Computational Intelligent Systems:
Evolving Dynamic Bayesian Networks

Thesis by

Isaac Olusegun Osunmakinde

Submitted In Fulfilment of the Requirements
For the Degree of

Doctor of Philosophy

In Computer Science

Of the

University of Cape Town, South Africa

Advisor

Dr Antoine Bagula

December 2009

Computational Intelligent Systems:
Evolving Dynamic Bayesian Networks

by

Isaac Olusegun Osunmakinde

DECLARATION

I declare that this thesis is my own original work. Where collaborations with other researchers are involved, or materials generated by other researchers are included, the parties and/or materials are acknowledged or are explicitly referenced as appropriate.

This work is being submitted for the degree of Doctor of Philosophy in Computer Science of the University of Cape Town, South Africa. It has not been submitted to any other university or institution for any other degree or examination.

Signed by candidate

Signature Removed

Isaac Olusegun Osunmakinde

December 2009

Date

An Abstract of

Computational Intelligent Systems:
Evolving Dynamic Bayesian Networks

Thesis by Isaac Olusegun Osunmakinde

Doctor of Philosophy in Computer Science
University of Cape Town, South Africa

December 2009

Dynamic Bayesian Networks (DBNs) are temporal probabilistic models for reasoning over time. They often formulate the core reasoning component of intelligent systems in the field of machine learning. Recent studies have focused on the development of some DBNs such as Hidden Markov Models (HMMs) and their variants, which are explicitly represented by highly skilled users and have gained popularity in speech recognition. These varieties of HMMs represented as DBNs have contributed to the baseline of temporal modelling. However they are limited in their expressive power as they are approximated and pose difficult challenges for users when choosing the appropriate model for diverse real-life applications. To worsen the situation further, researchers and practitioners have also stressed that applications often have difficulties when evolving (or learning) such network models from environments captured as massive datasets, due to the ongoing predominance of computational intensity (or nondeterministic polynomial (NP) time hard). Finding solutions to these challenges is a difficult task.

In this thesis, a new class of temporal probabilistic modelling, called evolving dynamic Bayesian networks (EDBN), is proposed and demonstrated to make technology easier so as to accommodate both experts and non-experts, such as industrial practitioners, decision-makers, researchers, etc. Dynamic Bayesian Networks (DBNs) are ideally suited to achieve *situation awareness*, in which elements in the environment must be perceived within a volume of time and space, their meaning understood, and their status predicted in the near future. The use of Dynamic Bayesian Networks in achieving situation awareness has been poorly explored in current research efforts. This research completely evolves DBNs automatically from any environment captured as multivariate time series (MTS) which minimizes the approximations and mitigates the challenges of choice of models. This potentially accommodates both highly skilled users and non-expert practitioners, and attracts diverse real-world application areas for DBNs. The architecture of our EDBN uses a combined strategy as it resolves two orthogonal issues to

address the challenging problems: (1) evolving DBNs in the absence of domain experts and (2) mitigating computational intensity (or NP-hard) problems with economic scalability.

Most notably, the major contributions of this thesis are as follows: the development of a new class of temporal probabilistic modeling (EDBN), whose architecture facilitates the demonstration of its emergent situation awareness (ESA) and emergent future situation awareness (EFSA) technologies. The ESA and its variant reveal hidden patterns over current and future time steps respectively. Among other contributions are the development and integration of an economic scalable framework called dynamic memory management in adaptive learning (DMMAL) into the architecture of the EDBN to emerge such network models from environments captured as massive datasets; the design of configurable agent actuators; adaptive operators; representative partitioning algorithms which facilitate the scalability framework; formal development and optimization of genetic algorithm (GA) to emerge optimal Bayesian networks from datasets, with emphasis on backtracking avoidance; and diverse applications of EDBN technologies such as business intelligence, revealing trends of insulin dose to medical patients, water quality management, project profitability analysis, sensor networks, etc. To ensure the universality and reproducibility of our architecture, we methodically conducted experiments using varied real-life datasets and publicly available machine learning datasets mostly from the University of California Irvine (UCI) repository.

DEDICATION

To my families

at all levels

for supporting me in pursuing my dreams

far away from home

ACKNOWLEDGEMENTS

To God be the glory for His mercies endureth forever and for keeping me alive until this day. I must say that this is the most difficult aspect of this thesis because so many people assisted me, but I must mention a few.

I would like to thank my advisor, Dr Antoine Bagula of the Computer Science Department, University of Cape Town, for his insightful encouragement and constructive ideas for the successful completion of this PhD programme. I do appreciate the support of testing the idea in my thesis successfully in the industry by my former advisor, Dr Anet Potgieter, with whom I started my PhD programme. I am grateful to the staff members of the Computer Science Department and the authorities at the University of Cape Town for the opportunities and financial support given to me.

Special thanks to Dr Eric Hermanson of the Mowbray Baptist Church (MBC), for his indefatigable efforts and contributions to the proof reading of this manuscript. I would like to thank Pastor Geoff Milligan and all the members of the MBC with whom we have had the pleasure of fellowshipping during our stay in Cape Town, South Africa.

Finally, I would like to thank my wife and my daughter, Mrs. Cecilia Oluseyi Osunmakinde and Grace Boluwatife Osunmakinde for their motivation and support in completing this programme. I am grateful to the members of the IEEE society and the staff members of the MDS.MIAS Department of the council for scientific and industrial research (CSIR) for the opportunities given to me. My appreciation also goes to our families such as the Osunmakindes and the Adebos, and our friends around the world for their encouragement, prayers, and contributions to the success of my PhD programme.

I say thank you and God bless you all.

Isaac Olusegun Osunmakinde, December 2009.

Table of Contents

List of Abbreviations

AI	Artificial Intelligence
BN	Bayesian Networks
CPTs	Conditional Probability Tables
DAG	Directed Acyclic Graph
DBN	Dynamic Bayesian Network
DMMAL	Dynamic Memory Management in Adaptive Learning
EDA	Extended Dependency Analysis
EDBN	Evolving Dynamic Bayesian Network
EFSA	Emergent Future Situation Awareness
ESA	Emergent Situation Awareness
ESN	Equipment Serial Number
EM	Expected Maximization
GA	Genetic Algorithm
HGA	Hybrid Genetic Algorithm
HMM	Hidden Markov Model
HHMM	Hierarchical Hidden Markov Model
IT	Information Technology
LRM	Logistic Regression Model
MCMC	Markov Chain Monte Carlo
MDL	Minimum Description Length
MI	Mutual Information
MIN	Mobile Identification Number
MLE	Maximum Likelihood Estimate
MTS	Multivariate Time Series
PDBN	Partial Dynamic Bayesian Network
PIN	Personal Identification Number
RDP	Representative Data Partitioning
SA	Situation Awareness
SBITA	Strategic Business and Information Technology Alignment
TADS	Telecommunications Anomaly Detection System

TOCO	Traversal Operators in Correct Order
TSP	Telecommunications Service Provider
TQM	Total Quality Management
2D	Two-Dimension

List of Figures

List of Tables

Chapter One

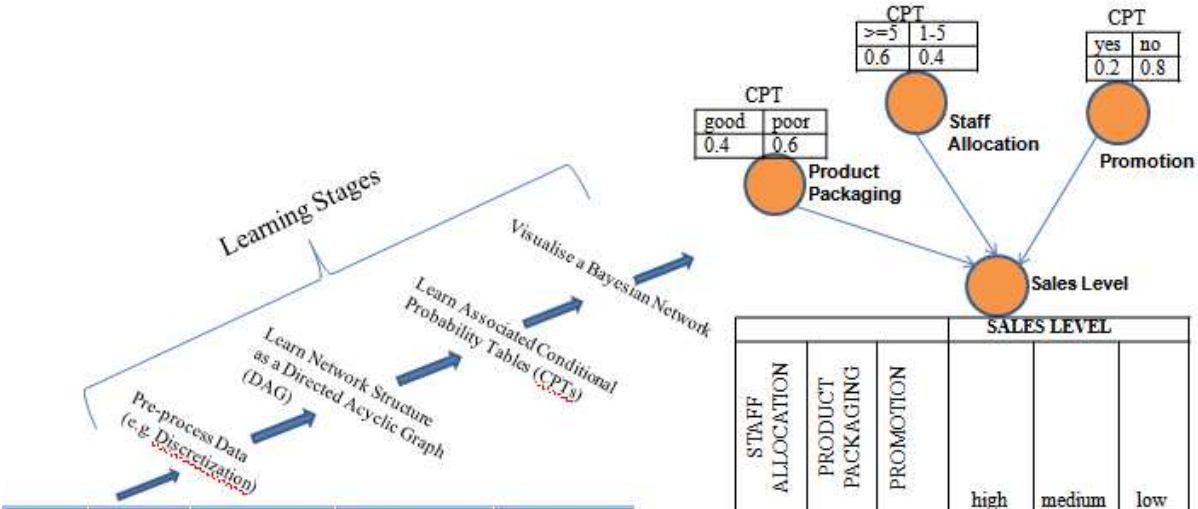
Introduction

1.1 Need for a New Class of Dynamic Bayesian Network (DBN) Modelling

Detailed understanding about how some conditions affect any environment of interest requires intelligence, such as assessing what drives a sales level of products at every point in time. Product promotion, for example, is a common factor that is known to drive the sales level, and packaging is the aesthetic element that can draw the attention of customers to the products. Allocation of experienced staff contributes partly to the level of sales. The interaction or contributions of all these sales attributes are referred to as causal models (or cause-effect relationships), which may be complex or simple depending on the nature of the environment. Unlike other decision-making models such as neural networks, decision trees, etc., the Bayesian Network (BN) is the only model in literature that can best be used to learn, discover and reason about the causes and effects of situations embedded in the environment, captured as datasets. Figure 1.1 describes a BN and provides a quick illustration for the processes involved in learning BNs from datasets. These processes are decomposed as follows into the problem of (i) data pre-processing, such as discretization of numeric data into their corresponding interval values, (ii) learning network structure as a directed acyclic graph, (iii) learning their associated conditional probability tables (CPTs), and (iv) visualization of the model. The inclusion of time when modelling BNs from datasets leads to the development of dynamic Bayesian network (DBN) models. The DBNs are temporal probabilistic models for reasoning under conditions of uncertainty over time, and are rapidly gaining popularity in the field of modern artificial intelligence (AI) for carrying out anticipatory planning. DBNs have been applied as reasoning components in intelligent system areas such as speaker recognition [1], Brain-Computer interfaces [2] and traffic modelling [3], and have gained popularity in speech recognition [7] [8].

Industrial practitioners and researchers observe multivariate time series (MTS) from the daily changing of variables in business activities or dynamical systems (e.g. operations, customers, products, medical systems, retail enterprises, sensor networks, etc.). Complex hidden relationships (or behavioural patterns) are often embedded amongst the variables that describe such activities, within and across the time steps. In order to comprehend temporal patterns from perceived ever-changing complex environmental information over time, sophisticated technologies are required for making effective and

dynamic decisions that ultimately lead to taking correct actions concerning events within a time constraint. Often, incorrect decisions are made due to the information gap between complex models of the environment and a multitude of possible situations [16]. The information gap leads to misalignment problems that affect decision-makers in business [17] [18]. This implies that users are unable to understand the related actions or interpret these models and their temporal implications in time to make the correct decisions. Finding a solution to this information gap problem is a challenging task.



INDEX	SALES LEVEL	PROMOTION	STAFF ALLOCATION	PRODUCT PACKAGING
1.	low	no	4	poor
2.	low	yes	2	good
3.	medium	yes	5	poor
4.	high	no	8	poor
5.	low	yes	1	good
6.	high	no	3	good
.	.	.	.	.
.	.	.	.	.
n	.	.	.	.

STAFF ALLOCATION	PRODUCT PACKAGING	PROMOTION	SALES LEVEL: high	medium	low
>=5	good	yes	0.18	0.47	0.35
		no	0.25	0.36	0.39
	poor	yes	0.38	0.27	0.35
		no	0.22	0.34	0.44
1-5	good	yes	0.61	0.07	0.38
		no	0.52	0.19	0.29
	poor	yes	0.44	0.33	0.23
		no	0.71	0.12	0.17

CPT

Figure 1.1: A Sample BN Model Learned from Business Environment to Guide in Unforeseen Situations about the Level of Sales of Products.

DBNs are often referred to as the extension of ordinary (or static) Bayesian networks (BNs) by intelligent systems engineers [3] [8]. Since a BN is static and encodes probabilistic knowledge of the environment as a graphical structure and conditional probability tables (CPTs), DBNs are expected to encode environmental knowledge as network structures and CPTs that change over time. The network structure and the CPTs are qualitative and quantitative knowledge respectively, which are captured (or discovered) from the environments, such as the BN learning shown in Figure 1.1. This shows the

discovery of the BN models from datasets through the learning stages. A number of Hidden Markov Model (HMM) representations of dynamic Bayesian networks with different characteristics have been developed to model these ever-changing environments. Murphy [3] presented some variants of HMM as explicit representations of DBN, such as: hierarchical HMM, coupled HMM and factorial HMM. Shenoy et al. [2] presented another DBN model for Brain-Computer Interfaces. It is a model of dynamics of hidden brain and body states. In [7], a type of structural learning of DBN is presented and applied to speech recognition. The varieties of the DBNs are reviewed in detail in Chapter Two.

Most of these DBNs have contributed to the baseline of temporal modelling but they are explicitly modelled under a number of assumptions by experts. For instance, they observed stationary assumption which made their DBN result in a repeated network structure at each time step. In reality, situations in some time steps may change. It is evident that most of these existing DBNs approximate their models, which potentially contributes to information gap problems. That is, they do not truly (completely) evolve the network structures and probability distributions; however the basis of DBN requires that they model both. Murphy [3] confirms that the HMMs, which are the basis of most of these representations of the existing DBNs, are limited in their expressive power. However, the varieties of DBNs available have opened up challenging problems of how to choose the most suitable model for various real-life applications. It is also asserted that some prediction models also suffer from convergence or exponential problems over wider time steps [8]. That is, the prediction steps get stuck towards zero or tend towards infinity. To complicate the situation further, the challenges have made technologies such as the DBNs too complex for non-experts, including practitioners [18] [19] (managers, software programmers, etc.). The capabilities of the DBNs have therefore not been fully exploited and this has hindered their wider applicability, especially in the absence of domain experts.

Since DBNs are described as extensions of ordinary BN models [3] [8], this implies that learning in BNs is also adapted into DBN learning. Researchers and practitioners have, however, stressed that learning (or emerging) such models from massive datasets is computationally intensive [20] [21]. Suppose that the table of records in Figure 1.1 is captured from the following real-life problem setting: *In developing countries, consider a small-scale retail enterprise which captures over five thousand MTS (multivariate time series) observations per hour per branch. On a promotion basis, the company would like to declare discounts on selected slow-moving products in specific low-sales branches. For business understanding, sales analysts require a safe completion of learning Bayesian networks from such massive datasets obtained and do not want to learn the models for weeks.* Thus, the learning stages in Figure 1.1 will be very intensive before a model can emerge. In practice, it is often convenient to say that massive datasets are relatively defined based on the capacity of machines and are dependent on users' execution urgency. For example, 50,000 records of dataset may be massive for one machine and moderate or small

for another. The intensity of the datasets implies that learning the two constituents of Bayesian networks (network structure and conditional probability tables (CPTs)) takes too much of processor time, probably leading to a halt of the learning process due to limited memory space. This affects business and research deliveries and may hinder the growing usage of the Bayesian networks in some industries that keep massive datasets to build intelligent systems. Since resource management is very precious to users, intelligent system engineers do not want to wait too long to make the reasoning component ready for use.

The massive datasets for Bayesian modelling are often stored on secondary storage devices, probably remotely. Most of the existing conventional algorithms load an entire dataset onto the limited memory for learning a network structure and its probability tables are constructed one at a time. Much time is expended in loading, learning the two constituents of the model and probably saving back, which could otherwise have been minimized. Though some hardware processors may be fast, carrying out the learning process on massive datasets, whose size is more than the available allocated memory, may currently not be practically feasible. Purchasing more expensive hardware (e.g. more memory or networked machines) does not solve this problem but can only support it because the datasets can grow massively. Solving these problems requires scalability of Bayesian learning, which is very challenging. According to Newman et al. [22], a system is said to be scalable when its initial requirements (e.g. massive dataset, more processors) increase, and its performance improves according to the growth. Scalability is further described as being complete if it balances optimization between two resources: (1) computational time and (2) memory usage [22]. Due to the reliability required of any intelligent system, a trade-off between the two resources may not be ideal for real-life problems of emerging the models from massive datasets.

A number of fairly recent studies have developed algorithms for learning good Bayesian networks from datasets, but they fall short in considering the scalability of their approaches [21] [23] [24]. Among the rationales in this research is studying how massive the datasets used in the existing learning algorithms are. Koivisto et al. [21] developed an algorithm for structure discovery in Bayesian networks using several mathematical theorems and propositions. They tested their algorithm on a maximum of 2,500 rows of alarm dataset. Larranaga et al. [23] came up with a variant of a genetic algorithm (GA), using classical GA operators and representing chromosomes as network. Their approach was experimented on a maximum of 3,000 rows of alarm dataset. Friedman et al. [24] proposed a learning algorithm by integrating a Markov Chain Monte Carlo (MCMC) procedure with variable ordering based on approximated posterior probabilities for scoring the network structures. They evaluated their algorithms on a maximum of 1,000 rows of alarm dataset. Myers et al. [25] also developed a variant of a GA by representing genes as variable nodes. They represented classical GA methods as add, delete and

reverse operators which are associated with the K2 algorithms in learning networks. They similarly tested their algorithms on a maximum of 1,000 rows of Asia dataset.

The total computational time for both network structures and their CPTs, and memory usages of the methods described above are not known, though these are key parameters upon which the scalability of learning networks from massive datasets depends. In addition, to minimize computational intensity or nondeterministic polynomial time (NP-hard), few or none of the researchers who proposed these methods claimed successful learning from update observations without re-scanning the entire old dataset. Also, successful modelling in a distributed environment is not guaranteed with most of the existing methods. However, these conventional learning algorithms are often implemented in the available open-source Bayesian learning applications (e.g. Weka [26], and GeNIe [27]). In view of these learning approaches, the basic measurements that influence the computational intensity problem are too many rows, column attributes and states in datasets. For example, in retail environments, sales transactions of products would be rows (or cases); columns might be product names, quantity, price, etc., while states are the number of parameters in each column (e.g. different price values). It is theoretically ideal and simple to have two states per column but real-life datasets have numerous states, which makes modelling more challenging.

Finding solutions to these challenges is difficult. In this thesis, we propose and develop a new class of modelling architecture called the Evolving Dynamic Bayesian Network (EDBN) to address these challenges. The EDBN integrates the temporal probabilistic modelling and economic scalable learning frameworks. The temporal modelling framework derives the emergent situation awareness (ESA) and the emergent future situation awareness (EFSA) technologies. The EFSA is a variant of the ESA, which predicts over current time steps. Situation Awareness (SA) refers to "…the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future [16]". Our research attempts to use the new ESA to bridge this information gap problem by evolving DBNs completely over current time steps, and gaining an understanding of local dynamics from emergent global behaviour. The complete evolvement of DBNs discovers hidden situations from environments captured as MTS datasets. The resulting temporal models from the environment enable decision-makers to reason over time about uncertainties embedded therein. The local dynamics are the smallest pieces of information needed for making correct decisions easily and interactively, while the global dynamics are the higher level of qualitative knowledge about changes in the model. We do not know any existing DBN modelling approaches that reveal hidden situations over current time steps in the same way that the ESA technology does. It therefore becomes important to have detailed understanding of what is currently going on in one's environment before projecting into the future.

The EFSA technology predicts trends over future time steps to mitigate the worries of choosing a DBN model type and to avoid convergence problems when predicting over wider time steps. Its prediction strategy is based on the automatic emergence of temporal models which span a two-dimensional (2D) orthogonal time space from historical Multivariate Time Series (MTS). Our temporal probabilistic modeling framework not only reveals environmental hidden situations but also guides decision-makers in the result interpretation processes in carrying out correct anticipatory planning or arriving at useful conclusions. The economic framework of the EDBN is a dynamic memory management in adaptive learning (DMMAL). This is an alternative solution that provides a complete scalable strategy primarily for the popular single-user machines. It optimizes computational time efficiency and dynamically prevents limited memory from crashing, using representative data partitioning (RDP) algorithms, genetic algorithms, configurable actuators and an adapter operator. The research objectives are summarized in the next section.

1.2 Research Rationale and Questions

In this research, we aim to achieve *situation awareness* by evolving Dynamic Bayesian Networks as temporal probabilistic models. Since most of the existing temporal probabilistic modelling approaches focus more on the explicit HMM representations of DBNs, they are still limited in their expressive power [3]. These existing models are *domain-dependent*, which require highly skilled users, whereas, in most real-life situations, domain experts are often absent. So therefore, we aimed to achieve *domain-independence* to enable our modelling technology to accommodate all users, including non-expert practitioners (e.g. decision-makers, managers, etc.). From our survey, we observed that seasoned programmers without a good mathematical background often struggle to interpret most existing DBN modelling approaches which contain highly mathematical terms. Our modelling includes algorithmic approaches, simplifying the inherent mathematics so that the outline can be easily applied and implemented to solve diverse real-life problems.

