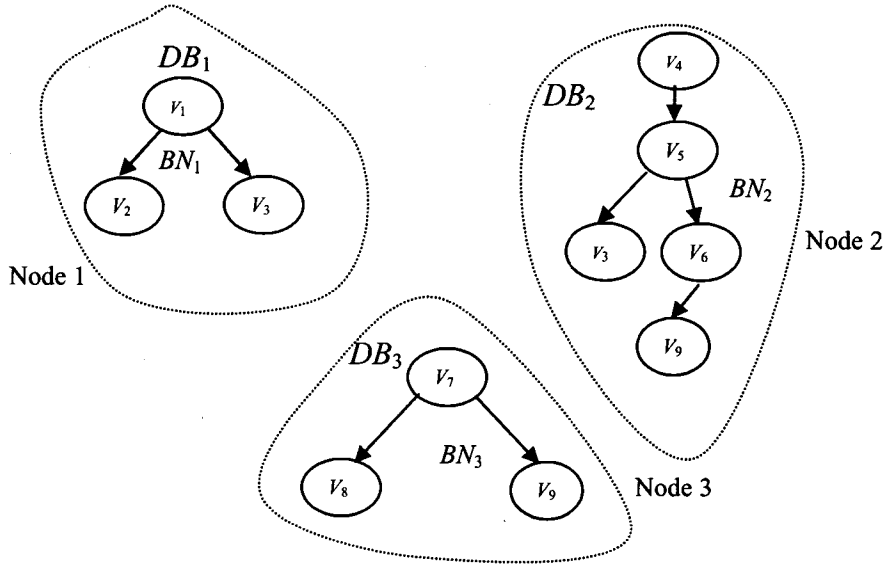


Fig. 5.1: Local Bayesian network models present in three peer nodes

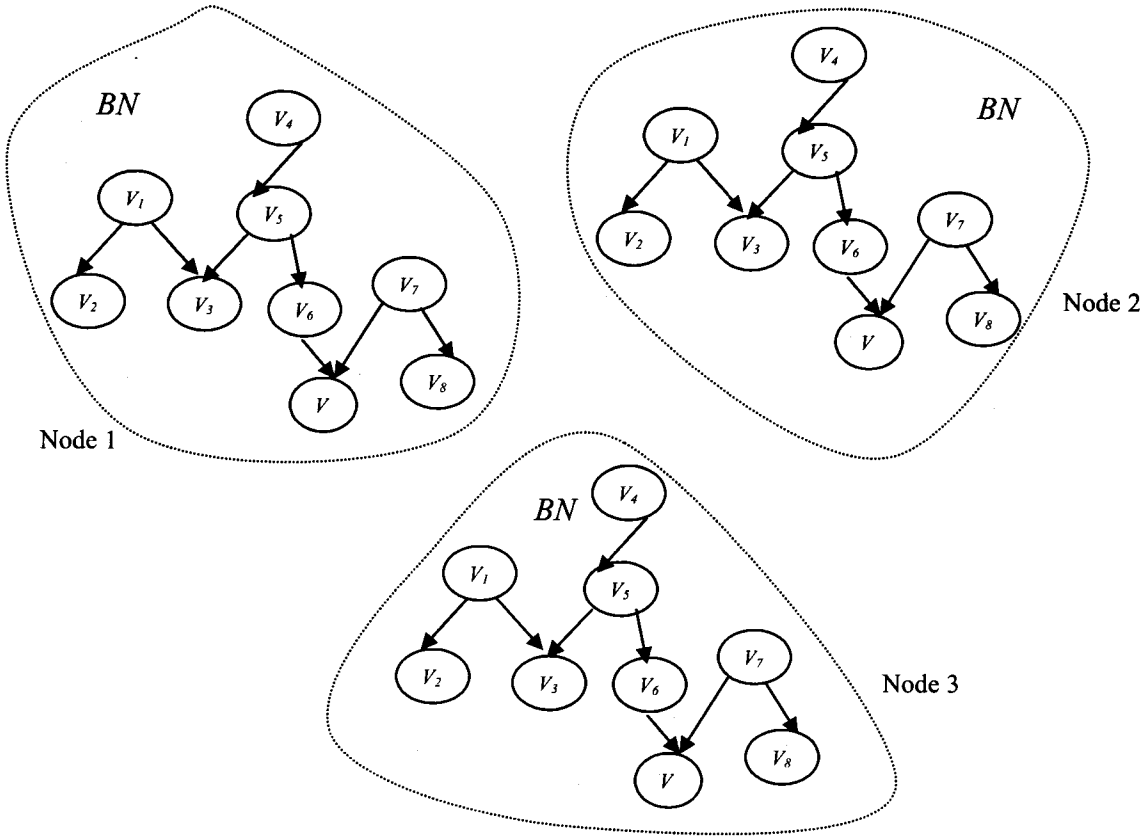


Fig. 5.2: Global Bayesian Network discovered at each node in the peer-to-peer network

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## 5.3. Majority Consensus based Methodology

We now present the consensus-based methodology to learn the structure of *BN* when data is distributed among different peer nodes. The main objective of our methodology is to ensure that all peer nodes in the network converge towards the correct *BN* according to their combined databases. To implement this approach, we develop two components, which execute on each peer node in tandem: *Majority Consensus Protocol* and a *Majority Bayesian Network Learning Algorithm*. The majority consensus protocol specifies the communication mode among the nodes, the frequencies of messages exchanged and the level of coordination required. The majority Bayesian network learning algorithm specifies the minimal amount of information sent by each node, updating the local Bayesian network at each node based on new information, and terminating when a global solution has been reached.

The primary steps in the protocol and the algorithm are summarized as follows:

- At each node, the majority Bayesian network learning algorithm will compute a local Bayesian network based on the variables associated with the data observed on each node.
- In each node the majority Bayesian network learning algorithm computes the confidence of the local Bayesian network using a structure-learning algorithm.
- In the meantime, the majority consensus protocol independently identifies the neighbors for each node. A membership protocol along the lines of Newscast [6] is employed by the consensus protocol to fill up the neighbor list for each node.
- The consensus protocol creates a message packet and transmits the local Bayesian network structure and the confidence level to the neighbors. This process is executed at each node.
- On receipt of the message, the majority Bayesian network learning algorithm at each node updates the local Bayesian network by new edges if the resulting confidence level of the updated Bayesian network is better.
- The above process is repeated until the structure of *BN* is learned at each node.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

In the following sections, we provide the details of the majority consensus protocol and the majority Bayesian network learning algorithm

### 5.3.1 Majority Consensus Protocol

The protocol is based on an optimization problem where all the nodes decide whether they have all the observations necessary to learn the structure of *BN*. The two main processes in the majority consensus protocol are *message construction* and *majority selection.*

*a. Message Construction*

The contents of the message exchanged between peer nodes represent the confidence level in the existence of edges. Here the edges represent the relationship between variables in a Bayesian network, and the confidence levels represent how closely the local Bayesian network represents the local database. The main goals of the message construction process are to encode sufficient information necessary for majority selection and maintain reasonable message length. As the contents of the message contain the confidence level in the edges, the message is represented as a 3-field tuple,($Edg_{ab}$, $conf_{ij}$, $len_{ij}$) where $Edg_{ab}$ represents the relation between any two variables $a$ and $b$ in the database; $conf_{ij}$ is the confidence level that represents the accuracy of the presence of the relation between nodes $i$ and $j$. The confidence levels associated with each edge (relation between variables) are encoded as (0- 24), (25 – 49), (50 – 74), (75 – 100). The $len_{ij}$ represents the count of total records in the database, which were considered in determining the confidence level. The confidence level and record length values are normalized to 100.

*b. Majority Selection*

The main objective of the majority selection process is to ensure that each node converges toward the correct majority. The correct majority refers to the correct confidence level for all the edges. Since the messages exchanged among nodes contain binary information, we measure the *consensus* as the proportion of nodes $i$ whose ad hoc solution agrees with the majority.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Each node has a neighbor list, that has been populated by the Newscast membership protocol. Once a message is constructed, nodes communicate their messages to the neighboring nodes. Each node will record the latest message it sent to its neighboring nodes and the latest message it received from its neighboring nodes. When the system initializes, the message received from the neighboring nodes is initialized to all zeros. After the initialization phase, each node $i$ will send a 3-field tuple message to its neighboring node $j$ containing the values $Edg_{ab}$, $conf_{ij}$, $len_{ij}$.. Based on the message received by each node $i$, a function is therefore calculated for every neighbor $j$. Every time a node receives a message it needs to save the confidence levels in its buffer. When the calculation of the function takes place, it is easy to refer to the confidence levels by the values of $i$ and $j$.