The computational intensity (or NP-hard) problems of Bayesian learning are an on-going research area. The ideas suggested such as conventional discretizations for addressing them have not yet given conclusive solutions as they are subject to trade-off. This implies that they optimize time during learning at the expense of limited space or vice versa, and some suggested ideas such as distributed learning on networks are expensive to acquire. Due to these rationales, we therefore formulated the following research questions:

Research question one

Can Dynamic Bayesian Network (DBN) models be explored to achieve the theoretical principles underpinning situation awareness about the behaviour of complex environments?

Answer

In the first instance, DBNs are said to be suitable for complex problems and they are extensions of Bayesian networks [3] [8]. It becomes clear from [8] [16] [28], that an environment is said to be complex if a percentage of its dependent variable Y has effects on a set of its independent variables $\{X_1, X_2,\ldots, X_n\}$. For instance, business, water quality, telecommunications environments, etc. are inevitably accountable for complexity. That is, these environments are inevitably inherently complex due to the many variables dependent on one another which make up their functioning.

In the second place, it is clear that SA comprises three major components, which are the perception of the elements, the comprehension of their meaning, and the projection of their status in the near future. Also, in our own context, DBNs are models that describe systems which dynamically change over time as users monitor and predict their future situations. Therefore, the temporality link between the SA and the DBN is a strong motivation for investigating our first question. Thus, a framework that can completely evolve DBN models is well suited for achieving situation awareness.

Research question two

Can Bayesian network models be learned from massive datasets in a reasonable time within limited memory space and without incurring additional expenditure on expensive new hardware?

Answer

Since one of the computational intensity (or NP-hard) problems of learning BN models from massive datasets is too much memory consumption, it has been affirmed [29] that the dynamic memory management scheme of *Loci* framework is an ideal candidate for managing Directed Acyclic Graph (DAG) decoration problems such as intensity. Also, from what we found, an agent-based system has the capability of speeding up development processes [8] such as the exponential time problem of execution during learning. Our modelling architecture is therefore based on the memory management scheme of the *Loci* framework and the agent-based systems, aimed at minimizing both learning problems to acceptable levels, due to their space and time efficiencies respectively.

Research question three

How can diverse real-world applications benefit from the global dynamics of the proposed modelling architecture to accommodate different users?

Answer

Existing DBN frameworks, which use explicit representations, become popular due to their success in addressing recognition problems [1] [7]. Our proposed modelling architecture will take multivariate time series data as input from diverse application areas (such as business intelligence, water quality management, project profitability analysis, etc.) or otherwise prepare multivariate time series data from their domain datasets without changing their originality. Thereafter, situation awareness and optimization in learning will consequently be demonstrated using the architecture.

1.3 Contributions to Knowledge

It is worth noting that part of the originality of our ideas includes illustrations and the semantics one can associate with them. The following paragraphs therefore highlight the major contributions of this thesis.

In Chapter Two, we critically review the existing DBN models and reveal their limited expressive power based on their representations which lead to approximated results. Our graphical illustrations and relevant comparisons show that the available variety of DBN models have opened up challenging problems on how to choose the most suitable model for specific real-life applications, especially by non-expert practitioners. Also, this thesis identifies and emphasizes the need to integrate Bayesian learning research into DBN frameworks. This would assist in providing more problem-solving models for researchers and practitioners, hence potentially increasing the knowledge base in this field.

In Chapter Three, we present fundamental principles surrounding the DBNs in a simpler way. This reveals the temporality links between the DBN backgrounds and the theoretical principles underpinning Situation Awareness. As extended knowledge is needed in Chapter Four for Bayesian modelling, we describe some sample theories and practical models required for developing algorithms to learn BNs directly from datasets.

In Chapter Four, we identify and place emphasis on avoiding backtracking for enhancing hybrid genetic algorithm (HGA), which learns BN models, and we integrate the ADtree method for its learning speed optimization. The impressive experimental results with public datasets motivate their successful application with regard to response rate to immediate detection of anomalies in telecommunications networks using real-life call datasets.

In Chapter Five, we develop and present the new EDBN architecture to resolve two orthogonal issues using temporal probabilistic modelling and economic scalable Bayesian learning frameworks. We design this new class of DBNs to achieve situation awareness technologies in order to accommodate all users – skilled users and non-expert practitioners. We also develop and design algorithms and models such as representative data partitioning (RDP) algorithms, the conditional probability table (CPT) actuators and configurable discretizer actuators, adaptive operators, etc., that constitute the two frameworks in the EDBN – temporal probabilistic modelling and economic scalable Bayesian learning.

In Chapter Six, we theoretically announce three well defined research niche areas of DBNs which lead to the development of the new ESA technology. We develop the ESA algorithms, which exemplify the improvement on ordinary Bayesian learning algorithms in automatically evolving DBN models, and a new approach to applying variable elimination algorithm to reasoning on DBN models. Additionally, wider real-life applicability of this new DBN technology, other than its popular speech recognition areas, is achieved, such as in the areas of business intelligence, project profitability analysis, and water quality management.

In Chapter Seven, we derive the temporal probabilistic theory for the new EFSA technology which predicts trends over future time steps in the absence of domain experts. We develop the EFSA algorithm, which facilitates the mitigation of the worries of practitioners and researchers in choosing a DBN model type from the multitude of varieties for specific applications. It ensures that convergence problems are avoided when predicting longer time steps using the 2D strategy. This guarantees EFSA wider applicability as it also accommodates all users (experts and non-experts).

In Chapter Eight, we develop the new DMMAL framework which mitigates computational intensity of Bayesian learning, especially on single-user machines, to a minimum acceptable level. We engineer scalable machine learning algorithms and design configurable actuators as combined strategy for the success of the DMMAL framework. The framework is evaluated on publicly available massive datasets using different learning algorithms implemented in Weka and GeNIe systems.

In Chapter Nine, we methodically summarize the thesis and propose some open problems for future researchers.

1.4 Organization of the Thesis

This thesis examines at its basic concept the achievements of situation awareness (SA) technologies integrated with an alternative economical solution to the computational intensity problems of Bayesian learning. The emergent SA technologies are achieved using a new class of temporal probabilistic modelling called evolving dynamic Bayesian networks (EDBNs). The models from the technologies

enable decision-makers to have a better understanding of what is currently happening over time within any complex environment of interest, as they reveal the future hidden situations of the environment.

Chapter Two presents in detail the characteristics of several HMM representations of DBNs and their variants developed by intelligent systems researchers as one of the major concerns of this research. It suggests that Bayesian learning research are inseparable from DBNs, discusses the suggested solutions to address the computational intensity (or NP-hard) problems of Bayesian learning, and evaluates the existing models and methods to make inferences. Chapter Three describes the background of situation awareness and the phenomenon of situation calculus as the theoretical concepts that we used to achieve our new class of dynamic Bayesian networks. The fundamental building blocks of DBNs are described herein, which includes the probability theory, the requirements for capturing the complexities of environments, the general description of Bayesian learning algorithms, the discretization algorithms, and the rudiments of Bayesian inference algorithms that revolve around Bayes' theorem. Chapter Four, describes the system model used for the implementation of the HGA components which integrate the ADtree structure. The detailed approach of our HGA and how its operators interact are presented herein. It includes HGA critical evaluations and the applications to demonstrate immediate detection of anomalies in call data.

Chapter Five presents, without technicalities, our new EDBN architecture which comprises a temporal probabilistic modelling and scalability frameworks. This includes the rationales behind the temporal probabilistic framework, and the rudiments surrounding the economic scalable framework of Bayesian learning. Chapter Six presents the theoretical development of the emergent situation awareness (ESA) technology as a new class of DBN which includes reasoning within these temporal models. The theoretical description as to how to apply the ESA technology which includes a system model extracted from the EDBN architecture and its algorithm are also presented. Chapter Seven presents the theoretical development of the EFSA technology as a variant of the ESA. It includes evaluations of the performance of the EFSA's prediction consistency and accuracy when applied to three diverse areas –future situations concerning rainfall onsets for farmers, treatment of medical patients, and counting people for possible future events. Chapter Eight presents the details of the DMMAL framework and its components, which include the algebraic understanding of discretizer and CPT agent actuators, and the design of configuration for the actuators. It also includes the representative data partitioning algorithms, the adaptive operator, and four experimental evaluations which include the computational time efficiency and memory scalability using publicly available massive datasets. Chapter Nine presents the concluding remarks such as summarising the research results and recommendations, and discusses suggested future research problems.

1.5 Declaration of Recent Research

This thesis is based on our recent accredited publications as follows:

Refereed Book Chapters/Series Publications

- Osunmakinde, I., O. and Bagula, A. (2009) Emergent Future Situation Awareness: A Temporal Probabilistic Reasoning in the Absence of Domain Experts, *In M. Kolehmainen et al. (Eds.): Adaptive and Natural Computing Algorithms (ICANNGA), Lecture Notes in Computer Science (LNCS)*, Springer-Verlag Berlin Heidelberg, ISBN: 978-3-642-04920-0. Volume 5495, Pages 340–349.

- Osunmakinde, I., O. and Bagula, A. (2009) Supporting Scalable Bayesian Networks Using Configurable Discretizer Actuators, *In M. Kolehmainen et al. (Eds.): Adaptive and Natural Computing Algorithms (ICANNGA), Lecture Notes in Computer Science (LNCS)*, Springer-Verlag Berlin Heidelberg, ISBN: 978-3-642-04920-0. Volume 5495, Pages 323–332.

- Balikuddembe, J., K., Osunmakinde, I., O. and Bagula, A. (2009) Software Project Profitability Analysis Using Temporal Probabilistic Reasoning; an Empirical Study with the CASSE Framework. Advanced Software Engineering & Its Applications, *Communications in Computer and Information Science (CCIS)*, Springer-Verlag Berlin Heidelberg 2009, ISBN: 978-3-642-10239-4. Volume 29.

Refereed Journal Publications

- Potgieter, A., April, K. A., Cooke, R. J. E. and Osunmakinde, I. O. (2009) Temporality in Link Prediction: Understanding Social Complexity. *Journal of Emergence: Complexity and Organization*, E:CO Issue Volume 11, No. 1, Pp. 83-96, ISCE Publishers, ISSN: 1521-3250.

- Bagula, A., Osunmakinde, I.O. and Zennaro, M. (2009). Emerging Situation Awareness in Ubiquitous Sensor Networks Using Temporal Probabilistic Networks, submitted for publications.

- Osunmakinde, I. O. and Bagula, A. (2009) Scalable Economic Framework for The Emergence of Bayesian Networks from Massive Datasets, submitted for publication.

Refereed Conference Publications

- Osunmakinde, I., O. and Bagula, A. (2009) EMERGENT SITUATION AWARENESS OF DRIVABLE ROUTES FOR AUTONOMOUS ROBOTS USING TEMPORAL PROBABILISTIC REASONING, *Proceedings of the 14th IASTED International Conference on Robotics and Applications*, Cambridge, Massachusetts, USA. ACTA press, ISBN: 978-0-88986-813-7.

- Balikuddembe, J., K., Osunmakinde, I., O., and Potgieter, A., E. (2008) Software Project Profitability Analysis Using Temporal Probabilistic Reasoning. In *Proceedings of the 2008 Advanced Software Engineering and Its Applications*, ASEA. IEEE Computer Society, Washington, DC, ISBN: 978-0-7695-3432-9, *Volume 00*, Pages 99-102.

- Osunmakinde, I., O. and Potgieter, A. (2008) Astute Decisions In Business Intelligence Using Temporal Probabilistic Reasoning, Competitive Research, *Proceedings of the 20th Annual Conference and Festival of the Southern Africa Institute for Management Scientists (SAIMS)*, Gauteng, South Africa. ISBN: 978-1-86854-729-6.

- Osunmakinde, I., O. and Potgieter, A. (2007) Emergence of Optimal Bayesian Networks from Datasets without Backtracking using an Evolutionary Algorithm. *Proceedings of the Third IASTED International Conference on Computational Intelligence*, Banff, Alberta, Canada, ACTA Press, ISBN: 978-0-88986-672-0, Pages 46-51.

- Osunmakinde, I., O. and Potgieter, A. (2007) Immediate Detection of Anomalies in Call Data – An Adaptive Intelligence Approach. *Proceedings of the 10th Southern African Telecommunications Networks and Applications International Conference (SATNAC)*, Mauritius. ISBN: 978-0-620-39351-5.

Chapter Two

DBN Models – Reviews

2.1 Introduction

Dynamic Bayesian networks are temporal probabilistic models which are often referred to as an extension of Bayesian network (BN) models over a set of random variables, say Z_1, Z_2, Z_3, … [3] [8]. In particular, In [8], modelling DBNs is described as the construction of three matrices (prior, transition and sensor). In [7], a DBN is defined as a model that encodes the joint probability distribution of a time evolving set X[t] = {X_1[t], …, X_n[t]} of variables. This implies that if there are *T* time slices of variables, the DBN appears to be a BN with *T x n* variables. Murphy [3] describes a DBN to be a pair of (B_1, $B_{\rightarrow}$), where B_1 is a BN which defines the prior probability $Pr(Z_1)$ and $B_{\rightarrow}$ is a two-slice temporal Bayesian network (2TBN) which defines conditional probability $Pr(Z_t \mid Z_{t-1})$. His representation as a directed acyclic graph (DAG) leads to equation (2.1).

$$Pr(Z_t \mid Z_{t-1}) = \prod_{i=1}^{N} Pr((Z_t^i) \mid pa(Z_t^i)) \tag{2.1}$$

where Z_t^i is the i[th] node at time t and pa (Z_t^i) are the parents in a DAG. For notational simplicity, he assumes that the parents of a node can be in either the same time slice or the preceding time slice.

Any type of BN or DBN consists of the DAG (qualitative knowledge) and its associated probability theory (quantitative knowledge) captured as conditional probability tables (CPTs) [8] [34]. Murphy models DBNs as a specific type of dynamical system where he varies only the probability distributions while the DAG remains fixed. This implies that he models only the dynamism of quantitative knowledge and makes the qualitative knowledge remain constant over time. This is obviously an approximation which also limits the expressive power of DBNs. This is part of our research concern leading to three well-defined research niche areas of DBNs, which include (1) DBN, whose CPTs and the DAGs do not vary but the representation of the probabilistic process changes, (2) DBN, which varies its CPTs but its DAGs remains fixed, and (3) DBN, which varies its CPTs and its temporal DAGs. Also, as Bayesian learning research progresses, it poses computational intensity (or NP-hard) problems which

some approaches have sought to minimise, but so far have not been able to solve conclusively. These approaches formulate part of the motivation of this research.

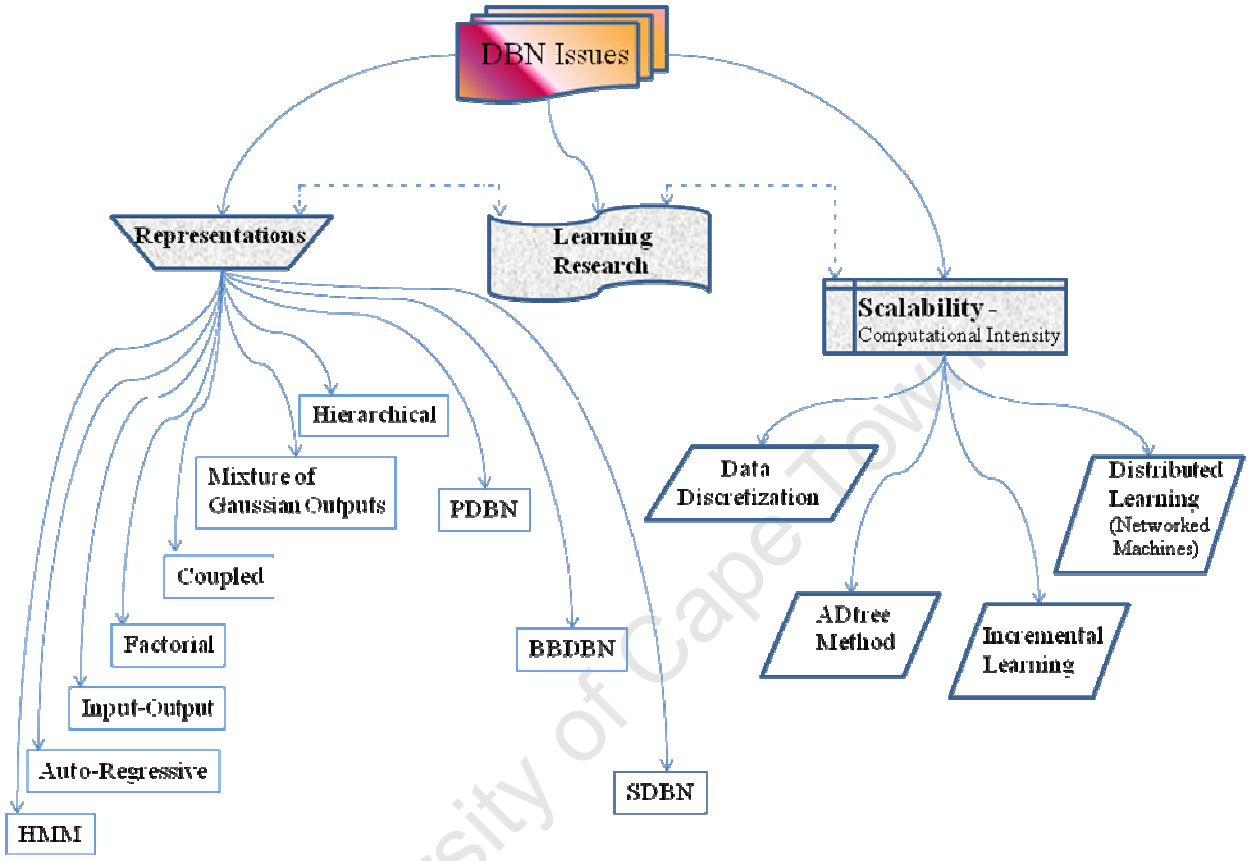


Figure 2.1: Major Issues of Discourse on DBN Models

Dynamic Bayesian Networks (DBNs) research has been conducted in the past, and continues to be carried out on various different levels. The issues of discourse in this chapter are summarised as shown in Figure 2.1. This chapter critically reviews the DBN model representations, learning research, ongoing intensity problems and suggested solutions, as shown in Figure 2.1. The DBN review approach first involves describing the various hidden Markov model (HMM) representations of DBNs and how researchers further advanced the models. HMM is the simplest form of a DBN model, as DBN uses distributed representations of a set of random variables for modelling while HMM uses a random variable. More specifically, various approaches to Bayesian learning research are discussed as they are instrumental to DBN modelling.

To advance DBN research and minimize the rigidity of its existing models, we therefore define a DBN as a belief network that truly evolves over time, by varying its probabilistic distribution and its

Directed Acyclic Graph, from multivariate time series (MTS) data. An MTS is a set of observations recorded at specified time intervals about the attributes describing a domain. This is the environment containing domain datasets where the DBN evolves over time. The time interval could be yearly, monthly, daily, or hourly. The major contributions of this chapter are as follows. It provides:

- Insight into the challenging problems presented by the many varieties of existing DBN models, especially for non-expert practitioners, in knowing how to choose the most suitable model for specific real-life applications. This is a motivation for the rest of the thesis.

- A critical evaluation of existing models and description of how to advance our scientific knowledge by emphasizing the need to integrate Bayesian learning research more into DBN frameworks.

The rest of this Chapter is arranged as follows: in section 2.2, we present in detail the characteristics of several hidden Markov models' (HMMs) representations of DBNs and their variants as developed by intelligent systems researchers. In section 2.3, we present the efforts made in Bayesian learning research which, this thesis asserts, are inseparable from DBNs. Section 2.4 discusses the suggested solutions for addressing the computational intensity problems of Bayesian learning. This includes numeric discretization, the use of ADtree method by Moore et al. [30], etc. Section 2.5 compares the existing models and methods to make inferences. We conclude the Chapter in section 2.6.

2.2 Representations of DBNs

A number of substantial studies have been conducted on some forms of DBN, but its capability has not been fully exploited, as discussed in the following subsections.

2.2.1 Hidden Markov Models (HMMs)

A common form of DBN is the HMM, which is a univariate representation of the dynamical system and has gained its wide applicability in speech recognition [7] [8]. A classical example of Umbrella World is shown in Figure 2.2.

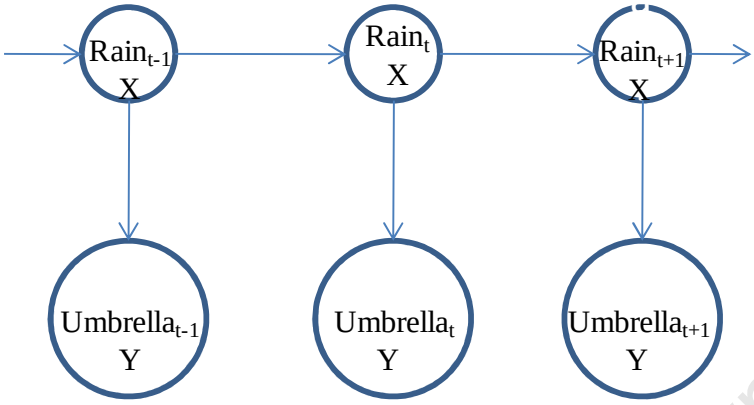


Figure 2.2: HMM representation of DBN for Umbrella World [5]

This is a repeated structure with three time steps. With the conventional definitions, the state variables X are assumed to be hidden, while the evidence variables Y are assumed to be observations of the represented DBN. The interaction of these variables is described by Markov assumptions, which state that $Pr(X_t \mid X_{t-1})$ and $Pr(Y_t \mid X_t)$ [3] [8]. These represent the conditional probability distributions of the model variables. More information on representing the CPTs of this HMM is similar to what was described in Chapter One. However, HMMs formulate the basis of most existing DBN representations. Murphy [3] carried out valuable extensive research on DBN models. He presented many variants of HMM as explicit representations of DBNs with different characteristics. Examples of his representations are Hierarchical (HHMMs), Auto-regressive, Abstract, Input-output, Factorial, Coupled HMMs and HMMs with mixture of Gaussian models.

2.2.2 Auto-Regressive HMM

The three time-steps DBN model represented in Figure 2.3 is an auto-regressive HMM [3] which modifies the Markov assumption stated above for classical HMMs. It additionally makes observation Y_1^{t-1} to contribute to the predictions of Y_2^t, etc. It is believed that more connections contribute to better prediction results. For the discrete variables, the CPTs here can also be estimated using learning algorithms such as maximum likelihood estimate (MLE) [8] [35].

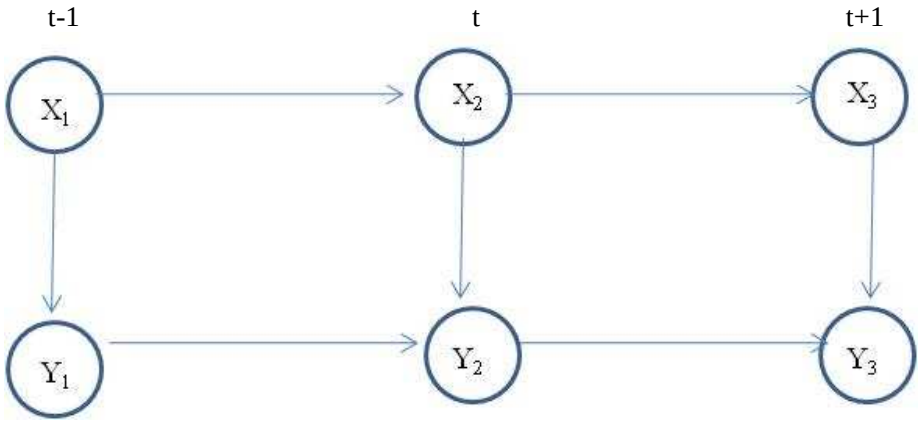


Figure 2.3: Auto-Regressive HMM

2.2.3 Input-Output HMM

Figure 2.4 shows an input-output HMM [3] [4] represented as a DBN. It is a probabilistic mapping of inputs from $U_{1:T}$ to outputs $Y_{1:T}$. For discrete input variables, the CPTs of variables X can be estimated in a three-dimensional matrix of $Pr(X_t = x_i \mid X_{t-1} = x_j, U_t = u_k)$

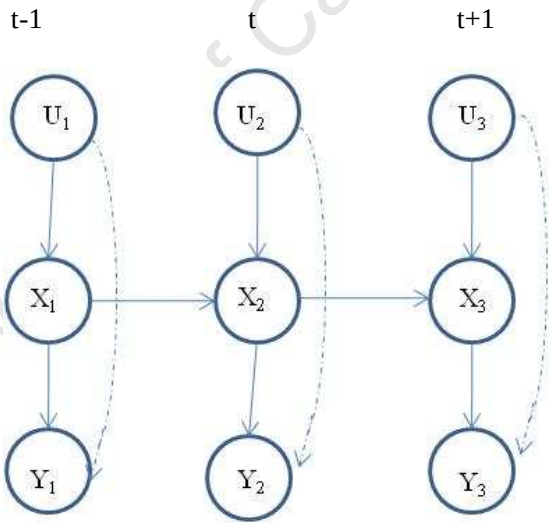


Figure 2.4: Input-Output HMM

2.2.4 Factorial HMM

Factorial HMM [3] [6] is another DBN represented in Figure 2.5. It has distributed representations of hidden state variables with a single output variable at each time step. Every chain of hidden variables *X* at each time step *t* is independent of the other. The CPTs of the hidden variables, $Pr(X_t \mid X_{t-1})$ can be

estimated as N_x by N_x matrices. Also, the CPTs of the observed variables, $Pr(Y_t \mid X_t, X_t, \ldots, X_t)$ have a tendency to be huge, due to all possible combinations of parents.

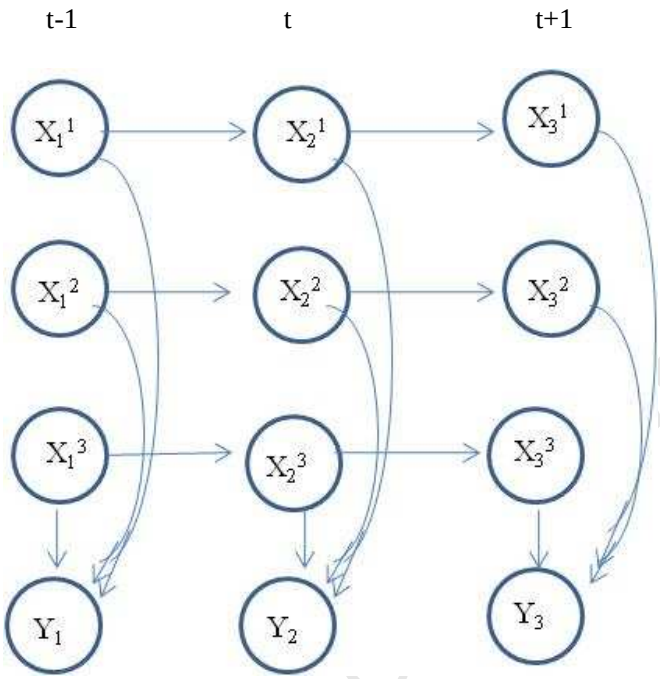


Figure 2.5: Factorial HMM with Three Chains

2.2.5 Coupled HMM

For another DBN representation called the coupled HMM [3] [5] as shown in Figure 2.6, the hidden variables are allowed to interact locally with their neighbouring variables. One can see in Figure 2.6 that every hidden variable has its local observation variable. It is a repeated number structure of HMMs that are connected.

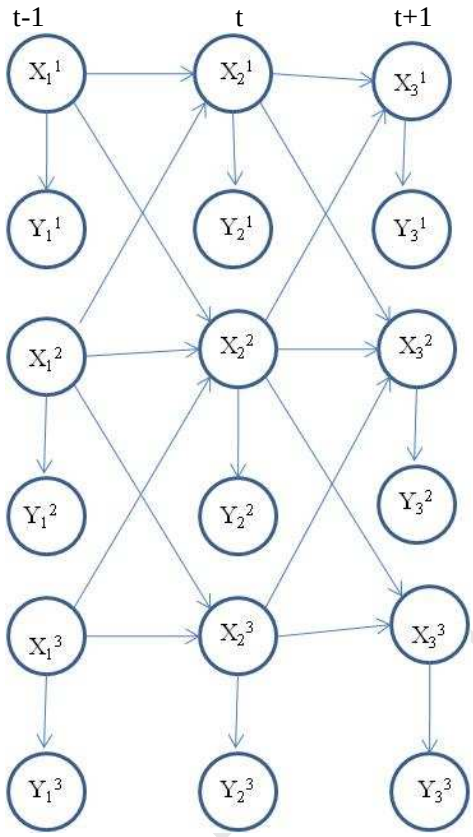


Figure 2.6: Coupled HMM with Three Chains

2.2.6 HMMs with Mixture of Gaussian Outputs

HMMs with Mixture of Gaussian Outputs [3] are another DBN represented in Figure 2.7. They are explicitly modelled and are popularly used as speech recognition models. Their associated CPTs of the observation variables, $Pr(Y_t \mid X_t = x, M_t = m)$ and $Pr(M_t = m \mid X_t = x)$, are estimated using a mixture of Gaussians models for every time step.

Hierarchical HMM [3] is an advanced form of HMMs that is used to model environments using hierarchical structures. In the hierarchical HMM, the states (otherwise called production) of stochastic (probabilistic) automation on the model can emit single observations, while the states (otherwise called abstract) emit strings of observations. The model generates a kind of recursive regular expressions (e.g. alphabetic characters). This is why it has gained popularity for speech recognition purposes [3].

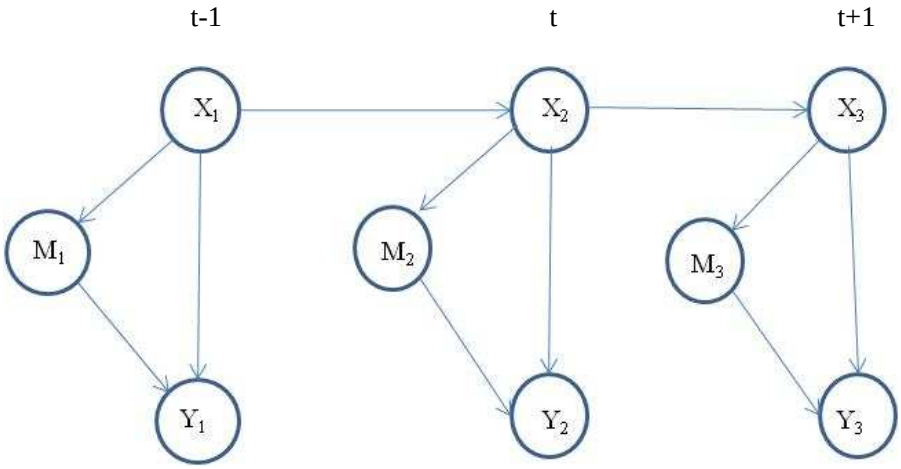


Figure 2.7: HMM with Mixture of Gaussian Outputs

These models have contributed to the baseline of temporal modelling but it becomes challenging to non-expert practitioners (e.g. decision-makers, managers, etc.) to apply these DBN models to reason about any complex environments in the absence of domain experts. Using these DBN model types requires highly skilled users because several characteristics, assumptions and natures of the domain problems are considered in modelling. This may become too complicated for non-expert users showing interest in the technology. Some other variants of HMM representations of DBNs which seem to be domain dependent on the applications have been proposed by other intelligent system researchers [2] [7] as well.

2.2.7 Partial Dynamic Bayesian Networks (PDBN)

In an attempt to improve the explicit DBN representations of HMMs, or remove their rigidity, to allow for DBN applications in a wider range of real-life areas, a few research efforts have been made to learn DBN models, but they are still limited in their expressive power. For convenience, they make stationarity assumptions which results in repeating the same structure over time.

Chang [36] presented dynamic stochastic systems called partial dynamic BN (PDBN), where observations are received over time. The method here interconnects multiple instances of same fixed networks (BNs) with temporal relations into the DBNs. The PDBN identifies a minimum set of state variables to separate evidence nodes (or variables) between two time steps. This in turn reduces the sizes of the CPTs. The state variables are referred to as transitional nodes. Figure 2.8 shows a simple two-time-steps PDBN where the transitional nodes are $\{T, A_{j+1}, B_{j+1}\}$, which separate the two sets of evidence nodes $\{E_j, F_j\}$ and $\{E_{j+1}, F_{j+1}\}$. Observe that same networks are repeated over time (stationarity assumptions).

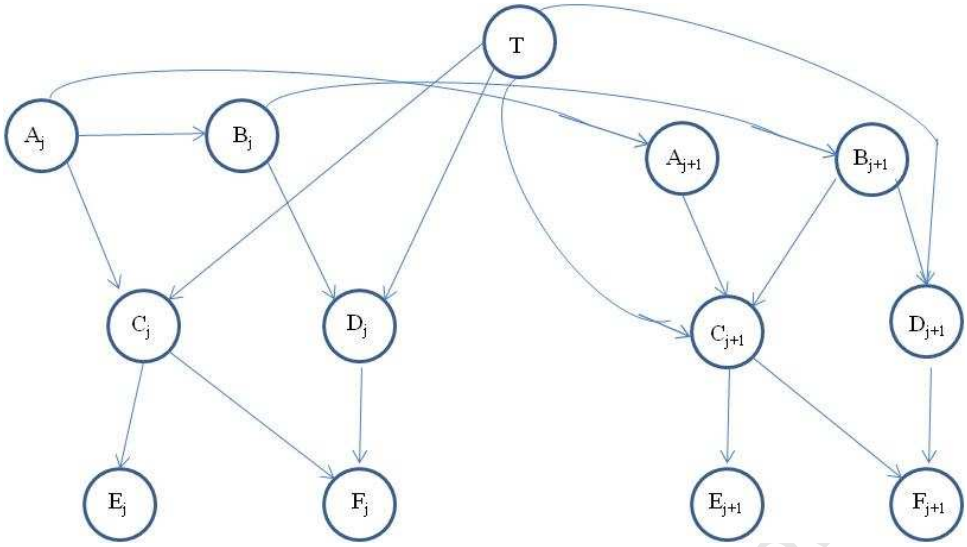


Figure 2.8: A Sample Partial DBN (PDBN)

Suppose T is the target node and all variables on the network are discrete. If one intends to reason about T under uncertainty given the accumulated observations up to the current time, it can be estimated as:

$Pr(T \mid E_j, F_j, E_{j+1}, F_{j+1})$.

2.2.8 Brain and Body DBN (BBDBN)

Shenoy et al. [2] developed a DBN framework that is specifically dependent on the characteristics of the domain of EEG and EMG signals. That is, they model the dynamics of hidden brain and body states. As shown in Figure 2.9, at each time instant t, the brain state B_t generates the EEG and EMG internal states E_t and M_t respectively. These in turn generate the observed EEG and EMG variables. For expert convenience or by approximating the models, observe that the same network structure is assumed and explicitly represented as their DBN over time, while it was claimed in [2] that the changing probabilities are learned from the input data.

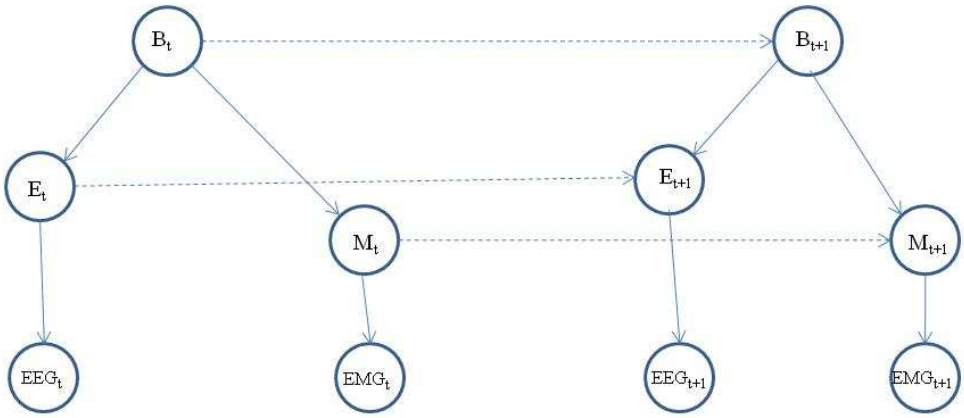


Figure 2.9: A DBN Modelling Brain and Body Process

2.2.9 Structural DBN (SDBN)

In [7], structural learning of DBNs is presented which is also applied to speech recognition. The method predefined the search class that is of interest. This implies that any structures of the DBN learned are limited within that confined class. This is an attempt to minimize computational intensity (or NP-hard). The major assumptions or rules of the search class are: (i) no direct dependencies are considered between observable variables, (ii) there are no future dependencies between the hidden variables, (iii) no hidden variables should have observable variables as parents, and additionally, (iv) transitions of hidden state variables are assumed to be stationary. That is, a learned network structure is repeated over time but with changing probability using some dynamic processes. This is similar to the explicit representations of the DBNs presented in the preceding paragraphs. Figure 2.10 shows a DBN of this type with triple (k, τ_p, τ_f), which defines the upper limit of the size of the search class.

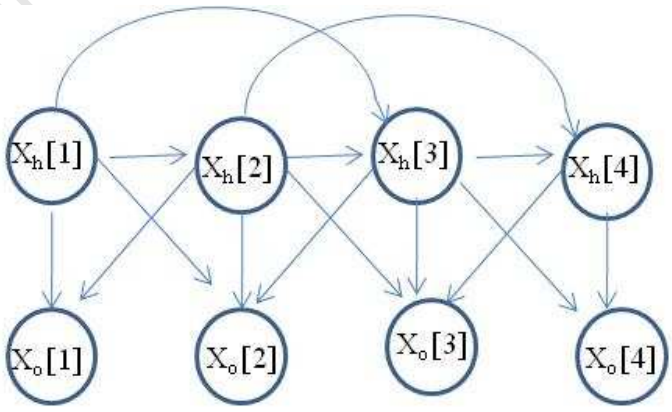


Figure 2.10: A DBN model with (k, τ_p, τ_f) = (2, 1, 1), T = 4

Several other recent references can be found that address the problem of dynamic modelling with Bayesian Networks. However, network structures are usually the visualized qualitative knowledge of the models and dynamic behaviour of the network structures to capture environmental changes which do assist in decision making is often avoided. In [9], a DBN-based model is used to measure security in networked information systems environment. The attack model is a special DBN derived to monitor the existence of vulnerabilities on a host or the connectivity between hosts. A two-slice temporal Bayes net (2TBN) that defines a DAG including collection of BS (base score) and TGS (temporal group scores) vertices is derived by domain experts to measure vulnerabilities. In this case, conditional probabilities depend on the expert knowledge and are assigned to the network. For instance, when a vulnerability has been successfully exploited in one time slice, then its probability of being exploited in the next time slice is equal to '1'. The work claimed to have good practical framework but no real-world experiment is conducted.

Chapelle et al. [10] proposed that click model based on DBN can be used to model users' browsing behavior. The modeling is based on several assumptions such as: (i) if two documents in position 1 and 2 are very relevant, it is likely that this document will have very few clicks, and (ii) if the two top documents are irrelevant event, it will have a lot of clicks, etc. The work presents a repeated network structure very similar to the standard HMM to introduce the notion of satisfaction and separately model the relevance of landing page at the search result page. Only one variable of the click model is observed while other variables are fitted to follow some statistical distributions such as beta and binomial functions for estimating probabilities. Most of the DBN results are either increasing or decreasing exponentially. This model can work in a controlled or smooth environment but may be challenging in real life situations. However, it is admitted by the authors that some of the assumptions about the model are not realistic.

In [11], activity recognition from interactions with objects using DBN is presented. Specifically, the method here detects an interaction between a nurse and a tool using a RFID tag system. From the interaction data, activities are recognized using the DBN model which is used to solve some misclassification problems in a drip injection task. A 2-slice temporal Bayesian Network (2TBN) equivalent to HMM with Q_t represents a step at time t to be classified, and O_t is the interaction data. When the interaction data O_t in time t is given to the DBN, the result of recognition Q_t in time t is obtained. In their experiment, Bayes Net Toolbox for MATLAB is used and the results obtained in time series is also exponentially increasing. When the DBN model is compared with a decision tree model, accurate results are guaranteed for the first three steps, but still admitted to overcome the problem of misclassifications. A more varying model called switching HMM is suggested for future work, which is

in the direction of our model idea herein. However, a represented HMM is used but the method used in the toolbox for estimating probabilities to perform the recognition is not explained.

Another proposal of DBN for analyzing air combat (AC) simulation results can be found in [12]. Here, the DBN model was applied in analyzing the evolution of AC scenario as well as the effect of different observations on the evolution and the final AC outcome. A repeated network structure and probabilities of the DBN are estimated from the AC simulation data produced with a discrete event AC simulation model (X-Brawler) using Hugin Bayesian software. A complex DBN structure is defined by knowledge engineer so that a single time slice consists of the variables B_t, R_t, and O_t, where O_t is a function of B_t and R_t. The analysis of the DBN is helpful as it does not necessitate the re-screening of the original simulation data. But it remains to be explained why the DBN model used is not compared with related DBN model results. This is probably due one of the problems we are solving herein - challenges of choosing the appropriate DBN model to address similar problems.

Shermin et al. [13] developed a DBN based model to infer gene regulatory networks (GRN) from expression profiles and for identifying regulatory interactions among genes with a special focus on computational cost. The method decomposes the inference problem into several smaller overlapping sub-problems using biological features of the domain. Using the domain knowledge, irregular genes are cut down from the candidate regulators set and thus reducing the search space. The computational time and accuracy of their model claim to better than that of Bayes Net Toolbox written and freely provided by Murphy [3]. There are no experiments about the performance of temporal trends of their model are given. It becomes very difficult to reproduce their method because no algorithm is explicitly stated.

In [14], a DBN based interest estimation for visual attention presentation agents is presented. The NIIScore (interest or non interest) DBN is used to estimate the interest of the user by taking into account which interface object the user pays attention to. A set of requirements is required for modelling their DBNs such as: (i) a measure for the interest of the user in the presentation, (ii) a measure for the interest of the user in some particular screen object, and (iii) a measure of the time the user's eye gaze dwells on an object. A new time slice is attached to the DBN of each interest object in the scene and a roll-up is performed for the previous time slice. A roll-up is a repeated connection of a fixed network structure to a DBN. Assigning conditional probabilities to the network is based on some form of thresholds such as IScore (interest score) and FIScore (focus of interest score). The work requires a knowledge engineer to model the NIIScore DBN using the JavaDBN tool. But again, no experiments about the performance of temporal trends of their model are given. The work does not compare the model with other related DBNs. We assume the work is not also sure of which modelling approach to compare with. This is one of the information gaps that this thesis is trying to bridge by providing a generalized DBN modelling.