Let $\Pi^i$ and $\Pi^{ij}$ denote the current majority values at node $i$ and the updated majority values received by node $i$ from node $j$, respectively. In the absence of any message transmission, parameters $\Pi^i$ and $\Pi^{ij}$ are initialized to the confidence levels and the record lengths values present in each node for every relation. Based on the message sent to or received from its neighbors and the initialized values of confidence levels and record length values, node $i$ calculates the following two parameters for each edge $Edg_{ab}$:

$$\Pi^i = \sum_{i,j \in E^i} (conf_{ij} - R * len_{ij}) \tag{5.1}$$

$$\Pi^{ij} = (conf_{ij} + conf_{ji}) - R * (len_{ij} + len_{ji})$$

where $E^i$ represents the set of neighboring nodes for node $i$, and $R$ denotes the majority ratio to make a decision on a certain confidence level. If no message was yet received from any neighbor, then $\Pi^i$ represents the majority value represented by the confidence levels computed by the local Bayesian network algorithm at node $i$. Throughout execution the ad hoc output of node $i$ is set according to the majority values in $\Pi^i$. If the sign of $\Pi^i$ is positive, then the relation is accepted. Each time there is a change in the local confidence levels, message is received or a new neighbor connects, $\Pi^{ij}$ and $\Pi^i$ values are calculated. Each node implements the majority selection process independently. Each node $i$ co-ordinates the confidence level in each edge with its

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

neighbor $j$ by maintaining the same values in $\Pi^{ij}$. For every edge, $i$ and $j$ stop exchanging messages as long as the following conditions are true: $\Pi^i \geq \Pi^{ij} \geq 0$, and $\Pi^j \geq \Pi^{ji} \geq 0$.

The pseudocode for the majority selection process is shown in pseudocode 1. From the pseudocode, it is easy to derive that when the protocol dictates that no node needs to send any message, it implies that for every node $j$ the $\Pi^{ij}$ values for the neighbors match with the $\Pi^i$ values. If there is a disagreement on the structure of $BN$, then due to locality there must be disagreement between two immediate neighbors. In such a case, at least one node $j$ must send its $\Pi^i$ values to its neighbor $i$. The number of message interactions is bound by the size of the system. This implies that the protocol always reaches consensus in a static state.

---

**Pseudocode 1** Majority Selection Process

**// Inputs**

*Newscast*($E^i$) // Newscast membership function populates neighbor list $E^i$

*Getmajorityratio*($R$) // Get majority ratio from the user (configurable parameter)

**//Initialization**

$\Pi^i = 0$

*for every* $i, j \in E^i$

$\Pi^{ij} = 0$

$conf_{ij} = 0$

$len_{ij} = 0$

**//Output on Reception of Message**

*if* *MessageRecv*($conf_{ij}, len_{ij}$) // Message arrived on link ($ji$) from node $j$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

$$\Pi^i = \sum_{i,j \in E^i} (conf_{ij} - R * len_{ij})$$

$$\Pi^{ij} = (conf_{ij} + conf_{ji}) - R * (len_{ij} + len_{ji})$$

*for every* $i, j \in E^i$

*if* $((\Pi^{ij} < 0)$ and $(\Pi^{ij} < \Pi^i))$ or $((\Pi^{ij} > 0)$ and $(\Pi^{ij} > \Pi^i))$

$$conf_{ij} = \sum_{l, i \neq i, j \in E^i} conf_{li}$$

$$len_{ij} = \sum_{l, i \neq i, j \in E^i} len_{li}$$

*MessageSent*$(conf_{ij}, len_{ij})$ // Send message to node $j$ from node $i$

---

Fig. 5.3 is a graphical illustration of the majority selection process. In Fig. 5.3, $R$ is set to 0.5. In Fig. 5.3a node $i$ and the values for parameters $\Pi^{ij}, \Pi^{il}, \Pi^{ik}$, and $\Pi^i$ are depicted. Prior to this state, nodes $j$, $k$, and $l$ have exchanged messages with node $i$ and node $i$ has computed the values based on equation (5.1). In Fig. 5.3b, a message is sent from node $k$ to node $i$. The result of this operation causes node $i$ to compute values for $\Pi^{ik}$ and $\Pi^i$. The values of $\Pi^{ik}$ and $\Pi^i$ reduce, but no additional messages are sent to nodes $j$ and $l$ as $\Pi^i$ is equal to $\Pi^{ij}$ and $\Pi^{il}$. In Fig. 5.3c, node $k$ sends another message to node $i$. On reception of this additional message, values of $\Pi^{ik}$ and $\Pi^i$ get further reduced. By now, the conditions of $\Pi^i < \Pi^{ij}$ and $\Pi^i < \Pi^{il}$ are satisfied, resulting in node $i$ sending messages to nodes $j$ and $l$ (Fig. 5.3d).

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

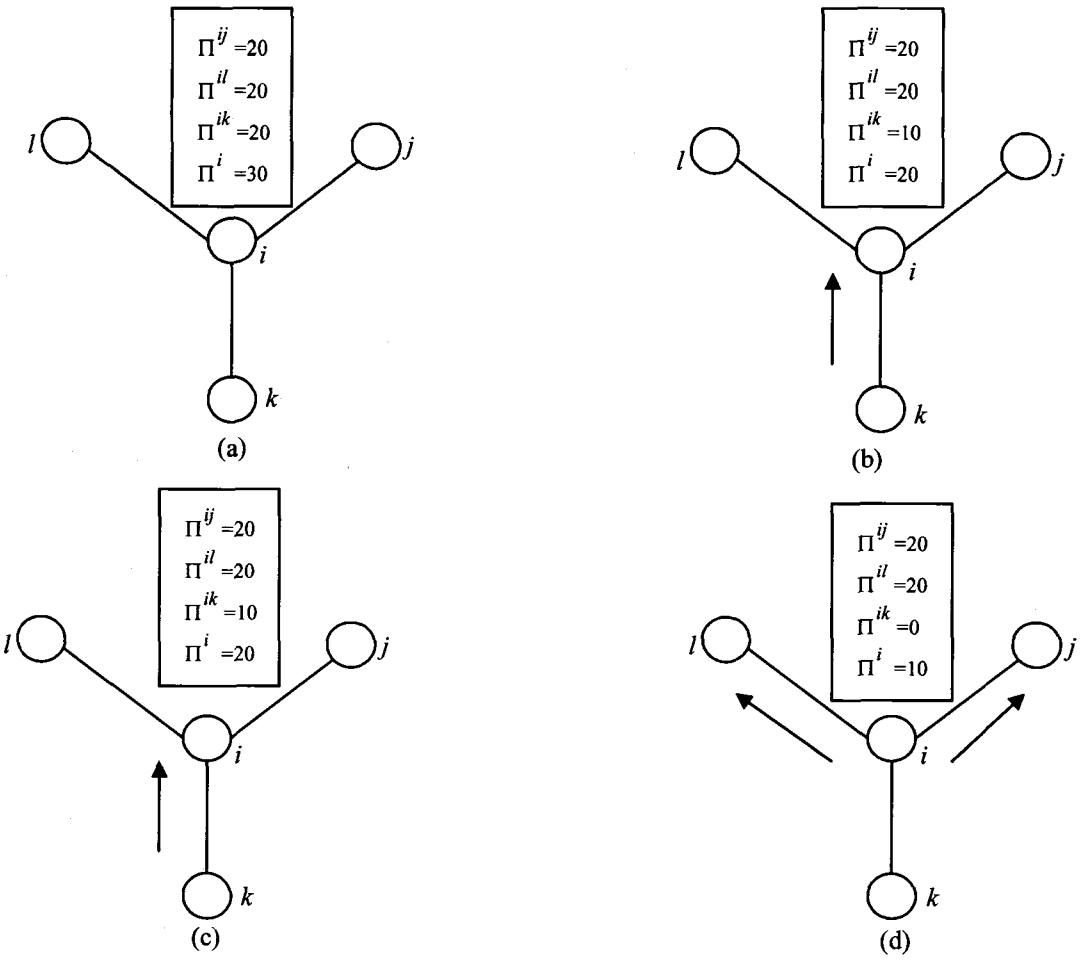


Fig. 5.3: A visual illustration of the Majority Selection process from the perspective of node *i*.