Another proposal of DBN for privacy intrusion detection can be found in [15]. In this privacy protection model, data is protected when data has to be disclosed or even when data is secured. Features of the model are manually extracted by domain expert from database's operator activities since the past of an operator can be used to determine its future. It is assumed in their model that the nodes, the dependencies among nodes, and the strength of the dependencies at slice *i* are identical to those at slice *j*. It is also assumed that the dependency and its strength between a pair of nodes across two consecutive instants will not change over time. Conditional probabilities are subjectively assigned from expert knowledge. This is a typical example of a DBN with fixed network structure and fixed probabilities. Hence, a 2TBN model is used for detecting the intrusion behaviour and unrolled over time. Inability to compare models with other related DBNs is also a problem here. It is evident that this is a general challenge among DBN researchers. The researchers acknowledged learning a DBN privacy intrusion detection model from real data as we propose in this thesis to ease their laborious work and improve the quality of results.

In reality, situations in some time steps may change. It is evident that most of these existing DBNs approximate their models. That is, they do not truly (or completely) evolve the network structures (or DAG) and probability distributions, but the basis of DBN or BN requires that both be modelled [8] [37]. Murphy [3] confirms that the HMM, which is the basis of most of these representations of the existing DBNs, is limited in its expressive power. According to Russell et al. [8], HMMs are unsuitable for complex problems. A framework that can directly reveal uncertainties over time about relationships between any domain variables will be well suited to achieve situation awareness in complex environments. Most of these recent modelling approaches presented are focused on one direction and do not claim the capability of handling situation awareness, which is usually of greater value in industry. These existing frameworks used explicit representations of DBN based on the HMMs. However, it is the varieties of these DBNs that have opened up challenging problems, especially for non-expert practitioners, of how to choose the most suitable model for specific real-life applications.

In view of all these issues, we propose an alternative approach to DBN modelling in this thesis, one that truly or completely evolves the DAGs and the CPTs over time. This creates awareness of hidden situations over time through the use of various technologies (e.g. the ESA and the EFSA), and practicable machine learning algorithms we developed therein. With little or no effort, this makes it easier for non-expert practitioners to implement and use DBNs for effective decision-making processes.

A possible future challenge for DBN learning from data, which is an ongoing research effort in ordinary BN modelling, is computational intensity (or NP-hard) problems. Section 2.4 presents suggested solutions for addressing these problems. However, we provide an alternative economical solution in this

thesis which is presented in detail in Chapter Eight. Various BN learning research studies that can be upgraded or adapted to learn DBN models are discussed in the next section.

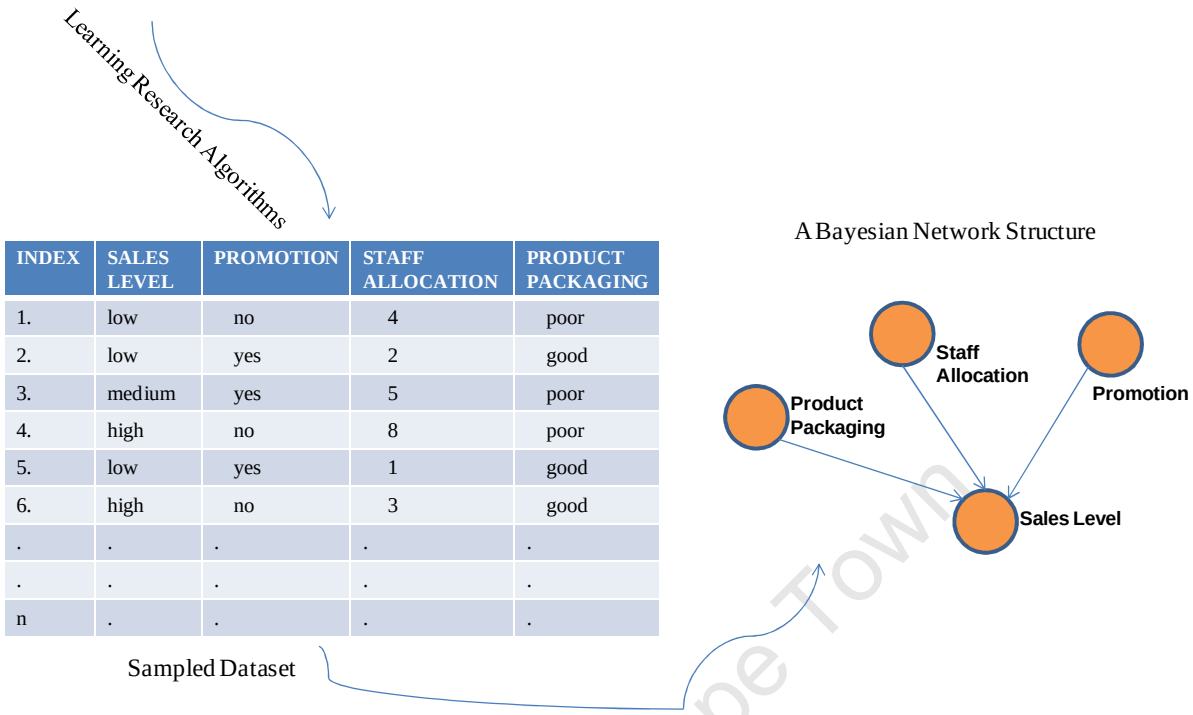
2.3 Bayesian Learning Research

Learning is an intelligent system technique of discovering Bayesian network models from environments captured as datasets. Figure 2.11 illustrates how learning research algorithms discover network structures from datasets. The followings recent studies in the subsequent paragraphs have been conducted on Bayesian learning research and have the potential to be improved and integrated into DBNs to learn models over time. The process of this improvement is, however, beyond the scope of chapter, as this section is focused on reviewing the existing BN learning research.

Wai et al. [38] developed the Minimum Description Length Evolutionary Programming (MDLEP) to learn BN structures based on the minimum description length (MDL) principle and on evolutionary programming. The MDL is a scoring function for measuring the fitness of models from datasets, where the scores are depicted in bits. In his approach, he integrated the MDL metric from information theory and developed variants of classical genetic operators which include: structure-guided mutation, knowledge-guided mutation, and freeze and defrost operators to learn Bayesian networks from data.

These operators reproduced new offspring from a parent network. Specifically, they defined structure-guided mutation as randomly adding edges, deleting edges, reversing edges and moving one node to another. That is, an edge is randomly added between two nodes if not already existing. If the edge already exists, it deletes it. Also, it randomly changes the direction of an edge and increases/decreases the number of parents of a child node to derive a new structure.

Knowledge-guided mutation is defined as using MDL to determine which edges should be removed or inserted in a network. For every edge it adds or removes, it computes and keeps its MDL. That is, if it decides to add a new edge to the current structure, it first considers and compares it with the already-stored MDL score and chooses the lower MDL score. Also, the freeze and defrost operators were used to identify and repair redundant nodes. That is, when there are several modifications to the network and there is no improvement, it prevents further changes on the network. This variant type of GA is claimed to discover extremely good networks. Some of their operators can be combined to save time.



INDEX	SALES LEVEL	PROMOTION	STAFF ALLOCATION	PRODUCT PACKAGING
1.	low	no	4	poor
2.	low	yes	2	good
3.	medium	yes	5	poor
4.	high	no	8	poor
5.	low	yes	1	good
6.	high	no	3	good
.	.	.	.	.
.	.	.	.	.
n	.	.	.	.

Figure 2.11: Learning Research Discovering Network Structures from Environments using Algorithms

Larranaga et al. [23] used a GA to generate variable orderings that are representations of chromosomes. The GA searches for a near-optimal ordering between variables. The variables evolved several networks during BN learning from complete data. They used genetic operators, such as rank selection to select parents for reproduction, crossover and mutation to generate new individuals. They used a K2 algorithm to estimate the quality of Bayesian structure from the variable ordering. K2 is a greedy heuristic algorithm that continually adds new nodes (or variables) to a current learning network until the node cannot maximize the probability score of the network. More information on heuristic algorithms is presented in Chapter Three.

Larranaga et al. showed in their experiments that a GA can discover good networks from data. There was no clarity though, on how their results proved better than what they had addressed in the literature survey of their work. For future research, it was intended to use only genetic algorithms from the best variable ordering obtained.

William et al. [39] identified sensitivity to variable ordering as a shortcoming of most greedy score-based algorithms such as K2. Variable ordering is the combination of attributes to form nodes in Bayesian structure learning. When an algorithm is sensitive to the ordering of network variables (A, B, C, D), it implies that the emerged optimal network of input pattern {B, C, D, A} will probably be different

from input pattern {D, A, C, B}. They developed a genetic algorithm to mitigate this problem by searching the permutation space of variables using a fitness function. Specifically, William et al. used an order crossover (OX) to generate a population from six variable nodes. For instance, OX reproduces offspring:

$$O_1 = \{6,2,\underline{5,3},1,4\} \quad and \quad O_2 = \{5,3,\underline{6,2},4,1\}$$

from parents:

$$P_1 = \{3,4,\underline{6,2},1,5\} \quad and \quad P_2 = \{4,1,\underline{5,3},2,6\}$$

Also, they implemented mutation as swapping of number positions as shown. Moreover, they refer to P_1, P_2, O_1 and O_2 as individuals. However, they claimed that they have minimized part of the K2 limitations and therefore, proposed that their results have the potential to use the GA only to learn networks in future research.

It is interesting to note that a similar approach is used by Myers et al. [25] who also made representations of GA elements and operators such as gene, crossover, etc. They developed a variant of a genetic algorithm as described below. They illustrated structure learning with five variable nodes, where each node is represented as a gene and a network structure is represented as chromosomes. Myers et al. used uniform crossover and mutation as add, delete and reverse operators that are commonly used in K2 algorithms. They illustrated with two parents simultaneously as follows:

Parents (P_1 and P_2) →

P_1 = [{A} {B A} {C A} {D B C} {E C}], P_2 = [{A B} {B} {C B} {D B} {E C D}]

offspring (O_1 and O_2) →

O_1 = [{A} {B A} {C B} {D B C} {E C D}], O_2 = [{A B} {B} {C A} {D B} {E C}]

These operators are used to modify the chromosome structure, but they are prone to cyclic structures. The improvement on what they used as genetic operators paves the way for future research.

Friedman et al. [24] proposed an algorithm by integrating a Markov Chain Monte Carlo (MCMC) procedure with variable ordering based on approximated posterior probabilities for scoring and finding

the network structures. This approach finds the posterior distributions on orders of variables rather than in network structures. They have two main ideas in their approach. They developed an expression to find the posterior probability of data, given a known order of network variables. An order of variables is defined as a total order on the index set of the variables [21]. In the second instance, they used the Monte Carlo sampling method to find the posterior probability over some sets of orders.

Friedman et al. estimate probabilities of some edge features from given orders of network variables. They sampled network structures from different orders of network variables by doing a random walk while they estimated the probabilities to investigate for the presence or absence of some other structural features in the structures. They repeated this process to return a representative final network that contained most of the features, and claim that their approach is reliable for learning Bayesian networks from data.

Koivisto et al. [21] developed two versions of exact algorithms for structure discovery in Bayesian Networks using several mathematical theorems and propositions. It was presented that learning optimal Bayesian networks from data is a computationally hard task, but infinite networks can quickly be generated from data. This served as motivation for the researchers, as deriving optimal networks from data has not been extensively studied.

A version of their algorithm uses an exact posterior probability to find a quick network, usually a DAG, from data. The second version of the algorithm focuses on finding optimal network structures from data. From their investigations, they came up with three observations, as follows: (i) it is necessary to consider all possible local dependency structures to derive an optimal network; (ii) for sufficiently small datasets with a small number of network variables, it is believed that exact algorithms can afford to exhaust a solution space to generate an optimal network structure; and (iii) using an exact algorithm on large datasets to find optimal networks takes almost super-exponential time. Koivisto et al. strongly urged that exact Bayesian learning algorithms be improved, so as to minimize computational complexity problems. These observations are therefore strong motivations for researchers in this field, that advanced frameworks will be required for all possible learning processes rather than relying completely on an algorithm. From Koivisto et al.'s contributions, they modified the exact algorithm to learn multiply connected larger networks better than Monte Carlo methods. They used strategy that suitably restricts a priori approach on the network structures during learning processes.

This is what makes researchers and practitioners stress that learning Bayesian networks from massive datasets is a computationally intensive problem which requires ongoing research [20] [23]. The intensity on the datasets implies that learning the two constituents of Bayesian networks (network structure and conditional probability tables (CPTs)) possibly comes to a halt after too much processor time has been taken. Some methods proposed to address this problem are presented in the next section.

2.4 Addressing Computational Intensity in Bayesian Learning

Since finding optimal Bayesian Network models from datasets has been shown to be computationally intensive [20] [21], recent studies [23] [25] [39] have investigated the challenges of learning networks from data. Some approaches have been proposed to support optimization of learning the network structures by minimizing the computational intensity (or NP-hard), as presented in the following subsections.

2.4.1 Predominant Approach – Data Discretization

Discretization is an important pre-processing step in machine learning [40] [41]. Discretization classifies numerical datasets into their corresponding interval values relative to the patterns in the data attributes. This is a predominant current approach.

A number of fairly recent studies have developed good conventional algorithms to discretize datasets but they fall short in considering the scalability of their approaches [40] [41] [42] on massive datasets. According to Newman et al. [22], an algorithm is said to be scalable if its initial requirements (e.g. massive dataset, more processors) increase and its performance improves according to the growth. The environment where these algorithms will experience the worst scenario of computational intensity (NP-hard) is the massive datasets. Among the rationales in this section is studying how massive the datasets used in the existing discretization approaches are.

Waldron et al.[41] proposed Genetic algorithms (GA) as a data discretization method where they use genetic operators such as crossover and mutation to clone chromosomes (discretization scheme), and change genes of chromosomes at random, respectively. They represented genes as a possible cut point in the datasets. Since a chromosome refers to a collection of genes in biology, it was represented as one possible discretization of an input file. A GA-Discrete Java application was adopted in their experiments to discretize datasets.

Waldron et al. claim that their GA produces results that are comparable with other discretization algorithms. It was however remarked that the stopping criterion of the GA is quite informal and could be inappropriate for datasets that feature a high degree of interdependencies between data variables.
Their GA technique was evaluated on a maximum of 48,000 records of Adult dataset.

Li et al. [42] suggested feature selection heuristics for discretizing bio-medical data. They used mean entropy as a measure of discriminating power to select the appropriate features to determine the discretized intervals.

In their approach, they rank all features into an ascending order according to their computed entropy values. They do not consider those features that are ignored by the entropy measure. They claim that this first feature selection reduces the dimensionality of the dataset by 90% to 95%. They calculate the average of the entropy values of the remaining features. They then rank the rest of the features and select those features whose entropies are less than the average of all the rest of the entropies. This implies that the entropies with smaller values are more discriminating. It was presented that this second step further reduces two-thirds of the features in the dataset. It was concluded that thousands of features can be accurately reduced to hundreds of important features. They also evaluated their approach on a notable lung-cancer dataset of 10,000 records.

Lee's supervised algorithm [43] is similar to Li et al.'s, but he used the entropy of intervals to achieve high quality discretization as a concise summarization of numeric attributes. This type of information theory maximizes intra-interval uniformity and minimizes inter-interval uniformity. This makes his approach context-sensitive in such a way that it takes into account the values of the target dataset variable. This approach uses the Hellinger divergence to measure the amount of information that each interval contributes to the target variable. This is otherwise known as the difference between the class frequencies of the target variable and the class frequencies of a given interval. The interval boundaries in this method are generated in such a way that every interval contains an equal amount of information as far as possible.

He proved the efficiency and practicability of this discretization method in such a way that it only requires users to provide the maximum number of intervals needed. He proposed, however, that one should work towards achieving the capacity to distinguish between true correlations and coincidence in future, if necessary. Lee used a maximum of 3,163 records of hypothyroid dataset to evaluate his work. This record size may be considered not massive enough in other applications.

The holistic taxonomy of methods compared by Dougherty et al.[40] is supervised and unsupervised discretization of continuous features in datasets. They identified the defining characteristics of several methods and carried out empirical evaluations on them. In particular, they compared simple binning, which is an unsupervised discretization method, to the entropy-based and purity-based methods which are supervised algorithms.

It was observed that some Bayesian modelling algorithms were significantly improved when the entropy-based method was used to discretize datasets. Lee's approach [43] is an example of the supervised method, since instance labels are considered to determine boundaries and we adopted this approach in our framework in Chapter Eight of this thesis. However, out of the 16 datasets used by Dougherty et al., the maximum size is an Australian dataset with 6,650 records.

Ismail et al. [44] confirmed Dougherty's work as they investigated the impact of discretization methods on the distributions of various datasets and found out that they work well. Their objective was to enlighten researchers as to how to choose the most appropriate discretization methods for specific applications. In their analysis, they induced a decision tree classifier and an error measure to determine the effectiveness of the discretization methods. It was also concluded that the supervised discretization approaches are superior to the unsupervised methods.

Thus, in an attempt to address the computational intensity challenges, the available open source network learning applications such as Weka [26], GeNIe [27], Hugin [45], etc., that implement these algorithms, force users to discretize all numeric values of the attributes present in the datasets. However, certain numeric attributes in real life, such as *registration* numbers, *months*, *years*, or *days* that are captured as numeric values, are not necessarily required to be discretized.

2.4.2 ADtree Data Structures Method

This subsection describes the ADtree method that contributes towards addressing the intensity problem. Moore et al. [30] proposed the ADtree method for generating sufficient statistics from datasets. They load it onto memory prior to Bayesian learning. It is a smart approach for reducing the data sizes but has its limitations. The ADtree is assumed to perform better when tested on discretized datasets rather than when operating on continuous values.

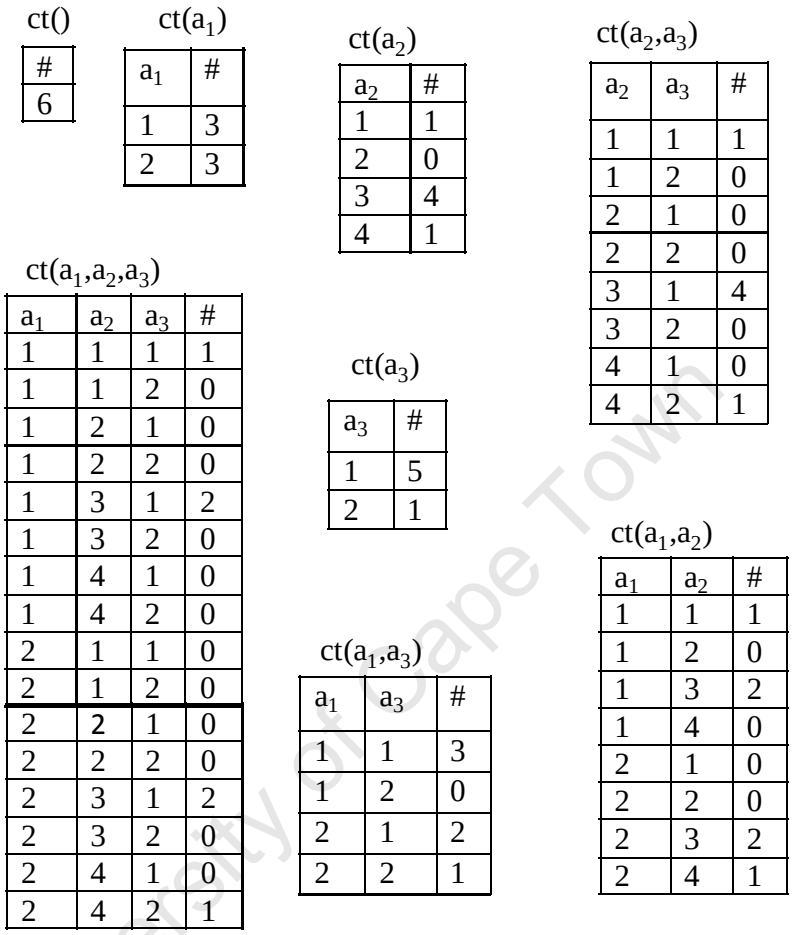
The ADtree is a sparse data structure which attempts to minimise memory usage as it focuses on counting the number of records in a dataset that match conjunctive queries, in order to construct contingency tables. The data structure is constructed in such a way that (i) it uses a sparse tree structure that never allocates memory for counts of zero, (ii) it never allocates memory for counts that can be deduced from other counts, and (iii) it does not bother to expand the tree fully near its leaves. Moore et al. illustrated with a dataset of R, say 6 records and M, say 3 attributes, represented in Table 2.1.

Table 2.1: An Assumed Sample Massive Dataset

a_1	a_2	a_3
1	1	1
2	3	1
2	4	2
1	3	1
2	3	1
1	3	1

The contingency tables are the building blocks of the ADtree structure. Each subset of data attributes, $a_i(1)$, …, $a_i(n)$, is said to have an associated contingency table represented as ct($a_i(1)$, …, $a_i(n)$). So, for the

three attributes in Table 2.1, the expected number of contingency tables is 2^3, which implies eight. The tables are shown in Figure 2.12.



ct()

#
6

ct(a_1)

a_1	#
1	3
2	3

ct(a_2)

a_2	#
1	1
2	0
3	4
4	1

ct(a_2,a_3)

a_2	a_3	#
1	1	1
1	2	0
2	1	0
2	2	0
3	1	4
3	2	0
4	1	0
4	2	1

ct(a_1,a_2,a_3)

a_1	a_2	a_3	#
1	1	1	1
1	1	2	0
1	2	1	0
1	2	2	0
1	3	1	2
1	3	2	0
1	4	1	0
1	4	2	0
2	1	1	0
2	1	2	0
2	2	1	0
2	2	2	0
2	3	1	2
2	3	2	0
2	4	1	0
2	4	2	1

ct(a_3)

a_3	#
1	5
2	1

ct(a_1,a_3)

a_1	a_3	#
1	1	3
1	2	0
2	1	2
2	2	1

ct(a_1,a_2)

a_1	a_2	#
1	1	1
1	2	0
1	3	2
1	4	0
2	1	0
2	2	0
2	3	2
2	4	1

Figure 2.12: Eight Contingency Tables for the Assumed Dataset in Table 2.1

The contingency tables are now logically grouped together to build the ADtree data structure. The ADtree is actually adopted in Chapter Four to accelerate a Bayesian learning algorithm we developed. The data structure method, however, falls short of handling massive datasets in Bayesian learning, as the memory consumption grows exponentially with a rapid increase in the number of attributes. The resulting ADtree structures generated become so huge that it crashes the memory allocations. An incremental learning approach was also devised to address the computational intensity problem, as discussed in the next subsection.

2.4.3 Incremental Learning

Alcobe [46] proposed a highly mathematical TOCO (traversal operators in correct order) heuristic approach to transform batch hill climbing algorithms to incremental methods. It was noted that there is

not enough work on incremental and its related approach to optimize Bayesian learning. This implies that there is a need for simpler approaches that can be easily implemented by everyone, such as seasoned programmers without good mathematical backgrounds.

His incremental hill-climbing Modelling method learns Bayesian structures from data. TOCO keeps the order of the models present in its former learning path. According to Alcobe, if new data instances are available and the order of the models changes, a new model is learnt from scratch. Otherwise, the old model is retained. TOCO combines the operators of K2 with an incremental search to optimize on learning time.

Alcobe explained that incremental hill-climbing algorithms are very sensitive to ordering effects. That is, when two sample orderings O_1 and O_2 of a training set are presented to the incremental algorithm, they are prone to output different domain models. Changes in sample orders is often believed to be the way a dataset is being incrementally prepared and used for learning. Outputting different models from same domain is a general research concern that must be avoided. He acknowledges that the quality of their final networks is not the same as that of the networks obtained when batch algorithms are used, as in the research discussed in section 2.4.

Alcobe said that the objective of his incremental approach is to reduce the time spent in learning only new network structures; however, the optimization of Bayesian network modelling requires learning the two constituents – network structures and conditional probability tables (CPTs). This implies that he does not guarantee optimal memory optimization. Due to the reliability required of any intelligent system models such as Bayesian networks, a trade-off between the two resources may not be ideal for real-life problems of emerging the models from massive datasets. This implies that more approaches will be required to optimize both the BN learning time and the memory usage.

2.4.4 Distributed Learning on Networked Machines

Aside from the strides that have been made, distributed learning approaches on powerful networked machines or environments have been proposed, but have issues of *communication failures* and *load balancing* [47] [48]. The issues arise often as a result of limited bandwidth of networks, especially in developing countries. These issues significantly affect the response time of such distributed learning whose success is determined by the following factors: (i) *communication* time, (ii) *knowledge integration* and (iii) the *learning components* [47]. The first factor is the duration to transport communication parameters, mobile agents and results between remote locations (workstations) and the server. Additional duration is required in the second factor of knowledge integration to combine all results received from workstations. The core factor that has not received adequate attention is the learning components, which require time for learning the models. For instance, a client workstation that receives processing loads

from a server is susceptible to memory failure, if most of the available memory slots are fully occupied or the processing takes too long to complete.

To address the two issues of failure and balancing, Krishnaswamy et al. [47] proposed a rough set technique to estimate the response time by identifying and computing costs for alternative strategies, and to choose a least cost strategy. Achieving these is a difficult task, and setting up these suitable networking environments, especially in under-developed countries, is obviously not cheap. This is a strong motivation as the efforts made in this thesis are partly directed at developing an alternative economical optimization approach that can solve the intensity problems, particularly on the popular single-user machines.

2.5 Comparing the DBN Models and Computational Intensity Addressing Approaches

Based on the review above and on our practical experience, Tables 2.2 and 2.3 summarise the comparisons for the characteristics, and our inference on the existing DBN model representations and the approaches addressing computational intensity problems respectively.

Table 2.2: Comparing the DBN Model Representations

DBN Model Representations	Network Structures (over time)	Probability Distributions (over time)	Model Complexity	Expressive Power	Our Inference
HMM	Remains fixed	Variability not always true	Not difficult to represent	Very restricted with assumptions	For skilled users
Auto-Regressive	Remains fixed	Variability not always true	Not difficult to represent	Very restricted with assumptions	For skilled users
Input-Output	Remains fixed	Variability not always true	Not difficult to represent	Restricted with assumptions	For skilled users
Factorial	Remains fixed	Variability not always true	Challenging to represent	Restricted with assumptions	For skilled users
Coupled	Remains fixed	Variability not always true	Challenging to represent	Restricted with assumptions	For highly skilled users
Mixture of Gaussian Outputs	Remains fixed	Varies using mathematical functions	Challenging to represent	Restricted with assumptions	For highly skilled users
Hierarchical	Remains fixed	Variability not always true	Challenging to represent	Restricted with assumptions	For highly skilled users
PDBN	Remains fixed	Varies	Challenging to represent	Restricted with assumptions	For skilled users
BBDBN	Remains fixed	Varies	Not difficult to represent	Very restricted with assumptions	For skilled users
SDBN	Remains fixed	Varies	Challenging to represent	Restricted with assumptions	For skilled users

Observe in Table 2.2 that most of the models somehow vary their probability distributions (or CPTs, quantitative knowledge) but make their network structures (qualitative knowledge) remain constant over time for convenience. However, a DBN model is meant to vary its knowledge over time [3] [8]. This is obviously an explicit representation or approximation which limits the expressive power of DBNs and this can easily be handled by experts or skilled users. This shows that finding a solution to an uneasy task often advances scientific knowledge.

Observe in Table 2.3 that discretization, ADtree and incremental learning methods have fairly positive results on moderate datasets but have challenges of exponential time and out-of-memory problems on massive datasets. Distributed learning on networked machines can handle massive datasets but have challenges of load balancing and communication failures. Therefore, this poses an open question: how do popular single-user machines learn BN models from massive datasets?

Table 2.3: Comparing the Computational Intensity Addressing Methods – Scalability

Methods	**Execution Speed**	**Memory Management**	**Cost**	**Our Inference**
Data Discretization	Fast on small up to moderate datasets	Good on small up to moderate datasets	Affordable	Exponential time and out of memory problem on massive datasets
ADtree Method	Fast on small up to moderate datasets	Good on small up to moderate datasets	Affordable	Exponential time and out of memory problem on massive datasets with too many states
Incremental Learning	Speed not significant with respect to batch approaches	fairly ok	Affordable	Trade-off issues
Distributed Learning on Networked Machines	fast	Good	Expensive to set up	Issues with load balancing and communication failures

2.6 Conclusions

In this Chapter we have critically reviewed what researchers have done so far in DBN models, described why Bayesian learning research cannot be separated from DBNs, and presented the suggested solutions to address the ongoing challenges of learning. Summarised results are shown in Tables 2.2 and 2.3.

Our analysis, illustrative descriptions and comparisons revealed that most of the existing DBN models are based on HMM representations but the full capabilities of DBNs have not been exploited. For instance, complex environments such as business sales, environmental water or oil pollution, etc. capture massive heterogeneous datasets, and uncertainties about meaningful knowledge are embedded therein. A sophisticated modelling approach is required for capturing with individual preferences which are ever changing. If only one DBN component (probability theory) is allowed to change while the other (DAG) remains fixed, we conjecture that the modelling method will approximate and struggle to meet up with the ever-changing complex environments. This may therefore lead to some loss of costly information. Moreover, as presented earlier, the varieties of the DBNs have obviously opened up challenging problems, especially for non-expert practitioners, of how to choose the most suitable model for specific real-life applications.

It is our opinion that most of these existing DBN frameworks still have limited expressive power in spite of expert intervention which may introduce biases; possibly, they have been avoiding computational complexity for the sake of convenience. One of the major potentials of this thesis is to advance DBN modelling to accommodate all users (both skilled and non-expert practitioners) and find an alternative economical solution to the ongoing learning problems which may affect DBN in future.

Chapter Three

DBN: Theoretical Backgrounds

3.1 Introduction

A dynamic Bayesian network (DBN) model is suitable for a modelling environment that changes over time, and it has the capability of predicting future behaviour of the environment [8]. It has popularly been used in areas such as speech recognition, traffic monitoring, audio-visual speaker detection, etc. [1] [3] [7]. Any DBN observes the first-order Markov model which states that future event V_{t+1} is independent of the past, given the present V_t [8]. The transition probability which describes how an event changes from one state to another is represented as $Pr(V_{t+1} \mid V_t)$. Some DBN models have been developed with explicit representation of hidden Markov models (HMM) such as [1] [7]. The simplest form of DBN that is often used as a baseline of comparisons is shown in Figure 3.1.

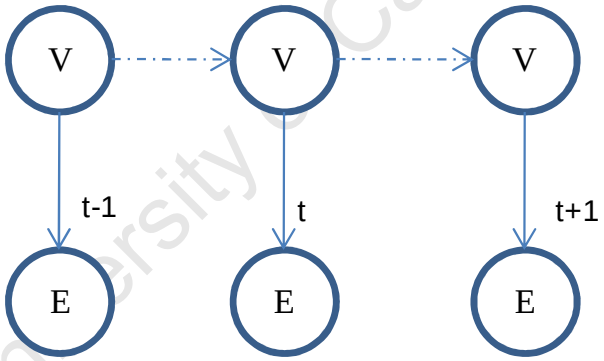


Figure 3.1: The simplest DBN (HMM) with V as state variables and E as evidence variables repeated in three time steps.

The states of events in huge explicit representations of DBNs have complex interaction due to the time dependency and the states may impact on the observed variables of the DBN at every time step. Explicit modelling of DBNs by experts in complex environments often become complicated and may not always capture hidden information embedded therein. This contributes to information gap problems between users and the environment which often lead to incorrect monitoring and predictions. This is one of the challenges for intelligent systems engineers in modelling and presenting DBNs. Automating DBN learning directly and completely from real-life professional environments (or datasets) in the absence of domain experts remains an open challenge. In this chapter, we describe the theoretical backgrounds of

monitoring and predicting the most likely hidden outcomes of the present and the future which are not known, based on the current and past patterns we understand. We therefore define three well-researched niche areas of DBNs as follows: (1) DBN whose probabilistic distribution and the DAG do not vary but the representation of the probabilistic process changes. This is otherwise referred to as stationary dynamics, and most DBNs based on HMM with expert representations fall into this class; (2) DBN that varies its probabilistic distribution but its DAG remains fixed; and (3) DBN that varies its probabilistic distribution and may vary its temporal DAGs.

The third class is the most challenging niche area that has not received adequate attention, because researchers have been approximating their models for subjective convenience. The complete emergence of the third class of DBNs motivates the principles of situation awareness to reveal and predict hidden patterns from complex environments over time. This implies a temporality link between the DBN and the theoretical principles underpinning the creation of awareness about unknown or risk situations in order to do anticipatory planning. The major contributions of this chapter are as follows:

- A precise graphical process of translating theory to illustrate BN learning, which gives insight into fundamental knowledge of learning DBNs.

- A new ambitious way (e.g. mathematical equations of situation calculus) of marrying or revealing the temporality links between the DBN backgrounds and the theoretical principles underpinning Situation Awareness.

- An approach to avoiding being abstract and presenting necessary theories relevantly for developing algorithms to learn BN models from datasets.

The rest of this chapter is arranged as follows: in section 3.2, we present the background of DBN in relation to its basic modeling requirements. In section 3.3, we describe the background of situation awareness and the phenomenon of situation calculus as the theoretical concepts we are using in order to achieve our new class of Dynamic Bayesian Networks. The fundamental building blocks of DBNs are described in the ordinary Bayesian networks presented in section 3.4. It includes the probability theory, the requirements for capturing the complexities of environments, the general description of Bayesian learning algorithms, the discretization algorithms, and the rudiments of Bayesian inference algorithms that revolve around Bayes' theorem. In section 3.5, we present examples of models and theories that can be used to develop a Bayesian learning algorithm. This includes minimum description length (MDL), mutual information (MI), power set theory, etc. Section 3.6 concludes the chapter.

3.2 Dynamic Bayesian Network Models

Dynamic Bayesian networks are often presented as temporal probabilistic models, often referred to as an extension of the Bayesian network (BN) models in artificial intelligence [3] [8]. A Bayesian belief network is formally defined as a directed acyclic graph (DAG) represented as G = {X(G), A(G)}, where X(G) = $\{X_1,\ldots,X_n\}$, vertices (variables) of the graph G and $A(G) \subseteq X(G) \times X(G)$, set of arcs of G. The network requires discrete random values such that if there exist random variables $X_1, \ldots, X_n$ with each having a set of some values $x_1, \ldots, x_n$, then their joint probability density distribution is defined in equation (3.1):

$$pr(X_1,\ldots,X_n) = \prod_{i=0}^{n} pr(X_i \mid \pi(X_i)) \tag{3.1}$$

where $\pi(X_i)$ represents a set of probabilistic parent(s) of child X_i [37]. A parent variable, otherwise referred to as *cause*, has a dependency with a child variable known as *effect*. Every variable X with a combination of parent(s) values on the graph G captures probabilistic knowledge (distribution) as a conditional probability table (CPT). A variable without a parent encodes a marginal probability. A BN can be modelled by eliciting the probabilistic knowledge from domain experts, if the environment is small. For more complex domains, the most suitable Bayesian networks are learnt from the environments captured as datasets, as proposed in [23] [25] [38] and [39], as they have presented many algorithms for learning Bayesian networks from datasets. The BNs' characteristics for capturing dependencies of variables make them suitable for handling complex problems [37].

However, the inability of BNs to capture time as temporal dependencies gave rise to the development of various ways of modelling Bayesian networks dynamically. The variables and the CPTs of the BNs are similar to the states and the probabilities used in the temporal dependencies of the DBNs. Since DBNs handle complex situations of multiple dependent events of the Markov model over time, researchers [3] [8] sometimes present the following three parameters required to construct a DBN model: prior matrix, $Pr(V_0)$; transition matrix, $Pr(V_t \mid V_{t-1})$; and sensor matrix, $Pr(E_t \mid V_t)$. The prior matrix defines the initial probability distribution of states V_0 at the start of evolving the DBNs. The transition matrix describes time dependencies for the transitions of DBN states V for all t. Also, the sensor matrix captures the probabilistic distributions from the relationships of observation variables E at any time step t. The three matrices can be estimated during the intra-frame (slice) and inter-frame learning of a DBN model using parameter-learning algorithms such as maximum likelihood estimate (MLE) and expected maximization (EM) [35] [49]. The intra-frame learning estimates the conditional probability tables or

distributions captured as CPTs for every time step *t*, while the inter-frame learns the CPTs across the time steps. In conjunction with the DBN matrices, equation (3.2) shows the combined joint probability distribution for any temporal model up to a finite time *t*.

$$\Pr(V_0, V_1, ..., V_t, E_1, ..., E_t) = \Pr(V_0) \prod_{i=1}^{t} \Pr(V_i \mid V_{i-1}) \Pr(E_i \mid V_i) \tag{3.2}$$

3.3 Situation Awareness and Calculus

This section presents the theoretical principles underpinning the development of situational awareness technologies. This includes the theoretical background of situation awareness with its situational calculus.

3.3.1 The Theory of Situation Awareness (SA)

Hidden situations often evolve from emergent patterns embedded in the real-world environment on an ongoing basis. In order to model the environment and its emergent situations, and evolving temporal patterns, a current situation model must capture explanations from real-world events in such a way that it facilitates understanding in SA.

The most established theory of SA is described in Endsley's model [16], which describes the current situation model in a mental model at three hierarchical levels. As shown in Figure 3.2, Levels 1, 2 and 3 of SA correspond to *perception*, *comprehension* and *projection* respectively. The three components are the perception of elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future.

Perception is the most fundamental level in the current situation model, which involves the monitoring of the environment. Comprehension involves the understanding of how information perceived is integrated to create knowledge. Projection of events in the future, based on perception and comprehension, is the highest level of the current situation model. This is the anticipation of likely future events. Thus, SA enables us to recognize what pattern is developing in our domain of interest in order to figure out what to do next. This is a decision-making process that necessitates the mental model described in Figure 3.3.

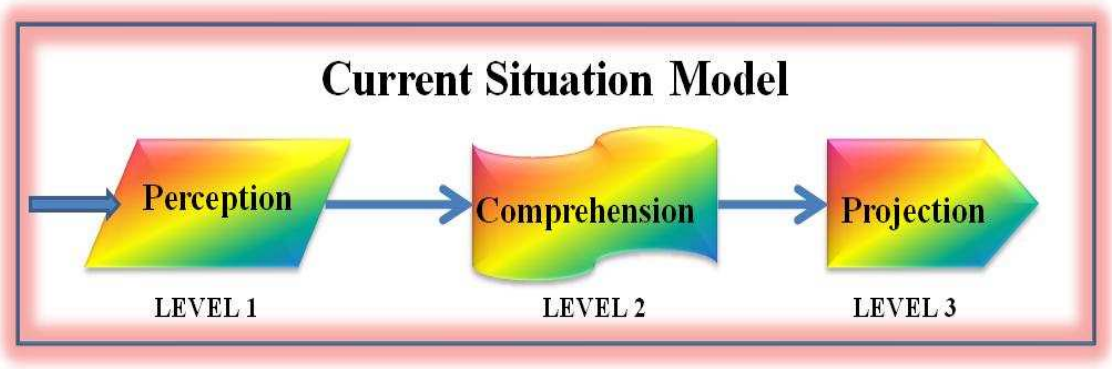


Figure 3.2: A Hierarchy From The Endsley SA Model [16]

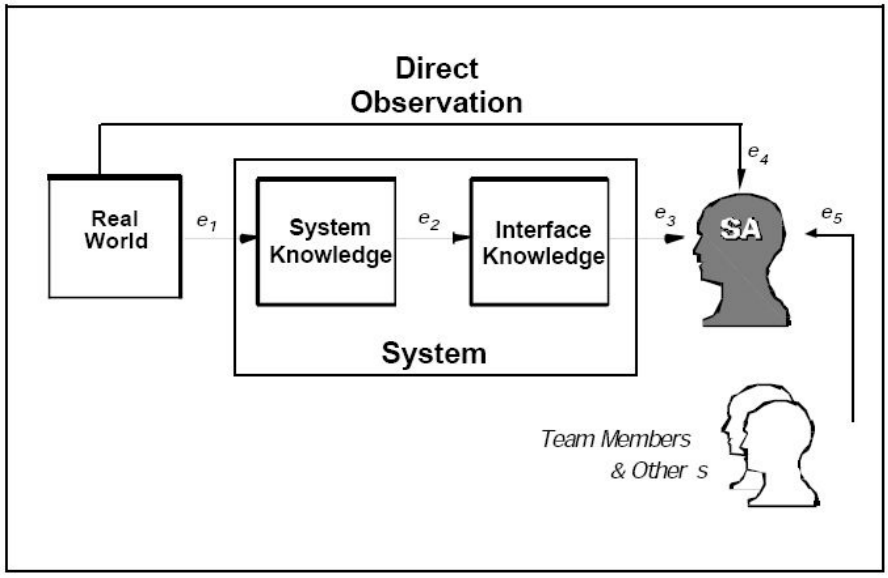


Figure 3.3: Endsley's Mental Model [16]

As shown in Figure 3.3, the major sources of information for SA as a mental model are from *system* and *interface knowledge* and may probably be affected with direct *observation* from the real world and other *factors* [16]. The e_3 is the *active* SA component obtained from the integration of system and interface knowledge. It is the crucial aspect of SA that is often prone to errors because its performance is dependent on the methodology used by the SA engineer as presented in [16]. Direct observation from real world, e_4, and distant observer, e_5, are other sources of information that may probably affect output of SA and consequently have an impact on decision-making. In some techniques, decision-makers may directly view SA results as being interactive or act over a network. However, it is understood from [16] that as a

state of knowledge, the patterns or results of SA are expected to have the capability of assisting with answers to the following four 'W' questions about any situation: (1) What is happening? (2) Why is it happening? (3) What will happen next? (4) What can I do about it?

Logic formalism or calculus is commonly used to express or elaborate on situational awareness as described in the next subsection.

3.3.2 The Phenomenon of Situation Calculus

Situation Calculus is a logic formalism, which expresses the inner awareness of a decision-maker (e.g. managers, soccer players, robots, agents, etc). Situations are states of knowledge resulting from executing actions [8]. This idea is illustrated in Figure 3.4.