### 5.3.2 Majority Bayesian Network Learning Algorithm

The main objective of the majority selection process is to communicate new edges to neighboring nodes, but the identification of these edges within the database of each node has to be determined. The majority Bayesian network learning algorithm is responsible for discovering edges based on its local data and combining edges discovered by other nodes by using a structure-learning algorithm.

Structure learning is a model selection problem, wherein we need to select a model based on the data. We adopt a searching and scoring based method K2 [5] which defines a "confidence" that describes the fitness of each possible structure to the observed data.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The K2 algorithm is similar to an optimization problem: find a structure of Bayesian network that maximizes the confidence. The confidence function has a decomposability property defined as follows:

$$conf(BN, DB) = \sum_i conf(v_i, pa(v_i), DB(v_i, pa(v_i))) \quad (2)$$

where $pa(v_i)$ represents the parent of variable $v_i$, and $DB(v_i, pa(v_i))$ denotes the data involving only $v_i$ and $pa(v_i)$. Once the confidence level is defined and computed for the variables at each node, the next step is to identify the local Bayesian network with the highest confidence level. The search for the optimal Bayesian network is NP-hard, leading to the need for sub-optimal search methods. As the K2 algorithm is characterized by a decomposability property. The sub-optimal search methods involve making a series of edge changes (adding or deleting of edges one at a time). The confidence levels are checked at the end of every edge change. Every time an edge is changed, a check is performed to ascertain if the resultant local Bayesian network is a valid directed acyclic graph. For every edge change, a confidence level $conf^a(BN^a, DB)$ is calculated for $BN^a$ before the change and $conf^b(BN^b, DB)$ for $BN^b$ after the change. The decision regarding whether $BN^a$ or $BN^b$ is the best fit depends on which of the confidence levels has a maximum value. As the score satisfies the decomposability property, the confidence levels of edges are maintained by computing the following score for each edge: $conf(v_i, pa(v_i)_2, DB(v_i, pa(v_i)_2))$.

Once a message is received from the majority consensus protocol, the contents of the message indicate new edges or edges with updated confidence levels. If the message contains discovery of a new edge, the edge is added to the local Bayesian network. If the message merely indicates changes in confidence level, the confidence level of the corresponding existing edge is updated. The confidence level computation is performed. The structure of the local Bayesian network is modified based on the difference between the confidence levels before and after the change. The pseudocode for the algorithm is provided in pseudocode 2 below.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

**Pseudocode 2** Majority Bayesian Network Learning Algorithm

**// Inputs**

*Newscast*($E^i$)

//Execute Bayesian Network Structure Learning Algorithm to discover the local edge list *Edg* at node *i.*

*BNJ(Edg)*

**//Initialization**

*for every* $e \in Edg$

*Compute(e.conf, e.len)* // Set confidence thresholds and update number of records

*for every* $e \in Edg$

*for every* $i, j \in E^i$

$e.conf_{ij}$=$e.len_{ij,}$=$e.conf_{ji}$ =$e.len_{ji}$ =0

**// Output on Reception**

//Message received from node *j*. Node *j* sends 3-tuple message containing edge(*a*,*b*), the confidence, and length values.

*MessageRecv* ($e_i$, *conf, len,j)*

*if* $e_i \notin Edg$

*Add(Edg , $e_i$ ) // Add $e_i$ to edge set Edg)*

*ComputeConf // Compute confidence levels for BN before and after the addition*

*If conf(prevBN) < conf(newBN)*

$e_i.conf_{ji}$ = *conf*

$e_i.len_{ji}$ = *len*

*else*

*ComputeConf // Compute confidence levels for BN before and after the addition*

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

*If conf(prevBN) < conf(newBN)*

*Update*($e_i$, $\Pi^{ij}$, $\Pi^{i}$) // *For edge* $e_i$ *calculate values for* $\Pi^{ij}$ and $\Pi^{i}$

*While(true)* // Ensure that neighbors have updated confidence values

*for every* $e \in Edg$

*for every* $i, j \in E^i$

*if* $((e(\Pi^{ij}) < 0)$ and $(e(\Pi^{ij}) < e(\Pi^{i})))$ or $((e(\Pi^{ij} > 0)$ and $(e(\Pi^{ij}) > e(\Pi^{i})))$

$$e(conf_{ij}) = \sum_{l, i \neq ij \in E^i} e(conf_{li})$$

$$e(len_{ij}) = \sum_{l, i \neq i, j \in E^i} e(len_{li})$$

*MessageSend*$(e, e(conf_{ij}), e(len_{ij}), j)$

---

The visual representation for the majority Bayesian network algorithm is provided below. On identification of an edge at each node, the confidence level is computed. At the same time, the node will begin an instance of majority selection process to determine if the edge is globally correct in the network. As each node runs an instance of the majority selection process, every 3-field tuple message contains a unique edge identifier in addition to the confidence and record lengths.

Fig. 5.4 describes four different scenarios addressed by the majority Bayesian network algorithm. In Fig. 5.4a the first scenario is depicted wherein node *i* has edges *E*1, *E*2, and *E*3 along with the confidence and length values. On receipt of a new edge *E*4 from node *k*, *E*4 is added to node *i* and the values $E4.conf_{ki}$ and $E4.len_{ki}$ are computed as shown in Fig. 5.4b. In Fig. 5.4c the second scenario is shown wherein node *i* receives updated confidence values for edge *E*3 from node *k*. On receipt of the updated confidence values from node *k*, the $\Pi^{i}$ values for *E*3 are computed at node *i* as shown in Fig. 5.4d. As the conditions $\Pi^{i} > \Pi^{ij} > 0$ and $\Pi^{i} > \Pi^{il} > 0$ are satisfied, messages are sent to node *j* and *l* to update their confidence in edge *E*3.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

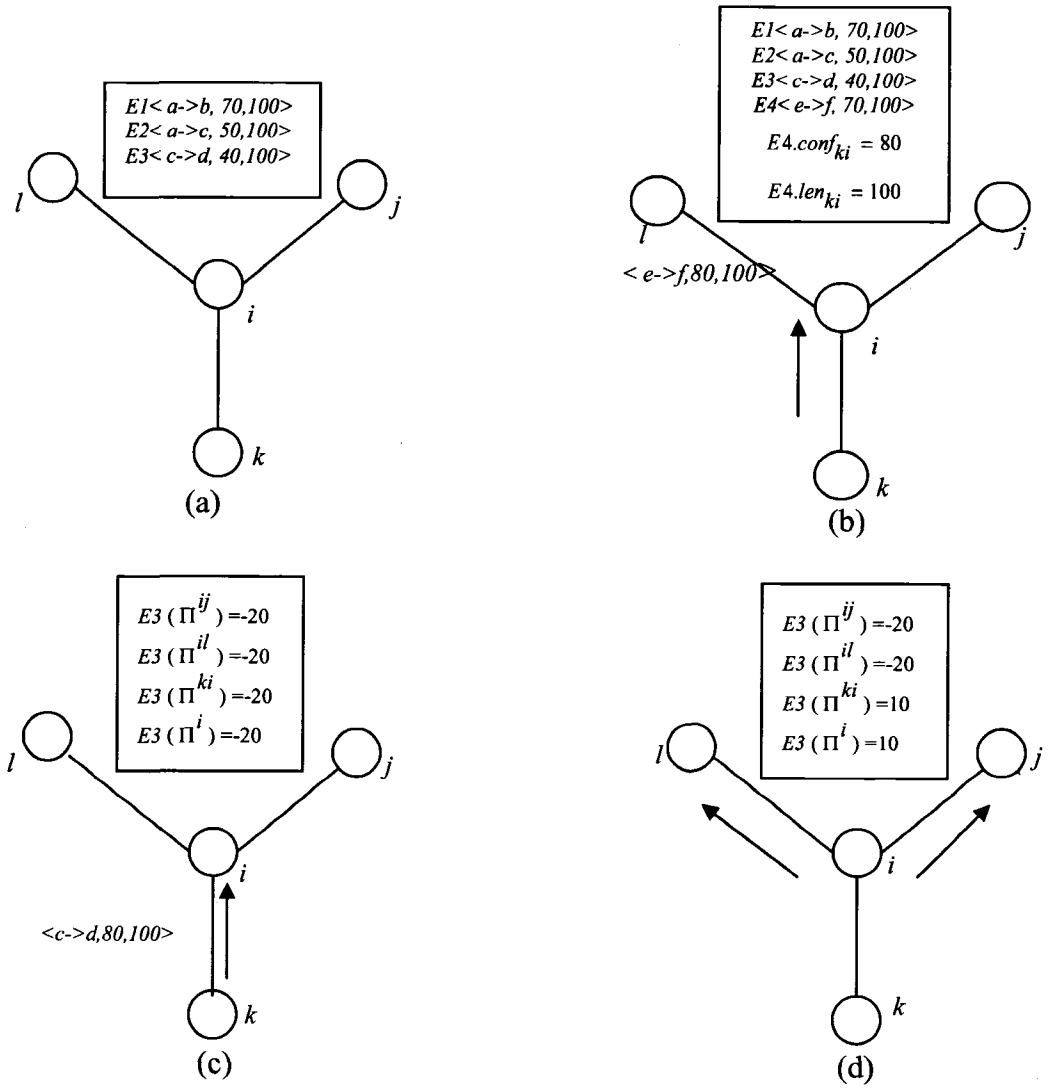


Fig. 5.4: A visual illustration of the majority Bayesian network learning algorithm from the perspective of node *i*.

## 5.4 Simulations and Analysis

To validate and verify the majority based consensus methodology, we implemented a simulator capable of running a network of 10,000 interconnected nodes. The nodes were connected in a random tree. For lack of real datasets, we tested our approach on the ASIA model dataset [13]. We generated 1,000,000 observations from this model, which were distributed among the peers in the network. In practice, however we do not have control over the distribution of the data among different nodes. Local Bayesian networks were

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

constructed using a *K*2 [5] structure-learning algorithm for learning the Bayesian network structure.

A software package based on BNJ [1] and Peersim [16] has been developed to implement our majority based Bayesian network learning algorithm. Peersim is an open source peer-to-peer systems simulator platform, using the Newscast protocol for the management of the overlay topology. The time is divided into rounds, and in each round each node implements the majority Bayesian network algorithm and the majority consensus protocol. The majority process and message construction, Bayesian network learning processes, execute on each node.

### 5.4.1 Effect of Majority Selection on Local Communication

One of the main performance characteristics of our protocol is estimation of the locality of the majority selection protocol. Locality is quantified by measuring the scope of a node. The scope of node *i* is defined as the number of neighbors whose confidence levels are maintained by *i*. The overall locality is measured by taking into account the maximum, average and minimum values of scope. Average locality is preferred for Majority Selection. In absence of average locality, the performance of the protocol degrades with increasing scalability. The scope also provides information related to message and processing overhead. Message overhead is computed based on the total number of messages exchanged between node *i* and its neighbors. Processing overhead is computed by calculating the number of cycles required at node *i* to arrive at a decision. Hence average locality also makes the communication fast and efficient.

Fig. 5.5 describes the average and minimum scope results of a simulation with 5,000 nodes connected in a random tree topology. The figure shows the locality of the protocol to determine the existence of one relational edge between two variables. The confidence levels for the relational edge are based on the amount of related data at each node. As the data is randomly distributed among the 5,000 nodes, the confidence levels for the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

relational edge are assigned randomly. Fig. 5.6 shows similar results for a network of 10,000 nodes.

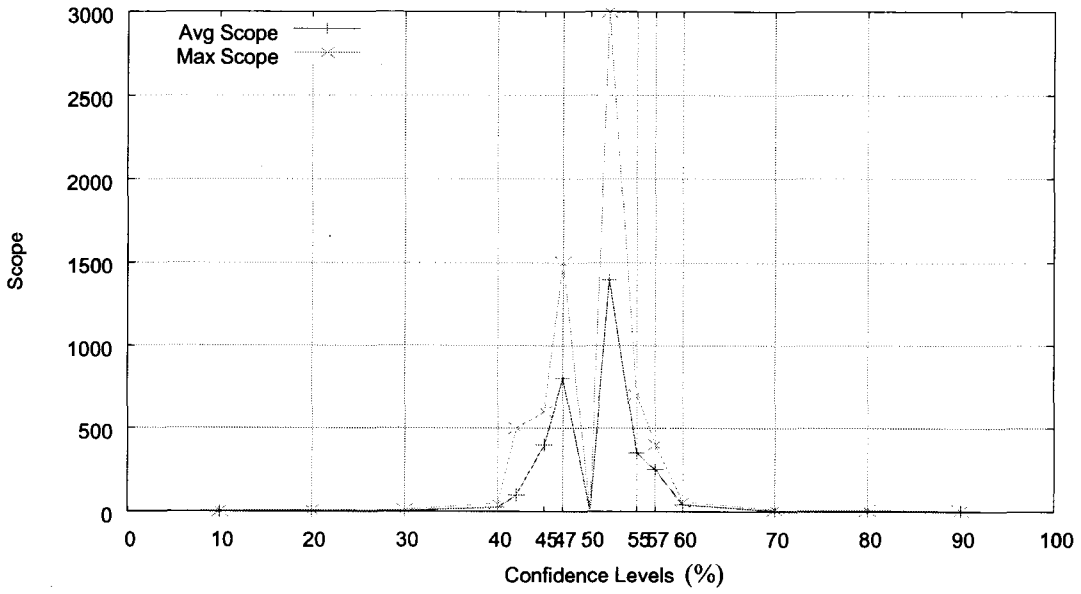


Fig. 5.5: Locality for network size of 5,000 nodes and majority value of 50%.

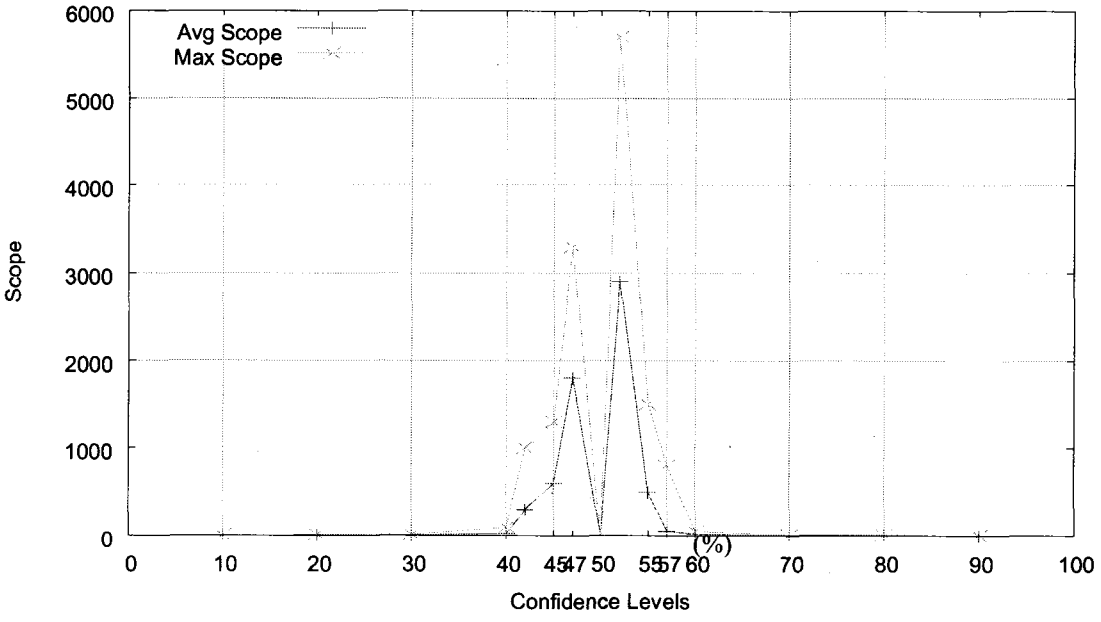


Fig. 5.6: Locality for network size of 10,000 nodes and majority value of 50%.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Figs. 5.5 and 5.6 show that if the data is distributed in an unbiased fashion, the locality of the protocol is assured. For a majority ratio of 50%, if the confidence levels are greater than 60% or less than 40%, the number of nodes involved in the communication of messages is limited to 10-12 nodes. As the confidence levels approach 50 %, the average scope grows. This indicates that more nodes have to be involved in the decision-making when the confidence levels approach the majority ratio.

From the above results in the above figures we can conclude that except for edges whose confidence levels are about equal to the confidence threshold, the rest of the edges are discovered using information gathered from a very small neighborhood, whose size is independent of the size of the network.