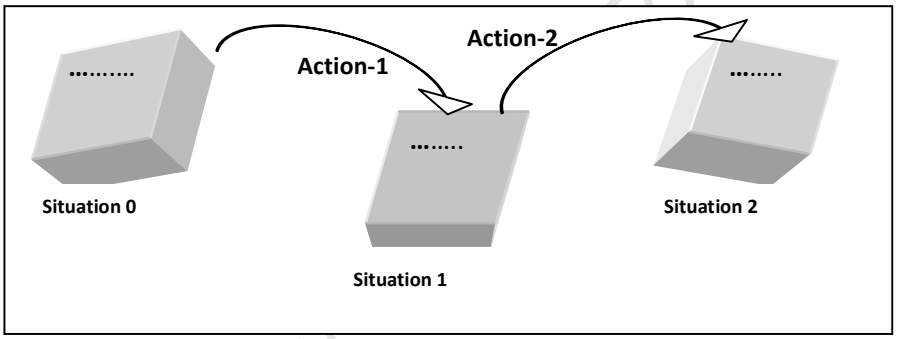


Figure 3.4: Situations Resulted From Execution of Actions

It is obvious in Figure 3.4 that the situations vary depending on the actions executed at any time step in the domain of interest. Specific example is environmental factors (e.g. climate change or weather) which significantly affect sales of products over a period of months. In new DBN technologies, BNs are dynamically evolved over time steps as temporal models of the situations. However, we logically derived mathematical equation (3.3) from Russell's description of situation calculus [8] and express it as follows:

$$S_0 = \varphi(\delta_k, S_0)$$
$$S_1 = \varphi(\delta_{k+1}, S_0)$$
$$S_2 = \varphi(\delta_{k+2}, \varphi(\delta_{k+1}, S_0)),$$

By conjecture,

$$S_i = \varphi(\delta_{k+i}, S_{i-1}) \quad (3.3)$$

for all values of k = 0, i = 0,1,2,…

where δ_k is an empty action and the rest of the actions are not empty. φ is a function and predicate, which may change as a situation changes with time. The descriptions of actions used in logic reasoning are preconditioned and are explicitly represented as axioms. These are collectively integrated as knowledge. Examples from Russell include *Possibility* (Poss ()) and *Effect* axioms. The situation calculus of the axioms is described and applied to an agent example that moves gold between two locations as follows: the possibility axioms of a world state that an agent can go between adjacent locations, grab a piece of gold in the current location, and release some gold that it is holding [8]:

$$At(Agent, x, s) \ \Lambda \ Adjacent(x, y) \Rightarrow Poss(Go(x, y), s).$$

$$Gold(g) \ \Lambda \ At(Agent, x, s) \ \Lambda \ At(g, x, s) \Rightarrow Poss(Grab(g), s).$$

$$Holding(g, s) \Rightarrow Poss(Release(g),s).$$

The effect axioms state that, going from x to y results in being at y, grabbing the gold results in holding the gold, and releasing the gold results in not holding it [8]:

$$Poss(Go(x, y), s) \Rightarrow At(Agent, y, Result(Go(x, y), s)).$$

$$Poss(Grab(g), s) \Rightarrow Holding(g, Result(Grab(g), s)).$$

$$Poss(Release(g),s) \Rightarrow \neg Holding(g, Result(Release(g), s)).$$

It may become laborious to think of all possible axioms in real-world situations. In [8], it is added that this logic formalism becomes awkward when actions are numerous. This exemplifies a frame problem which can be addressed by self-awareness. This means that if a system is shown an action to be taken at a point of decision, the system will work only at that point. It implies that it is better to train a system on how to determine appropriate actions to be taken at any point in time so that it can work for ever. Thus, the approach of our DBN technology tends to minimize the worries of users for knowing the actions to be taken on some situations as these would be understandable when evolving the local dynamics from the global behaviour over time. The next section illustrates Bayesian networks as necessary and significant steps to modelling the DBNs.

3.4 Building Blocks of Bayesian Networks

This section presents further details with regard to BNs being the building blocks of DBNs. The details include the fundamentals of probability theory used in capturing the quantitative knowledge (CPTs) of BN, the capabilities of BNs to capture complexities in environment, the BN learning and discretization algorithms for capturing qualitative (network) knowledge of BNs, and the Bayesian inference theorems for reasoning.

For the purposes of illustrating BN, Figure 3.5 shows the DAG and the CPTs of a Bayesian Network model as the core reasoning component of an intelligent system. In this case, it describes the operation of a block-lifting machine. The operation is monitored with the following attributes: battery (B), movement (M), liftable (L) and gauge (G) [37]. Each of the attributes contains states true (t) and false (f), with their respective probabilities captured as CPTs, such as L having 0.75 and 0.25. Figure 3.5 depicts conditional dependencies of the attributes which best describe the complexity of variables and the computation of the CPTs of the block-lifting machine. The estimation of the probabilities captured as CPTs is illustrated in equations (3.7) – (3.15) using the MLE (maximum likelihood estimate) algorithm. Learning the suitable networks from massive datasets is computationally intensive, as stated earlier. For BN learning, the important fundamentals required in capturing qualitative and quantitative knowledge are introduced in the next subsection.

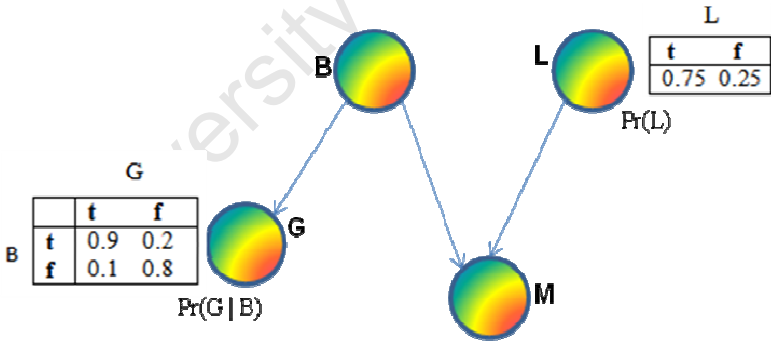


Figure 3.5: A Simple Bayesian Network Model of an Intelligent System for a Block-Lifting Machine [37].

3.4.1 Fundamentals of Probability Theory and Kolmogorov's Axioms

As presented above, a Bayesian network represents a set of random variables and their causal interrelationships in a given problem domain. A Bayesian network requires discrete random values such that if there exist random variables $X_1, \ldots, X_n$ with corresponding values $x_1, \ldots, x_n$, then their joint probability is also given by the expression [34]:

$$\Pr(X_1 = x_1, \ \ldots, X_n = x_n)$$

The important properties of discrete probability distributions are as follows [34]:

$$0 \le \Pr(X_i) \le 1, \text{ where } i = 1,2,\ldots, n$$

$$\sum_{i=1}^{n} \Pr(X_i) = 1 \tag{3.4}$$

Similarly in basic probability calculus, for any arbitrary universal set U, the following Kolmogorov's axioms hold [34]:

$$\Pr(U) = 1,$$

$$\Pr(X) \ge 0, \forall X \subseteq U ,$$

$$\text{If } X \cap Y = \phi, \text{ then } \Pr(X \vee Y) = \Pr(X) + \Pr(Y), \qquad \forall \quad X, Y \subseteq U \tag{3.5}$$

These basic properties are observed in Bayesian networks. One can see Pr(L) in Figure 3.5 which captures the CPT of its two states *true* and *false* as 0.75 and 0.25 respectively. The efficiency of reasoning under uncertainty with Bayesian networks is as a result of their (in)dependence characteristics described in the next subsection, which are capable of capturing the complexities in any environment.

3.4.2 Capturing Complexity of Systems with Bayesian Networks

It is believed that complexity is embedded in a system or environment if a percentage of a dependent variable *Y* has effects on a set of its independent variables $\{X_1, X_2,\ldots, X_n\}$ [28]. The environment becomes more complex if there are many such variables of Y. The following (in)dependence characteristics are instrumental of capturing complexities embedded in the environments by Bayesian networks. For better understanding, a sample Bayesian network shown in Figure 3.6 describes the characteristics.

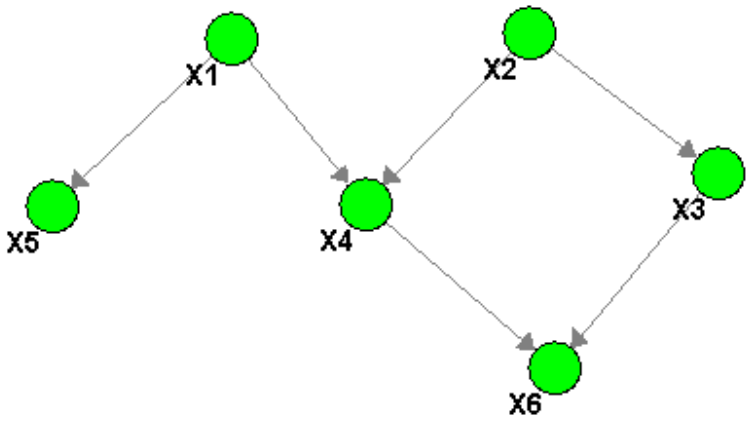


Figure 3.6: A Sample Bayesian Network

(i) **Conditional Dependence**: In Figure 3.6, X_5 is conditionally dependent on X_1; we denote and compute the probability as: $Pr(X_5 \mid X_1)$. More elaborately,

- X_4 is conditionally dependent on X_1 and X_2; we denote and compute the probability as: $Pr(X_4 \mid X_1, X_2)$,
- X_3 is conditionally dependent on X_2; we denote and compute the probability as: $Pr(X_3 \mid X_2)$,
- X_6 is conditionally dependent on X_4 and X_3; We denote and compute the probability as: $Pr(X_6 \mid X_4, X_3)$,

(ii) **Conditional Independence**: An evidence is a value that is assumed to be seen or observed with certainty in real life. A node is said to be non-empty if it contains value. If X_4 is a non-empty (evidence) node then X_6 is conditionally independent of X_1 and X_2. In other words, this means X_6 does not depend on X_1 and X_2 but only depends directly on X_4. Also, the only path from X_5 to X_4 is blocked by node X_1 hence, we say that X_5 is *d-separated* (conditionally independent) from X_4 [37]. Using a literal meaning for example, two nodes *A* and *C* are said to be d-separated if there is a node *B* between them. Using Bayes' rule (see subsection 3.3.5), we say that $Pr(X_5 \mid X_1)$ is conditionally independent of $Pr(X_4 \mid X_1, X_2)$. In fact, this is an important property that the belief network exploits to reduce the number of probabilities to be computed as a model which makes intractable problems feasible. That is, the theory of conditional independence makes a child node consider only its direct parent nodes when computing probabilities.

(iii) **Unconditional Independence**: A node is said to be empty if it does not contain a value or an evidence to propagate into other nodes. In Figure 3.6, suppose X_2 is empty, then we say that X_3 is unconditionally independent of X_4. If X_2 does not have an evidence to propagate into X_4, this probably implies that X_3 also does not propagate significant value into X_2. That is, by Bayes' theorem, we can also say $Pr(X_4 \mid X_1)$ is unconditionally independent of $Pr(X_3)$.

(iv)**Joint Probability**: A query node is a target node that we want to infer or whose probability we want to know. The joint probability of a model depends on what is expected in a query node given certain evidence. With reference to equation (3.1), the joint probability distribution for the BN in Figure 3.6 is shown in equation (3.6).

$$Pr(X1, X2, X3, X4, X5, X6) = Pr(X1)\,Pr(X2)\,Pr(X3 \mid X2)\,Pr(X4 \mid X1, X2)\,Pr(X5 \mid X1)\,Pr(X6 \mid X3, X4) \quad (3.6)$$

(v) **Conditional Probability Table** (**CPT**): This is a form of table that captures the conditional probability distributions for discrete variables. Each row in a CPT contains the conditional probability of each node value for all the possible combinations of values for its parents. Each column must sum up to one [8]. Examples of CPTs have been presented in Figure 3.5, such as Pr(G | B). However, to minimize the computational intensity or nondeterministic polynomial time (NP-hard) and to accelerate the induction processes of learning both network structures and their associated CPTs, the principles around discretization algorithms are essential and are presented in the next subsection.

3.4.3 Discretization Algorithms

Discretization algorithms are techniques which are used as preprocessing key operations in learning Bayesian models. They classify numerical data into their corresponding interval values relative to the patterns in the data attributes. The available open source Bayesian learning applications, such as Weka and Hugin, use discretization algorithms which are built around the simple binning and minimum description length (MDL) methods.

Simple binning employs an equal-width method using an unsupervised discretization approach which divides attribute values into *k* equal sizes. The seed *k* is supplied by users, equal-width finds maximum and minimum attribute values and these are used to determine data intervals. Simple binning also employs equal-frequency based on another unsupervised discretization approach, which divides the data *m* into *k* bins where each bin contains *m/k* adjacent values. In entropy-based discretization,

distribution of class labels can have three basic shapes which are [42]: (i) big intervals each containing the same class of points, (ii) big intervals but not all of them containing a same class of points, and (iii) a class of points randomly mixed over a range. For the first shape, if the middle point between the two classes is partitioned into two intervals, the entropy will be zero. In the second shape, there are more partitions of right intervals that are intended to be attained than left intervals. In the last shape, entropy-based discretization ignores the features as mixed points over a range.

The Hellinger-based algorithm, however, uses interval entropy function E(.) as a justification for quality discretization to accommodate any datasets. The entropy of any interval between *a* and *b* is shown in equation (3.7) and is similarly called the entropy of the midpoint [43].

$$\mathrm{E}([a, b]) \equiv \sqrt{\left|\sum_{i}\left(\sqrt{pr(x_i)}-\sqrt{pr(x_i \mid \overline{ab})}\right)^2\right|} \tag{3.7}$$

As a basis of the algorithm, the values x_i of the target attribute being discretized are sorted accordingly and they form a column of intervals. The probability distribution of x_i is represented as $pr(x_i)$. Equation (3.7) therefore computes the entropy of every interval *a* and *b* from the attribute. The minimum set of entropy values as required by users are selected for their corresponding discretized interval values. It was guaranteed [43] that the proof and the qualitative results of the algorithm showed the accuracy of the discretization. These discretization algorithms support optimization of the network models as the Hellinger-based algorithm is adopted as a proof of concept for discretizer agent's actuators in our EDBN architecture. However, any of the conventional algorithms can also be used, as one of the objectives of this research is on scalability of discretizations. It prevents out-of-memory problems during learning and makes Bayesian networks reliably available for reasoning in intelligent systems.

3.4.4 Bayesian Learning Algorithms

This subsection describes general and fundamental principles of learning algorithms research. The objective of every BN structure search (or learning) algorithm is to keep finding edges and nodes that will give the best network in a reasonable finite time. New nodes are not created but extracted only once, directly from the attributes of a learning dataset. The learning process of the algorithms is a combinatorial analysis but the objective is intended to minimize search time and space usage. We present a simple combinatorial analysis in Figure 3.7, which illustrates the complexity of network learning problems. It is a combinatorial problem because the use of the movement operators, such as arc reversals and arc deletions, can continue indefinitely. Arc reversal is changing the direction of an arc between two nodes

while arc deletion is the removal of an arc between two nodes. Observe that there are infinitely many Directed Acyclic Graph (DAG) solutions in β, if β = {Moveable, Gauge, Liftable, Battery} in Figure 3.7. The general outline which most learning algorithms therefore use as a basis of their approaches are as follows:

- Carry out a series of arc changes one at a time.
- Check whether the resulting graph is a valid network structure.
- Measure the network scores before and after every arc change.
- Replace the old structure with the new one depending on the difference between the scores.

Popular examples of various search techniques to learn network structures are heuristic-search, hill-climbing, and genetic algorithms [8] [49]. According to Russell et al. [8], a heuristic is an educated guess of learning for solving complex problems or making decisions. A heuristic-search technique is a search strategy that is guaranteed to find a good enough model in a reasonable time, but which may not always find an optimal or universal model from a dataset.

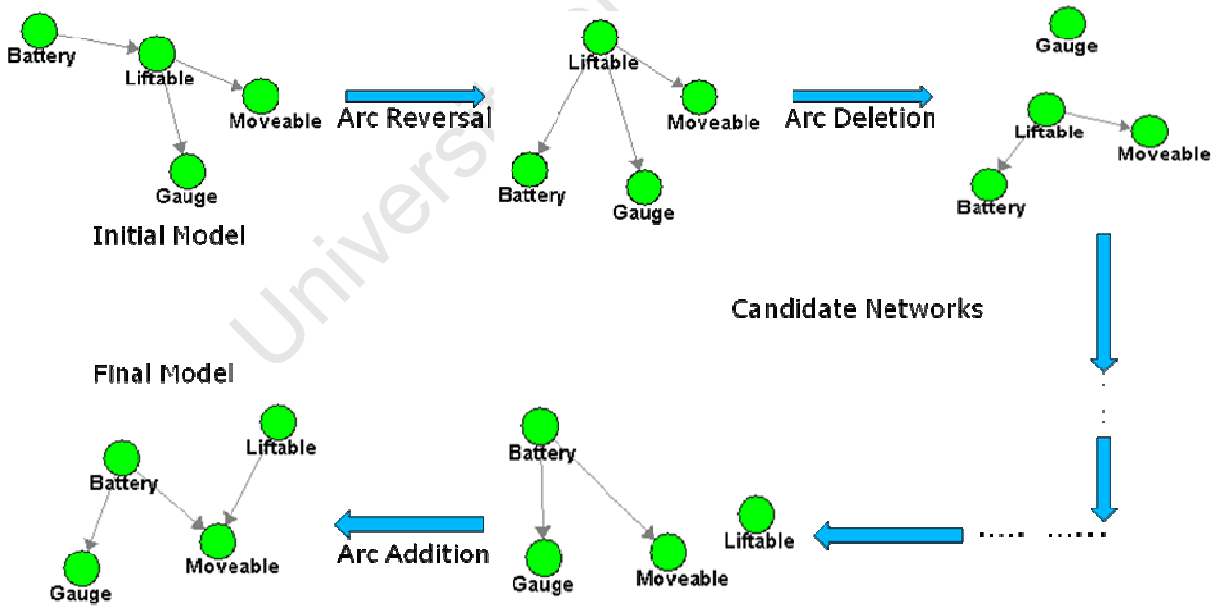


Figure 3.7: Combinatorial Analysis Problem in the Network Learning Space, β

The hill-climbing algorithms are greedy score-based algorithms [8] [46]. A greedy search is a best-first search in which the most promising attribute is chosen for expansion according to some specified rules.

The hill-climbing search algorithm continuously moves uphill and may be terminated when there is no further learning improvement in the specific network structures. It is useful to study the details of hill-climbing because most advanced learning and optimization techniques take it as a baseline of comparison due to its simplicity. Hill-climbing is sometimes popularly referred to as the K2 algorithm and it has common characteristics, as follows [8] [37] [46]: it learns structures fast; it does not maintain a search tree or space (no memory); it keeps only the current state (candidate network) and the objective function value (network goal). The network goal depends on the design, its implementation and is commonly reached by convergence; and it does not look ahead beyond the immediate neighbour of the current state. The commonly used learning operators are: addition, removal and reversal of arcs [20] [37]. Hill-climbing movement set design is critical because there is no universal pattern of choosing the operators at each learning step. Also, if the number of moves with the operators is enormous (e.g. in large datasets), the algorithm may be inefficient. Thus, the choice of learning operators and the details of the algorithms differ from one intelligent system researcher to the other, as presented in section 2.4. Despite the fact that hill-climbing is fast, it also gives rise to a further research question, viz: when does the learning process (choice of operators) stop to avoid the process getting stuck or converge at a local minimum? A local minimum is an intermediate solution point for an iterative process.

A Genetic Algorithm (GA) is an evolutionary optimization algorithm that solves problems by using the process of evolution [50]. Russell et al. [8] describe a GA as a stochastic search algorithm in which a large population of states is maintained. New states are generated by mutation and crossover processes, which combine pairs of states from the population. A GA elicits a natural selection whereby offspring populate the next generation according to the fitness of the parents. The genetic algorithm we developed and enhanced in [51] is integrated into the EDBN architecture proposed in this thesis as a proof of concept due to its efficiency. The characteristics of a GA are as follows: it maintains a solution space or search tree (e.g. generated populations), it is computationally intensive during optimization but guarantees optimal solution. Also, bit-string representation is critical in GA, if an application requires bits [23]. A bit is either a 1 or a 0. A bit-string is a sequence of 0's or 1's. An example of a bit-string is "10011". An application that requires bits is more prone to errors that can arise from bit-string manipulation. For the purposes of the current research, we are particularly interested in exploring GAs for the following reason: its smart characteristics can save a great deal of programming efforts. That is, the implementation of crossover, mutation and fitness function is clear; and it has wider academic applicability: furthermore, positive testimonies exist from early GA researchers. Moreover, Larranaga et al. [23] and Russell et al. [8] described the GA as one of the best optimization algorithms and as being highly versatile and useful for solving optimization problems, specifically learning BN models from data. Hence, we deduce from the above premises that a GA can reveal a true representation of underlying

models in datasets more effectively than the greedy search methods can. The following are fundamental definitions of Genetic Algorithm terminologies:

- A **Population** P is a set of individuals. For instance, if P = {{x_1}, {x_2, x_5}, {x_6}, . . . {x_n}}, then P denotes a population.

- An **Individual** is represented as a string over a finite alphabet. For instance, {x_1}, {x_2, x_5} in P are individuals. In other words, x_2 and x_5 are alphabets or strings that are finite within the individual {x_2, x_5}. A form of string that is not an alphabet is a bit-string that was defined in the last paragraph.