We assess the strength of our majority consensus protocol by measuring the message overhead. The message overhead calculation is based on the difference between the average confidence levels and the majority ratio. The total message overhead in our majority consensus protocol is equal to the messages exchanged between every node with its neighbors. The message exchanged is a 3-field tuple represented by $<Edg_{ab}, conf_{ij}, len_{ij}>$. If *Size* represents the length of each tuple in bytes, then the overall message overhead is given by $\sum_{i,j \in E^i} Size_{ij}$ only when the following conditions are met

$0 < ((conf_i - R)/(R)) < 0.2$ or $0 > ((conf_i - R)/(R)) > -0.2$.

The message overhead is approximately equal to zero when $((conf_k - R)/(R)) > 0.2$ or $((conf_k - R)/(R)) < -0.2$. It should be noted that the message overhead depends on the size of the neighbor list and the number of data bits exchanged between neighbors.

To prove that the message overhead incurred in majority consensus protocol is lower, we compare our protocol with a similar majority based protocol proposed by Ran Wolff *et.al.* [22]. According to Ran Wolff *et. al,* the message overhead in their majority based protocol is given by a constant function $1 + \sum_{i,j \in E^i} count^{ij}$, where $i$ and $j$ represent the nodes in the network, $count^{ij}$ represents the message length exchanged between the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

nodes $i$ and $j$, and $E^i$ represents the set of edges which collide with node $i$. If the majority based approach proposed by Ran Wolff *et al* is applied to discover distributed Bayesian networks where the majority of the confidence levels are not close to the majority ratio, then the message overhead is given by the constant function $1+\sum_{i,j\in E^i} count^{ij}$, while in the case of majority consensus protocol, the message overhead is approximately equal to zero. Thus, there is significant savings in message overhead by applying majority consensus protocol when the confidence levels are not close to the majority ratio.

### 5.4.2 Convergence of Majority Bayesian Network Learning Algorithm

In addition to locality, the other important characteristic of the majority consensus protocol is the convergence rate. We measure convergence by calculating the percentage of new edges discovered and the percentage of correct edges in the global Bayesian network. Fig. 5.7 shows the percentage of new edges discovered on an average basis during the execution of the majority consensus protocol. The convergence rate is measured by the total number of rounds completed before the discovery of the global Bayesian network. A round is defined as the amount of time taken by all nodes to execute one instance of both the protocol and algorithm. Fig. 5.7 also shows the percentage of correct edges discovered. In Fig. 5.7, the confidence levels of the edges in the majority of the nodes were away from the majority ratio. This leads to fewer messages exchanged and thereby most of the nodes agree on the majority decision quickly. All edges are discovered at the end of 12 rounds. Fewer rounds confirms the fact that if the confidence levels of the edges are far away from the majority ratio the nodes discover the new edges quickly. As the number of messages exchanged between the nodes decreases, the nodes can reach a decision quickly leading to fewer rounds.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

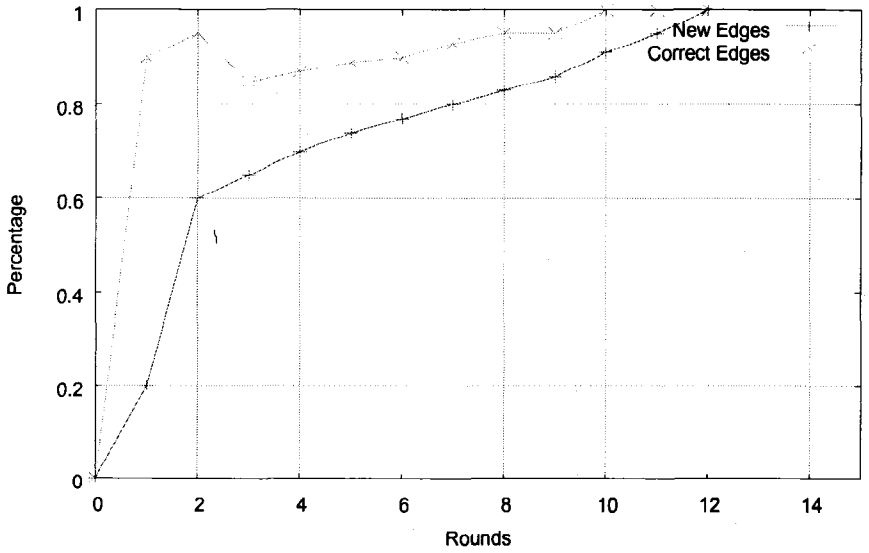


Fig. 5.7: Percentage of new edges and correct edges discovered for a network of 5,000 nodes on an ASIA dataset.

Fig. 5.8 shows the percentage of the nodes who arrive at a correct global Bayesian network model when the confidence level in the edges are less than or equal to 40%. For edges with confidence levels lesser than or equal to 30%, all the nodes quickly arrive at a decision to reject the edges. The number of rounds taken to arrive at this decision is less than or equal to 10. This result confirms the fact that all nodes converge quickly when the edges have confidence levels much lower than the 50% threshold

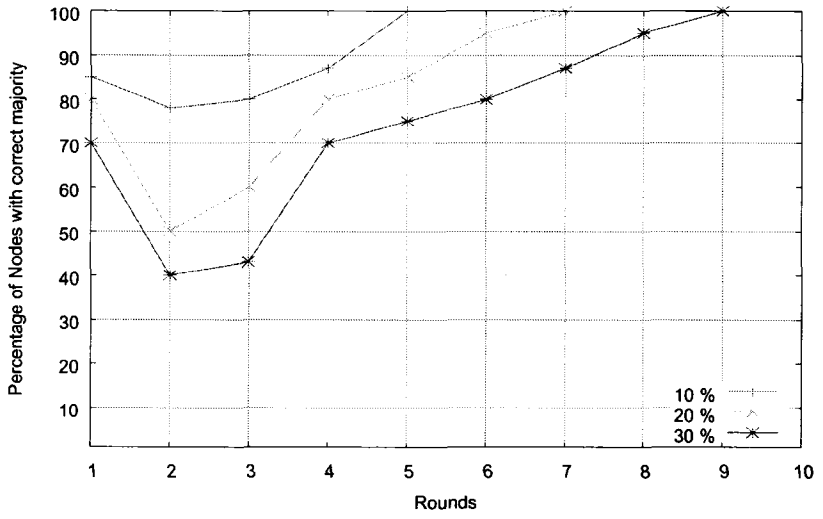


Fig. 5.8: Percentage of nodes which compute the right decision for confidence levels lesser than or equal to 30 % threshold.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Fig. 5.9 shows the percentage of the nodes that arrive at a correct global Bayesian network model when the confidence level in the edges is greater than or equal to 60%. For edges with confidence levels greater than or equal to 70%, all the nodes quickly arrive at a decision to accept the edges. The number of rounds taken to arrive at this decision is less than or equal to 10. This result confirms the fact that all nodes converge quickly when the edges have confidence levels much higher than the 50% threshold

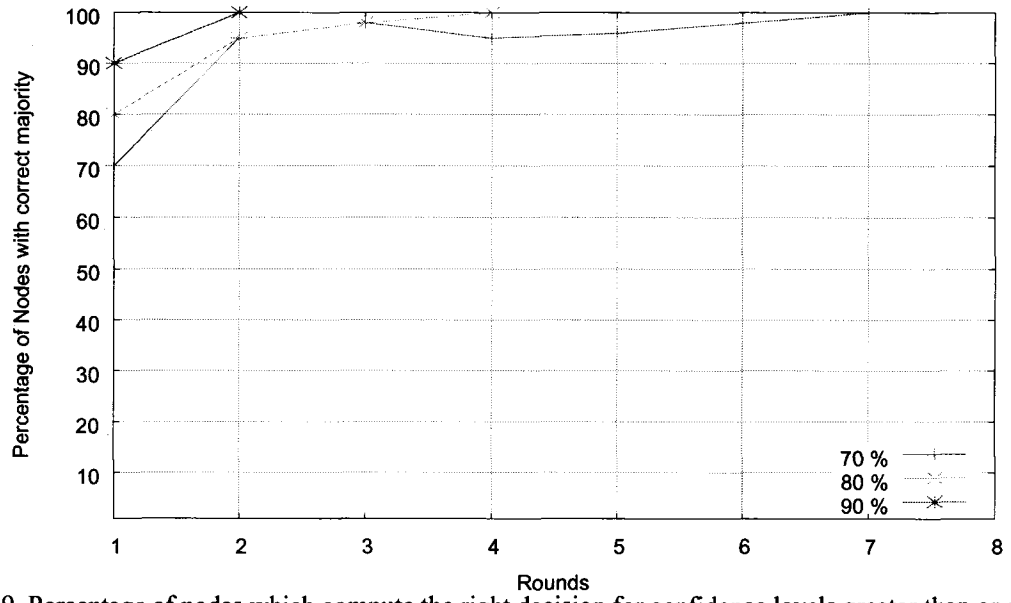


Fig. 5.9 Percentage of nodes which compute the right decision for confidence levels greater than or equal to 70% .

In Fig. 5.10, the confidence levels are closer to the threshold. The presence of a majority of nodes with confidence levels near the 50% threshold leads to more messages exchanged thereby delaying the agreement on decision making by all the nodes. Figure 5.10 indicates that edges with confidence levels near the 50% threshold require a large portion of the network to participate in order to arrive at a decision. Edges with confidence levels closer to 50% therefore take a significant amount of time for all nodes to agree.

Fig. 5.10 also shows that the number of rounds are considerably higher as compared to convergence in Figs 5.9 and Figs. 5.10. In spite of the larger convergence time in our

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

protocol for confidence levels closer to the majority ratio, the number of rounds are not dependent on the network size. We compare the convergence of our protocol with the convergence of a similar majority consensus protocol proposed by Burman *et.al.* [5]. According to Burman *et. al,* when a batch of transient faults hits an asynchronous distributed system by corrupting the state of some $f$ nodes, the state stabilizes in $O(f)$ time at all nodes, for any unknown $f$. State stabilization time is proportional to the network diameter, so the convergence of the protocol proposed by Burman *et.al.* increases with the network diameter, unlike our protocol whose convergence is independent of the scale of the network.

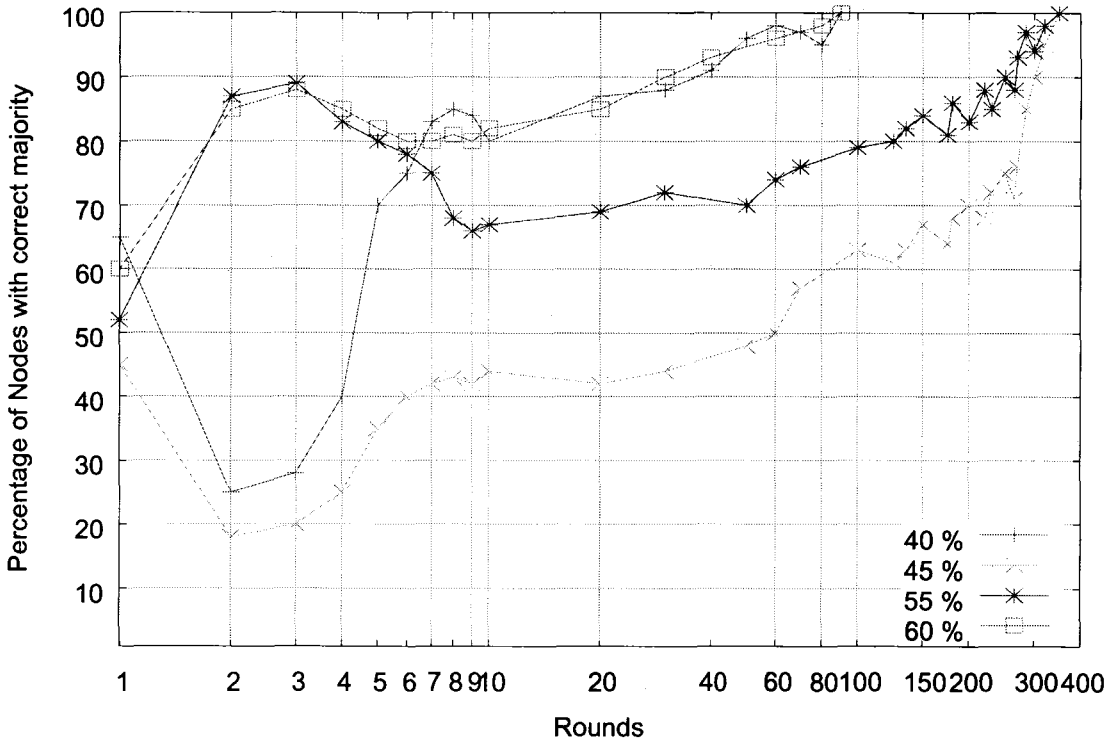


Fig. 5.10: Percentage of nodes which compute the right decision for confidence levels near the 50% threshold.

To better explain Figs. 5.7, 5.8, 5.9, and 5.10, the interaction between the majority consensus protocol and majority based Bayesian network algorithm needs to be analyzed. The majority consensus protocol behaves in a wave pattern characterized by positive

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

slopes and negative slopes. In the first round, every node assumes the confidence level to be zero. In the initial rounds, the protocol convinces more nodes of a certain confidence level leading to a wave with a positive slope, but when the majority of nodes do not agree on a certain confidence level, then the wave exhibits a negative slope. If the confidence level among the edges is between 70-90%, then the edges are learned quicker and agreed upon by most nodes. These edges are very significant and are expected to be discovered in the earlier rounds and quickly agreed upon by all the nodes in the network. The same analysis applies for edges with lower confidence levels (10-30%). The edges with low confidence levels are rejected earlier by most of the nodes. They are also discovered early but are quickly rejected as a greater portion of the local database is scanned. Now, the edges with confidence levels between (40-60%) require more rounds for all the nodes to agree with the majority decision. As these confidence levels are near the 50% majority threshold, the majority based Bayesian network algorithm takes a longer time for all the nodes to agree with the majority decision.

The above results also confirm the fact that the scalability of the methodology is dependent on the separation of the confidence levels from the majority ratio. To ensure that the performance of the methodology is independent of the size of the network, the majority of the confidence levels of the edges should be further away from the majority ratio.

## 5.5 Conclusion

We have devised a new methodology for discovering the structure of a Bayesian Network from data distributed in a peer-to-peer network. The crux of the methodology is the majority consensus protocol. The strength of the protocol is the locality of communication and quicker convergence if the confidence levels of the edges are not too close to the majority ratio. The locality of the communication allows a node running the majority BN algorithm to deduce a correct global BN model by interacting with fewer neighbors.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The efficiency of the local communication also depends on the confidence level of the edges. Simulation results verify that all new edges, except for edges with confidence levels closer to the majority ratio, are discovered by exchange of messages with a small neighborhood of nodes, whose size is independent of the size of the network. If the edges have a lower confidence level (10-30%), the communication is very efficient due to fewer exchanges of messages. Edges with a higher confidence level (70-90%) also result in fewer messages exchanged between neighboring nodes. As most of the edges have confidence levels which fall into the above two categories, the communication load mainly depends on the edges whose confidence level is between (40-60%). To control the communication load in our methodology, the confidence levels of the relations should be chosen based on the above constraints.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# Chapter VI

# CONCLUSIONS AND FUTURE DIRECTIONS

This dissertation has demonstrated that integration of distributed Bayesian Networks from data distributed in a peer-to-peer network can have faster convergence at a significantly lower communication overhead. The peer-to-peer model proposed in this dissertation is similar to the architectural model for distributed knowledge discovery. More specifically the peer-to-peer model can be implemented in the communication layer of the architectural model. The crux of the model is to perform the integration of the distributed Bayesian Network "in-network" without placing additional burden of integration on any computing node. This property of our model is a huge advancement over centralized integration of models proposed in previous approaches on distributed knowledge discovery.

Further, this dissertation shows that epidemic protocols are a good fit in the computation of distributed Bayesian Networks due to their robust and scalable manner of distributing information in peer-to-peer networks. Additional properties of epidemic protocols which make them very useful for our work are their ease of deployment and resilience to failures. This dissertation also presents a distributed algorithm by which every node in the peer-to-peer network can learn the exact distributed Bayesian network, as if it were given the combined database. The algorithm is based on majority consensus among all the nodes. It is entirely asynchronous, imposes very little communication overhead, transparently tolerates network topology changes and node failures,

This chapter lists the above contributions and expounds on several possible directions for future improvement.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## 6.1 Conclusions

We have presented a methodology to discover a distributed Bayesian network from data distributed among nodes in a peer-to-peer network. The crux of our methodology is to provide every node in a peer-to-peer network the ability to discover the exact global Bayesian network. The main contribution of our work is built around the three main components which are part of our methodology: Semantic Genetic Algorithm for structure learning of Bayesian Networks, Adaptive Fanout based Epidemic Protocol, and Majority Based Consensus Methodology.