- The **Fitness Function** is used to evaluate the fitness of individuals in candidate selection, that is, the capability of the individuals to survive the generation. For instance, the Minimum Description Length (MDL) is an example of a fitness function. More description is provided in section 3.4.4.

- The **Objective Function** tries to find an optimal value for a fitness function. For instance, the objective function is to find the minimum network score, when learning BNs from data and when the MDL is used as the fitness function.

- **Candidate Selection** is the choice of individuals in the population that are fit to achieve the objective of a problem definition. For instance, x_1 may be more probable to be the parent of x_2 while x_2 may be less probable to be the parent of x_1. This is determined by the fitness function or related measures applied to the models.

- A **Gene** is an entity or a subset in an individual. For instance, the orders or directions of $\overrightarrow{x_1 x_2}$ and $\overrightarrow{x_2 x_1}$ are separate genes of the individual {x_1, x_2}. In this research, this means that the measurement of dependency of x_2 on x_1 is different from the measurement of dependency when the mutation gives x_1 depending on x_2.

- **Reproduction** is the generation of offspring from parents. The common genetic operators such as *crossover* and *mutation* are used to carry out the reproduction process from one generation to another. Reproduction multiplies the population and therefore creates a solution space.

- **Crossover** is generally considered as the most important genetic operation. This refers to the process of creating a new individual (offspring) through the combination of genetic materials of two parents. For instance, let $\{x_1\}$ and $\{x_2\}$ be parents in a network of population, then, through crossover, they reproduce a new individual $\{x_1, x_2\}$.

- **Mutation** introduces new genetic material into an existing individual. Introduction of new genetic material in structure learning refers to change of order or direction of the gene (variables) (see gene description above). For instance, we can mutate x_1,x_2 to become x_2,x_1. In biological terms, a chemical process may act on genetic material and generate new genetic material. However, it will maintain the same structure but different chemical constituents.

Having understood the theories of learning algorithms for achieving suitable network structures, their CPTs are simply learned from the datasets, often with maximum likelihood estimate (MLE) or expected maximization (EM) algorithms [8] [35] [49]. For the relevance of this DBN research, we described and implemented the MLE because the EM is slower due to its time-consuming iterative processes (or slow convergence) and prone to local optima [25]. However, as a result of too many states, finding best network structures and computation of huge CPTs often add to the computational intensity of Bayesian learning. We found out that the computation of huge CPTs consumes more time than learning the network structures. Fortunately, an economical scalable solution is provided in our EDBN architecture. This is, however, outside the scope of this chapter.

The theory of MLE and Parameter Learning [8] [35] is a reasonable approach for estimating probabilities for the parameters of a learned BN structure, especially when a training dataset is large. It is a good estimation of Bayesian network probabilities because it becomes difficult to prefer one parameter over another (biased) and, thus, uniform priors are assumed over all parameters. This is an unbiased estimate because it allows estimation to be driven by the dataset. Supposing there are two variables x and y in a dataset with values that sum to N. If the parameter of x represents the proportion or the probability as γ, then the parameter of $y = 1 - \gamma$. Therefore, the likelihood of the dataset d with the required hypothesis h is given as follows [8]:

$$Pr(d \mid h_\gamma) = \gamma^x (1 - \gamma)^y \tag{3.8}$$

The likelihood of h is given by the value of γ that maximizes the log likelihood:

$$L(d \mid h_\gamma) = \log \Pr(d \mid h_\gamma) = \log (\gamma^x (1 - \gamma)^y) \Rightarrow x \log \gamma + y \log (1 - \gamma) \quad (3.9)$$

For the next equations (3.10) to (3.12), differentiate L w.r.t γ, and equate to zero to obtain the maximum likelihood estimate of γ. Since the differential of

$$\log \gamma = \frac{1}{\gamma} \quad (3.10)$$

then,

$$\Rightarrow \frac{dL(d \mid h_\gamma)}{d\gamma} = \frac{x}{\gamma} - \frac{y}{1-\gamma} = 0 \quad (3.11)$$

Therefore,

$$\gamma = \frac{x}{y+x} = \frac{x}{N} \quad (3.12)$$

This is the deduction of the instance counts which are sufficient statistics discussed by [35]. Hence, this formulates the basis of the quantitative knowledge of Bayesian learning that estimates the CPTs associated with each node of Bayesian networks. The CPTs are estimated from the instances of training datasets. For instance in Figure 3.5, the probabilities in each CPT are calculated such that all the node states are estimated. For example, the knowledge of node L is estimated as marginal probabilities. Since L has two states *t* and *f*, then the probabilities of Pr(L) are calculated in equations (3.13) and (3.14).

$$\Pr(L = t) = \frac{\#(Ins\tan ces\_of\_L = t)}{Total\_ins\tan ces\_of\_L} \quad (3.13)$$

$$\Pr(L = f) = \frac{\#(Ins\tan ces\_of\_L = f)}{Total\_ins\tan ces\_of\_L} \quad (3.14)$$

These results are captured in the (CPT) associated with node L. Also, the conditional probabilities of node G given node B are estimated using equation (3.15).

$$\Pr(G = t \mid B = t) = \frac{\Pr(G = t, B = t)}{\Pr(B = t)} \tag{3.15}$$

$$= \frac{\#(Ins\tan ces\_of\_G = t, B = t)}{\#(Ins\tan ces\_of\_B = t)}, \text{ for all values of states } t \text{ and } f.$$

The rest of probabilities of nodes in Figure 3.5 are estimated in the same way.

3.4.5 Bayes' Theorem and Bayesian Reasoning

After obtaining the model from Bayesian learning, the most important benefit of using Bayesian networks in real-life applications is in carrying out probabilistic inference (or reasoning). Bayesian inference is a type of statistical inference in which probabilities are interpreted as degrees of belief. Its fundamental computation is derived from Bayes' theorem, described in equation (3.16) or (3.21). If decisions must be made between two or more choices, then we make a query to see which is most likely. That is, if H_o in equation (3.16) is the hypothesis node and E is the evidence node, then Bayesian inference can be computed from Bayes' theorem. The normalising constant E can be calculated as the sum of all mutually exclusive hypotheses [37].

$$\Pr(H_o \mid E) = \frac{\Pr(E \mid H_o)\Pr(H_o)}{\sum_{i=1}^{n} \Pr(E \mid H_i)\Pr(H_i)} \tag{3.16}$$

Look at equation (3.16) as a formalization of a scientific test of adding enough evidence, as a result of which a hypothesis will be accepted or rejected. In practice, a Bayesian network constitutes a complete probabilistic model of the nodes in a domain that specifies a joint distribution over the nodes. According to [34], a Bayesian network contains the information needed to answer all probabilistic queries about its nodes. The query is a request to understand information about a node set as hypothesis, if some information is provided by other node sets as evidence to the network. The probability outcome of the query is as a result of propagations of the impacts or beliefs of probabilities contributed to by such evidence nodes. The propagation is a mechanism that is propelled through the links in the network. During query or belief updates, every node interrogates its neighbours and compares their parameters. A

node that has more information about the query is thus activated and its parameters will contribute more beliefs. Thus, multidirectional propagation will continue in the network until equilibrium is reached.

The concept of conditional probabilities forms the basic building block of Bayes' theorem. For any two random variables X_i and X_j, the conditional probability of X_i given X_j is defined in equation (3.17).

$$\Pr(X_i \mid X_j) = \frac{\Pr(X_i, X_j)}{\Pr(X_j)} \tag{3.17}$$

Using equation (3.17), together with the chain rule [37] of conditional probabilities, we have equation (3.18).

$$\Pr(X_i, X_j) = \Pr(X_i \mid X_j) \Pr(X_j) \tag{3.18}$$

The order of choosing X_i and X_j does not matter in equation (3.18), as in equation (3.19).

$$\Pr(X_j, X_i) = \Pr(X_j \mid X_i) \Pr(X_i) \tag{3.19}$$

Now, if we equate the right-hand sides of the equations (3.18) and (3.19), we have equation (3.20).

$$\Pr(X_i \mid X_j) \Pr(X_j) = \Pr(X_j \mid X_i) \Pr(X_i) \tag{3.20}$$

Thus, Bayes' theorem is given as equation (3.21).

$$\Pr(X_i \mid X_j) = \frac{\Pr(X_j \mid X_i) \Pr(X_i)}{\Pr(X_j)} \tag{3.21}$$

From equation (3.21), $\Pr(X_i \mid X_j)$ is called the *posterior probability*. The posterior probability of X_i (the hypothesis) is obtained after making observations or studying the behaviour (X_j) of a user. It is the original degree of belief when the likelihood and prior are combined.

Also, $\Pr(X_j \mid X_i)$ is the *likelihood* function of X_i given X_j in the posterior. It is taken as the conditional probability of what we know (the evidence X_j) based on what we do not know (the hypothesis X_i). In other words, it is the probability of seeing the observation X_j given that the hypothesis X_i is true.

In Bayes' theorem in equation (3.21), the *prior probability* of X_i, $Pr(X_i)$, is the probability of X_i before making any observation or any inference.

The *marginal probability* of X_j is mostly taken as a normalizing constant and is expressed as:

$$\frac{\Pr(X_j \mid X_i)}{\Pr(X_j)}$$

It is called the *scaling factor* because it gives a measure of the impact that observations have on the belief of the hypothesis.

Bayes' theorem in equation (3.21) therefore forms the basis of probabilistic reasoning, leading to various sophisticated Bayesian inference algorithms. Examples of popular inference algorithms which are discussed extensively in [3] [8] [49] are Bucket tree, Judea Pearl, junction tree and variable elimination algorithms. In probabilistic reasoning, causal (top down), Diagnostic (bottom up) and explaining-away are the three patterns of reasoning observed in Bayesian networks [37]. The *causal* inference technique uses a cause to infer an effect. In other words, when a cause is observed, we can generate or predict the possible effects. For instance, from Figure 3.5, an example of causal inference is given by making a query on a network using Pr(M = true | L = false). This shows that L is cause while M is effect. The *diagnostic* inference technique uses effects (or symptoms) as evidence to infer causes. It is normally used in expert systems. From Figure 3.5, an example of diagnostic inference is Pr(L = true | M = true). *Explaining-away* is a reasoning technique where the change in belief is a possible explanation, if an alternative explanation is actually observed [37]. For instance, we may want to ask: what are the Pr(B | G) and Pr(B | M) ? These two results can be explained to make better decisions. This type of reasoning is called *Berkson's paradox*, and it uses a causal reasoning step embedded within a diagnostic reasoning.

The backgrounds of DBNs and ordinary Bayesian networks have been presented extensively in this Chapter. However, the next section specifically presents the theories required for the development of Bayesian learning algorithms as relevant to the Genetic Algorithm (GA) presented in Chapter Four. The GA forms one of the building blocks of our proposed EDBN architecture.

3.5 Mathematical Set and Information Theoretic Measures

This section describes the information-theoretic measures and mathematical component (e.g. Power set in set theory) used as genetic operators for learning Bayesian networks from data and as a means of balancing between efficiency and decomposability. These operators have been widely and successfully

used in many areas of scientific and engineering applications [37] [52] [53]. For better understanding, the operators are described specifically in relation to how the Genetic Algorithm we developed learns Bayesian Networks from data.

3.5.1 Power Set Theory

In mathematical set theory, if S is a finite set, then the Power Set of S represented as Pw(S) is the set of all subsets of S [54]. It is a formalized concept of ordering elements of a set. The magnitude of a set S is the number of elements in the set. Supposing the cardinality or magnitude of |S| = n elements, the Power Set of S implies

$Pw(S) = 2^n$ subsets. For instance, if set $S = \{x_1, x_2, x_3\}$, then the Power Set of S implies

$Pw(S) = \{\{\}, \{x_1\}, \{x_2\}, \{x_3\}, \{x_1, x_2\}, \{x_1, x_3\}, \{x_2, x_3\}, \{x_1, x_2, x_3\}\}$.

This is an algebraic concept that we adopted to generate populations of the GA in Chapter Four. The individuals of the initial population $S = \{x_1, x_2, x_3\}$, are referred to as parents which reproduce offspring (the subsets of Pw(S)). This population of parents is modified via crossover processes, which elicit sexual reproduction and natural adaptation.

3.5.2 Mutual Information (MI)

In probability and information theory, Mutual Information (MI) [50] [53] is a smart model that measures information of a random variable X found in a random variable Y, that is, a measure of the amount of information-sharing between two nodes of a network. Its unit of measurement is a bit. The model is given in equation (3.22).

$$I(X, Y) = \sum_{x,y} \Pr(x, y) \times \log_2 \frac{\Pr(x, y)}{\Pr(x)\Pr(y)} \tag{3.22}$$

where Pr(x, y) is joint probability distribution of x and y, while Pr(x) and Pr(y) are the marginal probabilities of x and y respectively. We observe that MI has been applied successfully in many areas of machine learning processes. Some of its important properties are stated as follows:

- MI performs better on probabilistic related variables.

- I(X, Y) = I(Y, X). This implies that, MI is commutative or symmetric.

- $I(X, Y) \geq 0$.

Suppose X and Y are independent, then X contains no information about Y and neither does Y contain information about X. Therefore, their mutual information will be zero. This is obvious in the relations as shown in equation (3.23).

$$\log \frac{\Pr(x, y)}{\Pr(x)\Pr(y)} = \log 1 = 0 \tag{3.23}$$

Thus, MI is a measure of independence such that:

- I(X, Y) = 0, iff X and Y are independent random variables.

This implies that,

- I(X, Y) > 0, iff X and Y are dependent (i. e. share information).

3.5.3 Extended Dependency Analysis (EDA)

The original Extended Dependency Analysis (EDA) is a re-constructability analysis technique [55], which is used to evaluate the level of significance that exists between variables of a training set. The level of significance between two given attributes is determined by their information-sharing. In addition to the original EDA, our modelling technique includes the mutation process of the classical genetic algorithm to reproduce a new sub-candidate network. It furthermore, finds the dependency between every two variables using the model of Shannon Information Content [52], and produces the winning sub-candidate networks. Shannon et al. define information content, $h(x)$ as a natural measure of an outcome x, defined in equation (3.24). For all possible outcomes of **x**, the information content is defined in equation (3.25).

$$h(x) = \log_2 \frac{1}{pr(x)} \tag{3.24}$$

$$h(x) = \sum_{x} \log_2 \frac{1}{pr(x)} \tag{3.25}$$

Information content is measured in bits. The smallest number of bits possible to store the information of **x** effectively is used. Similarly, for information transmission over a network, the information content represented using **67.51** bits is preferred to using **98.31** bits to represent the same information.

3.5.4 Minimum Description Length (MDL)

One of the important components that ensures that a GA is robust is its fitness function, which in our modelling approach measures the optimality of a candidate network. A candidate network is the Bayesian network model presented to MDL for scoring. The Minimum Description Length (MDL) is suitable for measuring the fitness of Bayesian Networks as models obtained from datasets [37]. We use an objective function based on the MDL to minimize the number of bits required to store a dataset and its corresponding Bayesian network. In our Bayesian modelling design, the MDL is iteratively invoked until the minimum score is obtained. The MDL is calculated using equation (3.26).

$$L(D, B) = \sum_{i=1}^{m} \log_2 \frac{1}{pr(v_i)} + \frac{|B| \log_2 m}{2} \tag{3.26}$$

where D = training dataset, L = length of bits required to store both the dataset and its corresponding network. |B| = number of parameters in B, as defined in equation (3.27).

$$B = \sum_{i=1}^{N} (j_i - 1) * k_i \tag{3.27}$$

where j_i and k_i are the cardinalities of the child node and its parent set respectively. Also, N = number of nodes in a network and m = number of samples. The number of bits that are appropriate for representing each numeric parameter is given by $\log_2 m / 2$. Also, $pr(v_i)$ = probability of a sample of D, computed for all values of *i*, . . ., m. That is, $pr(D) = pr(v_i, \ldots, v_m)$ where $pr(v_i)$ is a probability distribution computed using a candidate network. Therefore, the first component (information content) in expression 3.28

measures the length (number of bits) required to store the dataset only and the second component in expression 3.29 measures the length (number of bits) required to store the network model.

$$\sum_{i=1}^{m} \log_2 \frac{1}{pr(v_i)} \tag{3.28}$$

$$\frac{|B| \log_2 m}{2} \tag{3.29}$$

Thus, expressions 3.28 and 3.29 give the MDL scoring measure in equation (3.26) for the optimal network.

There are several other fitness functions for scoring Bayesian networks that have been proposed, such as the Bayesian Dirichlet (BDe) metric, the Bayesian information criterion (BIC), etc. The BDe uses Bayes' rule with some maximum likelihood estimators. The details of these fitness functions are not relevant for the modelling approach in this thesis.

3.6 Conclusions

Through this chapter, we have described the theoretical backgrounds of DBNs that methodically allow developing the proposed EDBN architecture and accounting for the inherent ongoing computational intensities associated with learning BNs from massive datasets. Specifically, the basic principles and mathematical models required for developing learning algorithms such as GA were presented for modelling BNs. In the following chapters of this thesis, these algorithms and mathematical principles are extended, based on DBNs in order to achieve SA technologies and provide economical solutions to the intensity problems.

In the theoretical backgrounds presented, probabilistic modelling was: used as objective focus when describing the DBNs; the focus in the fundamentals of probability theory; characterised when capturing the complexities of systems in Bayesian networks; used as one of the bases of Bayesian inference; and the idea used to describe situation awareness.

The focus of this chapter is not on developing practicable SA technologies via DBN models, but rather on providing fundamental theories that can be used to evolve BNs simply and dynamically. The next chapter presents our enhanced GA that we developed to first emerge Bayesian networks as an important component of our proposed architecture.

Chapter Four

Emergence of Optimal Bayesian Network Models – Genetic Algorithms

4.1 Introduction

Bayesian Network technology is very useful for encoding probabilistic knowledge as graphical structures. It is rapidly gaining popularity in modern Artificial Intelligence (AI) for solving real-life problems involving reasoning under uncertainty. This network belief technology has been successfully used for reasoning in the areas of audio-visual detection [1], anomaly detection [56], and social networks [57], for example. Knowledge is expensive to acquire, and most of the modelling time, there are no domain experts or knowledge engineers to interpret environments and model knowledge as Bayesian belief networks. Unlike simple known patterns obvious in limited datasets, large organizational datasets contain complex and unknown patterns. Since data can be obtained in confidence and contain useful hidden information about the environments, Bayesian Networks offer a great advantage that can capture and encode this hidden information as knowledge. Knowledge engineers may find it very difficult to manually build a Bayesian Network model that can reveal and explain the activities in these datasets. Models must capture context – namely, given environmental states must be captured in the models and the models must evolve with a changing environment in order to be fully adaptive.

One can recall that finding an optimal Bayesian Network from data has been shown to be computationally intensive [20] [21] and recent studies [21] [23] [24] [25] [39] have investigated the challenges of learning networks from data. With a different view from the issues raised in Chapter One, a number of algorithms have been proposed to learn some forms of network structures. Koivisto et al. [21] developed an algorithm for structure discovery in Bayesian Networks using several mathematical theorems and propositions. Larranaga et al. [23] came up with a variant of a GA using classical GA operators and representing chromosomes as network. Friedman et al. [24] proposed an algorithm that integrates a Markov Chain Monte Carlo (MCMC) procedure with variable ordering based on approximated posterior probabilities for scoring the network structures. Myers et al. [25] also developed a variant of a GA by representing genes as variable nodes. Myers et al. represented classical GA methods as add, delete and reverse operators which are used in the K2 algorithms [8] to learn networks. With the research concern of this chapter, one of the limitations of these algorithms and techniques is their inability

to graphically ascertain the quality of their final networks. Many of these algorithms are hard to implement because of the inclusion of highly mathematical terms and symbols. Also, we identified backtracking as an overhead that contributes to the computationally intensive problem of learning Bayesian Networks from data. Backtracking consists of undoing the addition of an edge from a network when it does not improve the score of the structure or when it results in a cycle.

In this chapter, we present a simple computational intelligence technique called a Hybrid Genetic Algorithm (HGA) that is implemented for learning optimal Bayesian Networks from datasets. It is a core modelling component of our evolving Dynamic Bayesian Network (EDBN) architecture, the detail of which is, however, beyond the scope of this chapter. It not only adopts the ordinary GA operators but also integrates information theoretic measures as learning components and mathematical power sets as population constructions. The information theoretic measures that we used include: Mutual Information (MI), Extended Dependency Analysis (EDA), and Minimum Description Length (MDL), as described in Chapter Three. Our approach improves on the original EDA by integrating the model of Shannon's information content [52]. In mathematical power sets, the crossover process is implemented, which is advantageous to our network solution space because it tries to guard against illegal structures in order to prevent the cycle problem. Much of the existing research [23] [39] often encounter cycle problems and therefore use the remove operator to backtrack. This contributes to the overhead learning problem. To avoid backtracking when evaluating an entire candidate network structure using MDL, we introduced an inner loop by using MI to check if a produced offspring is probabilistically fit as a candidate sub-structure before adding it to the entire candidate network structure for scoring.

During this implementation phase, the HGA adopts the ADTree technique [30] to speed up the learning process as an improvement over our initial presentation [58]. ADTree uses memory to summarize an entire training dataset to reduce the data size where a network is emerged. The implementation results of our HGA have demonstrated a very good modelling capability when evaluated with publicly available block-lifting data [37], Asia network [59], the work of William et al. [39] and when compared with the models generated using Weka [26] and Hugin [45]. Moreover, like most existing algorithms, our technique can learn networks from mixed (numerical and nominal) or nominal (textual) datasets. In addition, HGA can also learn networks from numerical datasets that are not discretized. Most existing systems do not show this capability but instead need to discretize all attributes of the datasets. Our major contributions in this chapter are as follows:

- Identification and avoidance of backtracking for enhancing HGA and the integration of ADtree method for its learning speed optimization.