The main contributions of our work are:

- Semantic genetic algorithm (SGA) to learn the best Bayesian network structure from a database. In SGA, we introduced semantic crossover and mutation operators to aid in obtaining accurate solutions. The crossover and mutation operators incorporate the semantic of Bayesian network structures to learn the structure with very minimal errors. SGA has been proven to discover Bayesian networks with greater accuracy than existing classical genetic algorithms

- A model for dynamic fanout based epidemic protocol to aid in fine-grained control of the information dissemination process. Our dynamic fanout model is best suited for distributed computing applications in which a global function within a pre-determined response time needs to be computed. One such application is distributed knowledge discovery where the global knowledge discovery model is constructed from local knowledge discovery models learned at each peer node. The accuracy and speed of the global knowledge discovery model construction process is dependent on the underlying controlled information dissemination process. In the presence of a dynamic data environment, the knowledge discovery process must proceed according to the availability of data in peer nodes. We present two approaches to quantify the fanout and thereby ensure all nodes receive a specific message within a bounded time.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

- The first approach is called *Round Based Dynamic Fanout*, in which each node transmits a message with a varied fanout from round to round. In this approach, the network topology is flat with no hierarchy. The fanout values are quantified based on the infection pattern and redundancy message pattern over rounds.

- In the second approach, *Cluster Based Dynamic Fanout*, we shift from a flat membership network to a cluster membership network. In the first approach, the fanout remains constant during a round. In the second approach, nodes are clustered based on geographical proximity criterion and we vary the fanout between clusters of nodes. This implies that during each round, nodes in different clusters disseminate information using different fanout values.

- Our final contribution is the use of a majority based consensus methodology to learn the structure of a Bayesian Network from data distributed in a peer-to-peer network. The methodology is comprised of the *Majority Consensus Protocol* and the *Majority Bayesian Network Learning Algorithm*. The algorithm and protocol work in tandem to discover the global BN at each node in the network.

- The key strength of our Majority Consensus Protocol is faster convergence with significantly lower message overhead. Also, the performance of our majority consensus protocol is not dependent on user configurable parameters. The parameter independence makes the protocol more predictable and robust. The majority consensus protocol is implemented locally and requires no synchronization among the nodes. The impact of the locality of communication leads to faster convergence of the computation of the global Bayesian network model at the expense of lower message overhead. By locality, we imply that each node updates its BN based on the link information provided by a small set of neighbors.

- The key strength of the Majority Bayesian Network Learning Algorithm is to correctly ascertain the confidence in the cross links discovered by the Majority

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Consensus Protocol. By using the algorithm the time taken by each node to discover the global BN does not exceed the time taken by a centralized BN learning approach.

## 6.2 Future Directions

In the future, we expect to eliminate a few of the assumptions we made in our methodology and use the methodology to learn different knowledge discovery models such as association rules, and classification. Eliminating some of the assumptions we made in our work would make our methodology more beneficial to model designers. Using the model to learn different knowledge discovery models would provide us with key insights into the model's flexibility.

The issues in the adaptive fanout model that need further investigation include: a) good choice of user specified infected nodes list, b) scalability of network in the cluster based dynamic fanout approach. For the round based dynamic fanout approach, the design of a good frequency distribution of infected nodes provided by the system user needs to be carefully examined. The cluster based dynamic fanout approach needs to be implemented on a large cluster size. We aim to address these issues in our future work.

One of the critical assumptions of the majority based consensus methodology was related to the construction of the database. The database was distributed homogeneously among the nodes in the peer-to-peer network. In a homogenous database all nodes have all of the variables whose edges have to be discovered. One of the directions for future work would be to learn the structure of a Bayesian network from a heterogeneous database. In a heterogeneous database, every node has a subset of variables and their observations. This restriction leads to the exchange of observations between nodes. To achieve this, the Majority Bayesian Network algorithm will have to be modified accordingly. The majority consensus protocol would need no changes, as it is independent of the type of the database.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Another assumption is related to dynamic changes in data. The simulations performed to validate our majority based consensus methodology were based on static data. Relaxing this assumption would give us more insight into the operation of the majority consensus protocol. The convergence of our methodology will be affected in the presence of dynamic data. The majority consensus protocol will have to be modified to alleviate the delay in convergence due to addition or deletion of data from the local databases.

Future work might also use our methodology to learn the parameters of a Bayesian network along with structure and other knowledge discovery models like association rules, decision trees, and clustering. Our methodology has been constructed in such a fashion that to discover any other type of knowledge discovery model would only entail changes in the majority Bayesian network algorithm. The different components of models (for example, rules and transactions discovery for association rule mining) and different confidence level interpretation (for examples, frequency and support for association rules mining) should be taken into consideration. Our methodology can gain widespread acceptability if it is successfully applied in order to discover different types of knowledge discovery models.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# References


[1] Bailey, N. 1975. *The Mathematical Theory of Infectious Diseases and its applications*, Hanfer Press.

[2] Bayesian Network Tools in Java (BNJ) http://bnj.sourceforge.net/

[3] Birman, K.P., Hayden, M., Ozkasap, O., Xiao, Z., Budiu, M., and Minsky, Y.,. Bimodal multicast. *ACM Transactions on Computer Systems*, 17(2):41–88, May 1999.

[4] Blanco, R. Inza, I., and Larrañaga, P. (2003). Learning Bayesian Networks in the Space of Structures by Estimation of Distribution Algorithms. *International Journal of Intelligent Systems*, *18*(2), 205-220.

[5] Burman, J., Kutten, S., Herman, T., Patt-Shamir, B., "Asynchronous and Fully Self-Stabilizing Time-Adaptive Majority Consensus", 9th International Conference on Principles of Distributed Systems, 2005.

[6] Chen, R., Sivakumar, K., and Kargupta, H., "Collective Mining of Bayesian Networks from Distributed Heterogeneous Data," *Knowledge and Information Systems Journal*, vol. 6, pp. 164-187, 2004

[7] Cheng, J., Bell, D.A. and Liu, W., An algorithm for Bayesian belief network construction from data, Proceedings of AI & STAT'97 (pp.83-90), Ft. Lauderdale, Florida, 1997.

[8] Cooper, G., and Herskovits, E. A., "A Bayesian Method for the Induction of Probabilistic Networks from Data," Machine Learning, vol. 9, no 4, 1992, pp. 309-347.

[9] Cotta, C. and Muruzabal, J., "Towards more efficient evolutionary induction of Bayesian networks," In M. Guervos et al., editors, Lecture Notes in Computer Science, Volume 2439, 2002, pp. 730-739.

[10] Daley, D., and Gani, J. 1999. *Epidemic Modelling. An Introduction*, Cambridge Express.

[11] Demers, A. J., Greene, D. H., Hauser, C., Irish, W., and Larson, J., Epidemic algorithms for replicated database maintenance. In *Proceedings of the Sixth*


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

*Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 1–12, Vancouver, British Columbia, Canada, August 1987.

[12] Dijk, S. V., Thierens, D., and Gaag, L. C. (2003). Building a GA from Design Principles for Learning Bayesian Networks. *Proceedings of the Genetic and Evolutionary Computation Conference*, 886-897.

[13] Eugster, P., Guerraoui, R., Handurukande, S., Kermarrec, A., and Kouznetsov, P. 2003. Lightweight probabilistic broadcast. *ACM Transaction on Computer Systems*, 12, 341-374.

[14] Gupta, I., Birman, K., and Van Renesse, R. 2002. Fighting fire with fire: using randomized gossip to combat stochastic scalability limits. *Quality and Reliability Engineering International*, 18, 165–184.

[15] Gupta, I., Van Renesse, R., and Birman, K. 2001. Scalable fault-tolerant aggregation in large process groups. *Dependable Systems and Networks*, Jul 2001, 433-442.

[16] Fayyad, M., Shapiro, G., Smyth, P., and Uthurusamy R., "Advances in Knowledge Discovery and Data Mining", 1999, AAAI Press

[17] Gupta, I., Van Renesse, R., and Birman, K., Scalable fault-tolerant aggregation in large process groups, 2001.

[18] Heckerman, D. A., "A tutorial on learning with bayesian networks," TechnicalReport 9506, Microsoft Research, 1996.

[19] Heckerman, D., Geiger, D., and Chickering, D. M. (1995). Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, *20*(3), 197-243.

[20] Hedetniemi, S.M., Hedetniemi, S.T., and Liestman, A., A survey of gossiping and broadcasting in communication networks. *Networks*, 1986.

[21] Henrion, M. (1988). Propagating Uncertainty in Bayesian Networks by Probabilistic Logic Sampling. *Uncertainty in Artificial Intelligence* 2, 149-163.

[22] Jelasity, M., Montresor, A., and Babaoglu, O. 2004. Detection and removal of malicious peers in gossip-based protocols. In FuDiCo II:S.O.S. Bertinoro, Italy. http://www.cs.utexas.edu/users/lorenzo/sos.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[23] Jelasity, M., Montresor, A., and Babaoglu, O. 2005. Gossip-based Aggregation in Large Dynamic Networks. *ACM Transactions on Computer Systems*, 23, 219-252.

[24] Jelasity, M., Kowalczyk, W., and Van Steen, M. 2003. Newscast computing. Technical Report IR-CS-006, Department of Computer Science, Amsterdam, The Netherlands.

[25] Jiawei H., and Micheleine K. "Data Mining: Concepts and Techniques," The Morgan Kaufmann Pub., 2000.

[26] Jensen, F. V. (1996). *Introduction to Bayesian Networks. Springer-Verlag*, New York, Inc., Secaucus, NJ.

[27] Johnson E. and Kargupta H., "Collective, Hierarchical Clustering from Distributed, Heterogeneous Data," In Large-Scale Parallel KDD Systems, Lecture Notes in Computer Science, Springer-Verlag. Zaki, M. and Ho, C. (Ed). 1999.

[28] Kargupta, H., Huang, W., and Johnson, E., "Distributed clustering using collective principal components analysis," Knowledge and Information Systems Journal, 2001.

[29] Karp, R.M., Schindelhauer, C., Shenker, S., and Vocking, B., Randomized rumor spreading. In *IEEE Symposium on Foundations of Computer Science*, pages 565–574, 2000.

[30] Kempe, D., Kleinberg, J., and Demers, A. 2001. Spatial gossip and resource location protocols. In *ACM Symposium on Theory of Computing*, Cete, Greece, Jul 2001, 163-172.

[31] Kenji, Y. (1997), Distributed cooperative Bayesian learning strategies, in `Proceedings of the Tenth Annual Conference on Computational Learning Theory', ACM Press, Nashville, Tennessee, pp. 250-262.

[32] Kermarrec, L. Massouli´e, and Ganesh, A. 2006. Efficient and adaptive epidemic-style protocols for reliable and scalable multicast, *IEEE Transactions on Parallel and Distributed Systems* 14, 593-605.

[33] Krivitski, D., Schuster, A., and Wolff, R., A Local Facility Location Algorithm for Large-Scale Distributed Systems. Journal of Grid Computing, 2006

[34] Kutten., S. and Patt-Shamir, B., Stabilizing time-adaptive protocols. *TheoreticalComputer Science*, 220(1):93.111, 1999.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[35] Kutten, S., and Peleg, D.,Fault-local distributed mending. In *Proceedings of the Fourteenth Annual ACM Symposium on Principle of Distributed Computing (PODC 95)*, pages 20.27, Ottawa, Canada, August 1995

[36] Lam, W. and Bacchus, F., "Learning Bayesian belief networks: An approach based on the MDL principle", Computational Intelligence, pp. 262-293,1994.

[37] Lauritzen, S. L. and Spiegelhalter, D. J. (1988), `Local computations with probabilities on graphical structures and their application to expert systems (with discussion)', Journal of the Royal Statistical Society, series B 50, 157-224.

[38] Lauritzen, S. L., "The EM algorithm for graphical association models with missing data", Computational Statistics and Data Analysis, pp. 191-201, 1995.

[39] Lam, W. and Bacchus, F. (1994). Learning Bayesian Belief Networks – An Approach Based on the MDL Principle. *Computational Intelligence*, *10*(4), 269-293.

[40] Larrañaga, P., Kuijpers, C. M. H., Murga, R. H., and Yurramendi, Y. (1996). Learning Bayesian Network Structures by Searching for Best Ordering with Genetic Algorithm. *IEEE Transactions on Systems, Man, and Cybernetics*, *26*(4), 487-493.

[41] Larrañaga, P., Poza, M., Yurramendi, Y., Murga, R. H., and Kuijpers, C. M. H. (1996). Structure Learning of Bayesian Networks by Genetic Algorithms: a Performance Analysis of Control Parameters. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, *18*(9), 912-926.

[42] Larranaga, P., et al, "Structure learning of Bayesian networks by genetical algorithms: a performance analysis of control parameters," in IEEE Trans. On Pattern Analysis and Machine Intelligence, vol 18, no 9, 1996.

[43] Lauritzen, S. L., and Spiegelhalter, D. J. (1988). Local Computations with Probabilities on Graphical Structures and Their Application on Expert Systems. *J. Royal Stat. Soc. B*, *50*(2), 157-224.

[44] Luo, J., Eugster, P., and Hubaux, J. 2003. Route driven gossip: Probabilistic reliable multicast in ad hoc networks. *Infocom,* April 2003, 2229-2239.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[45] Madigan, D. and Raftery, A., "Model selection and accounting for model uncertainty in graphical models using Occam's window", Journal of the American Statistical Association, pp. 1535-1546, 1994.

[46] Meila, M. and Jordan, M. I., "Estimating dependency structure as a hidden variable", in NIPS, 1998.

[47] Myers, J., and Levitt, T. (1999). Learning Bayesian Networks from Incomplete Data with Stochastic Search Algorithms. *Proceeding of the Fifteenth Conference on Uncertainty in Artificial Intelligence.* 476-485. Morgan Kaufmann Publishers.

[48] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufman, San Mateo.

[49] Peersim: http://peersim.sourceforge.net

[50] Singh, M., "Learning Bayesian networks from incomplete data", in Proceedings of the National Conference on Artificial Intelligence, pp. 27-31, 1997.

[51] Spiegelhalter, D. J. and Lauritzen, S. L., "Sequential updating of conditional probabilities on directed graphical structures", Networks, pp.570-605, 1990.

[52] Prodromidis, A., Chan, P., and Stolfo, S., "Meta-learning in distributed knowledge discovery systems: Issues and approaches," In Advances in Distributed and Parallel Knowledge Discovery. AAAI/MIT Press, 2000.

[53] Suzuki, J., "A construction of Bayesian networks from databases based on an MDL scheme", Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence, Morgam Kaufmann, pp. 266-273, 1993.

[54] Tanese, R. (1989) Distributed Genetic Algorithms. Proceedings of the Third International Conference on Genetic Algorithms, 434–439.

[55] Thiesson, B. , "Accelerated quantification of Bayesian networks with incomplete data," in Proceedings of the First International Conference on Knowledge Discovery and Data Mining, AAAI Press, pp. 306-311, 1995.

[56] Thomas, A., Spiegelhalter, D. and Gilks, W., "Bugs: A program to perform Bayesian inference using Gibbs sampling," in Bayesian Statistics', Oxford University Press, pp. 837-842,1992.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

[57] Van Renesse, R., Birman, K. P., and Vogels, W.,. Astrolabe: A robust and scalable technology for distributed systems monitoring, management, and data mining. *ACM Transactions on Computer Systems*, 21(3), 2003.

[58] Van Renesse, R., Minsky,Y., and Hayden, M., A gossip-style failure detection service. In *IFIP Conference on Distributed Systems Platforms an Open Distributed Processing*, 1998.

[59] Wong, M. L., Lam, W., and Leung, K. S. (1999). Using Evolutionary Programming and Minimum Description Length Principle for Data Mining of Bayesian Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 21*(2), 174-178.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# VITA

Sachin Shetty holds a Bachelor of Science degree in Computer Engineering from Mumbai University (1998) and a Master of Computer Science from the University of Toledo (2002). His dissertation focuses on designing distributed knowledge discovery algorithms for large-scale peer-to-peer networks. His work has been published in 10 conferences and refereed journal papers. He is the instructor for ECE 355 and ECE 455/555. He won first place in the graduate student competition in Old Dominion University's Research Expo 2006. His research interests are wireless sensor networks, learning distributed Bayesian Networks in peer-to-peer networks, multi-agent systems, and self-organization of agents in peer-to-peer networks.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.