- Experimental results with public datasets motivate its successful applications in optimizing response rate for immediate detection of anomalies in telecommunications networks using real-life call data.

The rest of this chapter is arranged as follows: in section 4.2, we describe the system model used for the implementation of the HGA components which integrate the ADtree structure. The detailed approach of our HGA and how its operators interact are presented in section 4.3. In section 4.4, we present the well-defined outlines for applying the enhanced HGA. Section 4.5 critically evaluates the HGA as it benchmarks with the Nilsson and ASIA networks, and illustrates backtracking. Section 4.7 reapplies the enhanced HGA to optimize modelling for immediate detection of anomalies in call data. Section 4.8 concludes the chapter.

4.2 The System Model of HGA

A Genetic Algorithm (GA) is an evolutionary optimization algorithm that solves problems by eliciting the process of evolution [50]. Figure 4.1 briefly shows our system model of the HGA as decomposable components for its learning processes.

The HGA was developed from the domain of evolutionary algorithms to emerge optimal Bayesian networks from datasets. For its learning process, it uses genetic operators engineered from information theoretic and mathematical fields including Mutual Information (MI), Extended Dependency Analysis (EDA), Mathematical Power Sets and Minimum Description Length (MDL). Unlike our HGA, existing genetic algorithms (GAs) use genetic operators that usually use backtracking, which is an overhead for a learning algorithm. In this chapter, we prevent backtracking using an inner-loop, and carry out several evaluation experiments. The HGA control component methodically coordinates the iterative learning processes of all the operators. The population constructor receives strings of attributes from data sources and generates the space of initial parents which are modified through processes that elicit sexual reproduction and natural adaptation. The iterative construction of offspring from parents leads to new generations until solution space is exhausted. In Figure 4.1, the EDA receives a subset of offspring from the HGA controller and produces a sub-candidate network. It passes candidates of not more than two variables at a time to the MI. The HGA controller iteratively invokes the fitness function (MDL) until a minimum score is obtained. The detailed description of Figure 4.1 is presented in Section 4.3.

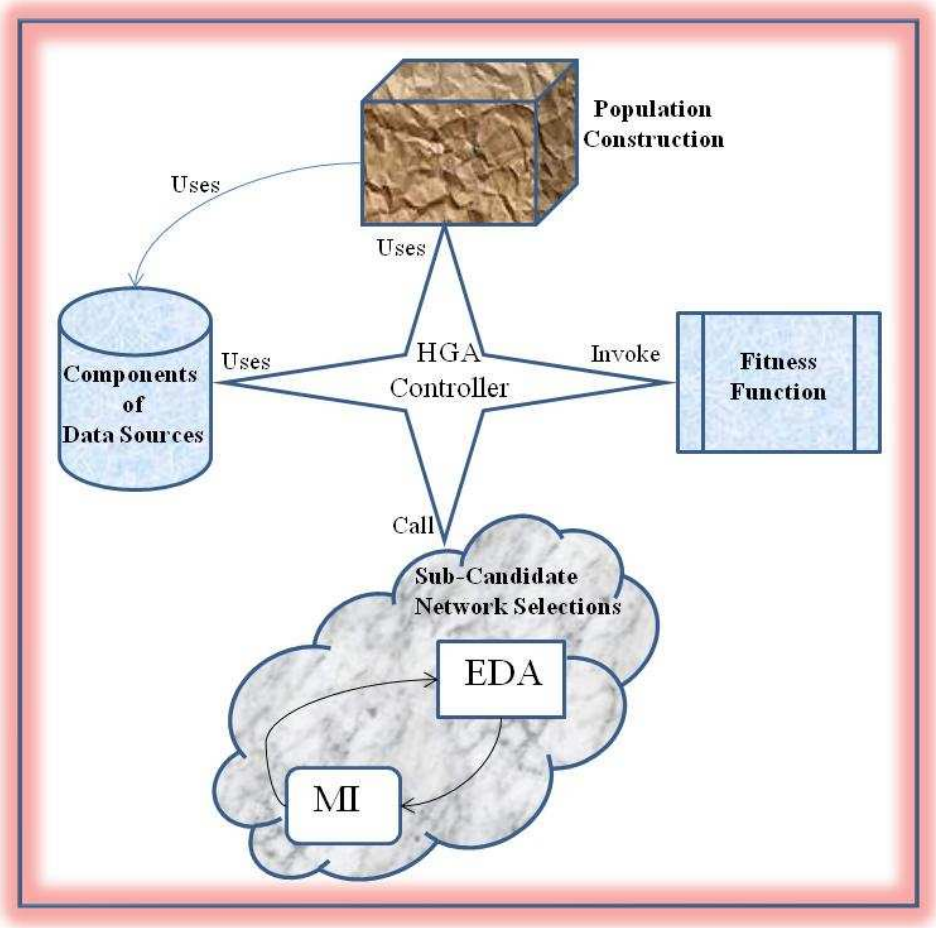


Figure 4.1: A System Model for the Hybrid Genetic Algorithm (HGA)

4.3 The Descriptive Approach of HGA

The HGA learning begins with an initial population P, say $\{\{x_1\}, \{x_2\}, \{x_3\}, \{x_4\}\}$ called parents, and of which the independent network: (x_1), (x_2), (x_3), (x_4) is formed. Every parent in the initial network is evaluated with the MDL model-scoring measure (or fitness function). Every parent in P such as $\{x_1\}$ produces offspring during several crossover processes with other parents, such as $\{x_2\}$, $\{x_3\}$, $\{x_4\}$. The mathematical Power Set implements this crossover process and tries to guard against cyclicity. Furthermore, we implemented a network checker that confirms the absence of the cycles. There is an equiprobable selection of every other parent for producing a child subset. This means that the probability of producing offspring from a pair of parents is 0.25 each for the above example population. To avoid backtracking during the evaluation of an entire candidate network structure using MDL, we introduced an

inner loop by using MI to check whether an offspring produced is probabilistically fit as a sub-candidate network in the entire Directed Acyclic Graph (DAG).

For every survived new offspring such as $\{x_1, x_2\}$, the prior probability that $\{x_1\}$ or $\{x_2\}$ is a parent of each other, is 0.5 respectively. Since they are equiprobable, it becomes difficult to decide which should be the parent. So, a new offspring $\{x_2, x_1\}$ is produced using the EDA by mutating $\{x_1, x_2\}$. There is also equal prior probability 0.5 of $\{x_2\}$ or $\{x_1\}$ being a parent of each other. It is also difficult to decide between the two. Therefore, these two different individuals will respectively formulate candidate sub-networks. In order to decide which attribute is the parent, the two candidate sub-networks compete with each other and the winning candidate sub-network will emerge. This means that a more fitting candidate sub-network will serve as a building block to the entire DAG. The selection of a more fitting candidate sub-network is guided by Shannon's information content in which the EDA decides and chooses a candidate sub-network that will minimize the score of the DAG. This optimization process is repeated until the whole solution space is exhausted and the best Bayesian Network model emerges.

The new generations of individuals can be produced from the current generation of offspring, such as $\{\{x_1, x_2\}, \{x_1, x_3\}, \{x_1, x_4\},. . ., \{x_3, x_4\}\}$. In contrast with this, one of the distinguishing features of our methodology is that we control the new generation of individuals to avoid repetition of already-existing individuals (or redundancies). The larger subsets of individuals will be broken down into smaller ones at the learning stage. Since the possible smaller subsets of individuals are produced in the older generations, we observed that less time may be wasted if the reproduction of newer generations is controlled.

The HGA keeps optimizing the learning until convergence is reached and the solution space is exhausted. Given the above description, our network learning component of the EDBN is a variant of the classical GAs from biological processes. The HGA is standardized and formally presented in a simplified outline shown in subsection 4.4.

4.4 The Hybrid Genetic Algorithm (HGA)

For the purpose of applying the HGA, the outlines are formally presented as follows:

1. Set $i = 0$, as a generation of an initial population.
2. Construct an initial population, P_i, from training set attributes $X = \{x_1, x_2, \ldots, x_n\}$ to form singleton sub-candidates, which produce a candidate network.
3. Evaluate the fitness of the candidate network using MDL.
4. While the solution space is not exhausted or $i <$ total number of generation N_g

- Select a sub-candidate network to undergo a crossover process which produces new offspring (non-singleton sub-candidate).
- Perform a mutation process on the sub-candidate to form another new offspring using EDA. The two sub-candidates have equal probability of being selected as a building block of the entire candidate network (DAG). The two sub-candidates compete for fitness and minimal score using MI and EDA as inner-loop.
- Optimize DAG with the winning sub-candidate. The optimal DAG formed in every generation is evaluated by the MDL and offspring of higher generation are produced from the offspring that are fit from the previous generation.
- Select a new generation $P_{i+1.}$

5. Repeat steps in 4 until the minimum score for the optimal candidate network emerges from the use of the MDL.

The details of various standard information theoretic measures used as genetic operators in the Bayesian learning algorithm were presented in Chapter Three.

4.5 Experimental Evaluations

One of the objectives of our HGA is to bring theory to practice with an emphasis on applications and practical work. To ensure the universality of our modelling approach, we used publicly available datasets [31] [37] with the evaluation information supplied with them, and compared them with the ASIA network [59]. We also used Weka [26] and Hugin [45] to validate the performance of our HGA, as shown in the next subsections.

4.5.1 Structural Evaluations by Using the Nilsson Network as Benchmark

This subsection provides structural evaluations by measuring the similarities between the network learned by our HGA and the Nilsson network [37] using the same dataset. We validate this measurement by comparing the resulting network with the network learned using Weka and Hugin. Figures 4.2 to 4.5 show the networks. Nilsson [37] provides, together with the benchmarking network, a small dataset that contains the operation of a block-lifting machine. The operation is monitored with attributes: battery (B), movement (M), liftable (L) and gauge (G). By comparing Nilsson's network in Figure 4.2 with Figure 4.3, it shows that there is an opposite orientation between nodes G and B. One can see that Figures 4.3 to 4.5 are the same. The Weka modelling time is 0.08 secs while the HGA took 0.07 secs. Hugin’s time was

unavailable. By avoiding backtracking on the small dataset, HGA proves its power by minimizing computational intensity – it executed 14.3% faster than Weka.

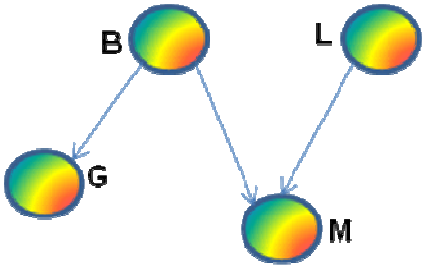


Figure 4.2: Block-Lifting Model From Nilsson.

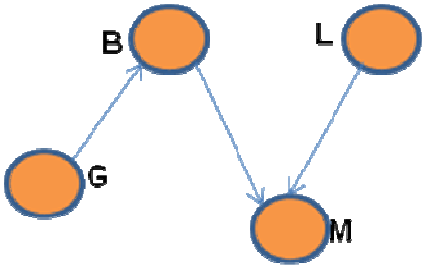


Figure 4.3: Block- Lifting Model From Implemented HGA.

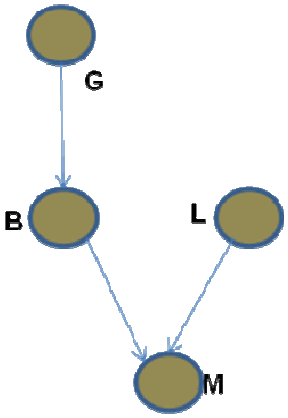


Figure 4.4: Block-Lifting Model from GA of Weka.

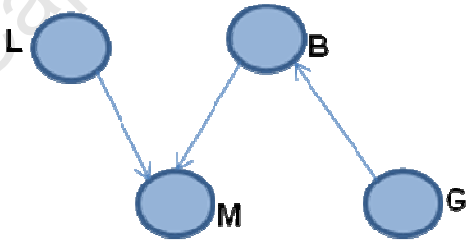


Figure 4.5: Block-Lifting Model from Hugin.

4.5.2 Structural Evaluations by Using the ASIA Network as Benchmark

We also validated the performance of our HGA by benchmarking it against the standard ASIA network that is commonly used in the Bayesian learning literature [23] [39]. The original ASIA network [59] and the ASIA network that we learned are shown in Figures 4.6 and 4.7 respectively. The MDL score given by the HGA is 59145.869 bits. The ASIA network, with nine attributes, is a medical model used to diagnose cancer diseases. We evaluated the HGA using 10,000 instances of training data generated from the original network [59]. The qualitative output obtained in Figure 4.7 has 9 similar links (orientations) with a missing link compared to the original network. The happy graph in subsection 4.5.3 highlights more benefits of the HGA compared with an existing modelling method of the ASIA network.

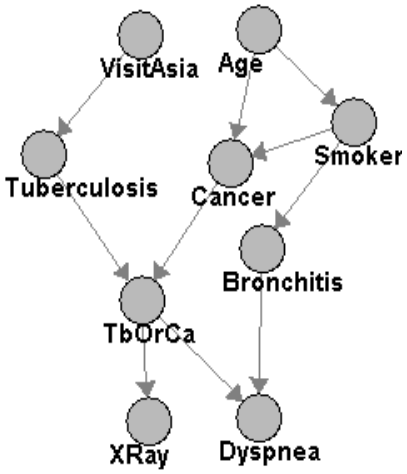


Figure 4.6: The Original ASIA Network.

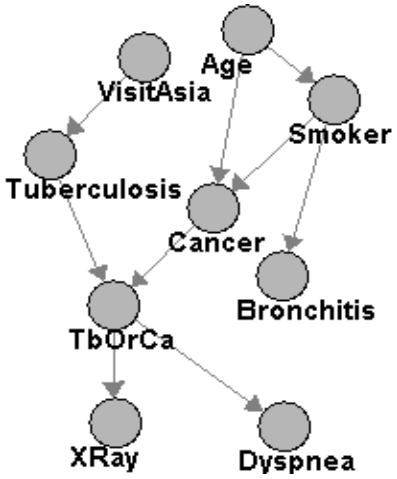


Figure 4.7: The ASIA Network Learned from HGA.

4.5.3 Validation of Avoiding Backtracking Using Happy Graph

Russell et al. [8] refer to a learning curve for an algorithm on a particular domain as a happy graph. In Figure 4.8, as the iterative optimization increases, the fitness function (MDL) score for the candidate network improves (or minimizes). Instead of the network score becoming worse, and later backtracking, the local convergence is reached at (2, 90000) before the solution space is exhausted. This evidence shows that the inner-loop in our HGA prevents this backtracking problem and the learning process does not get stuck at the local convergence.

Figure 4.8 is referred to as a happy graph. The existing methods add an edge to the network and, if it worsens the optimization, it is deleted. This is an overhead of backtracking which is avoided by the inner-loop in our approach. We also compare our result with William et al. [39], who used a similar ASIA network like ours but their result is based on average fitness of scores. In their process of learning, as shown in Figure 4.81, they obtained a series of increase (worsening) and decrease (improvement) of fitness scores and calculated the average. Time is expended as they backtrack on the learning process, which could otherwise have been avoided. Thus, by comparing the two Figures 4.8 and 4.81, our modelling approach in this chapter contributes to the preliminary reduction of overheads (or intensity) in learning Bayesian Networks from datasets.

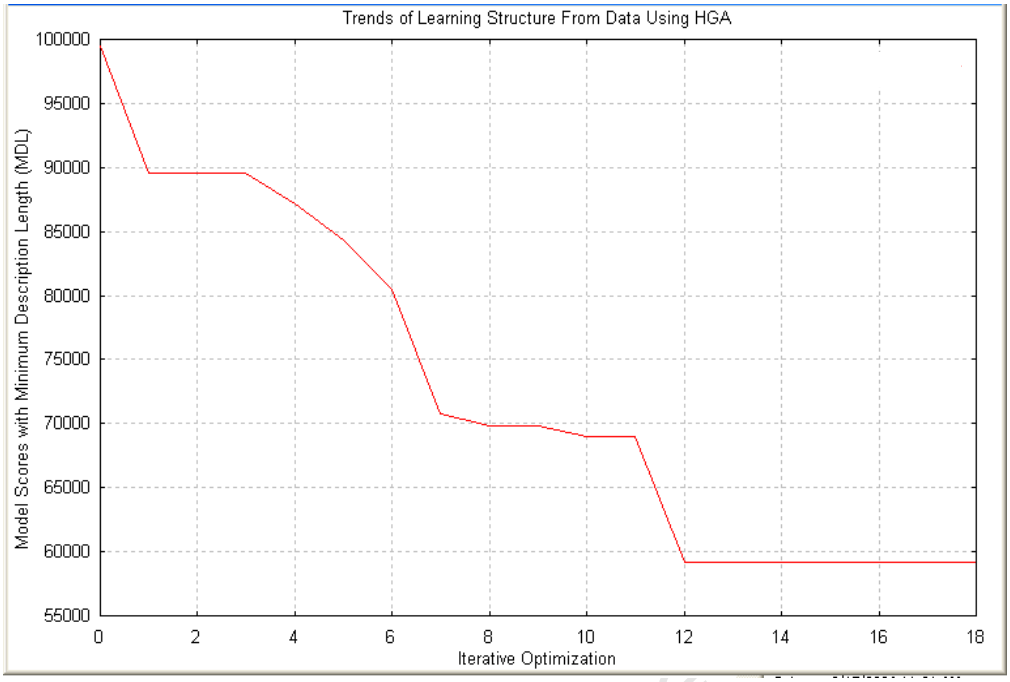


Figure 4.8: Avoiding Backtracking Process during Optimization of Learning ASIA Network Using the HGA

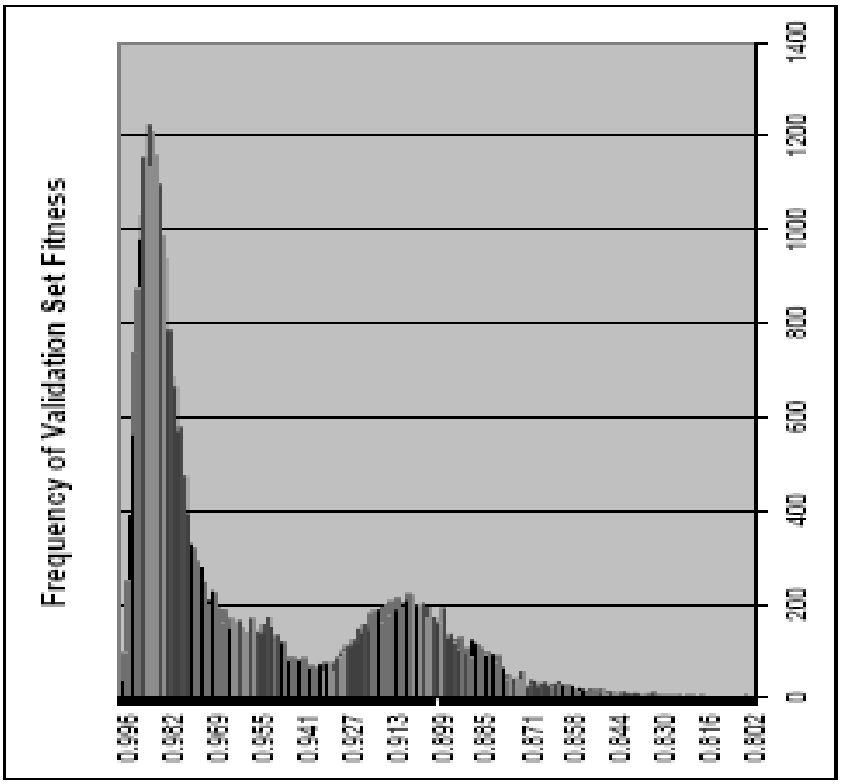


Figure 4.81: Several Backtracking Process during Optimization of Learning ASIA Network [39]

4.6 Applications in Telecommunication Networks

The objective in this section is to reapply the HGA to model individual subscriber behaviour to enhance the *response rate* of anomaly detection techniques for achieving immediate detection.

Anomaly detection in telecommunications call data tries to discover deviant behaviour of individual subscribers. Malicious behaviour has been noted as one of the key causes of such anomalies, which have consequently led to unquantifiable losses of revenue to many telecommunication networks world-wide. Although the intentions of most subscribers to these networks are unknown when making phone calls, their behaviour pattern is reflected in their call data. Recent studies have investigated the challenges of anomaly detection but have not given conclusive solutions to address this problem. To enhance the *response rate* of our recently developed Telecommunications Anomaly Detection System (TADS) [58], this section applies the improved HGA which efficiently facilitates immediate detection that will assist call analysts and managers with quick decision-making. We therefore make a conjecture that if the individual subscriber modelling of the HGA is optimized then the *response* rate of the TADS will be enhanced as it acts upon anomalies in telecommunication networks immediately when these occur.

4.6.1 Anomaly Problems in Telecommunication Networks

Since the telephone is one of the fastest means of communication, many subscribers use contract and pre-paid phones for crucial business activities. Security deficiency is observed in the fixed and mobile networks. In 2001, half a million fixed lines in South Africa were disconnected, many as a result of fraudulent activities [60]. The security measures that include the use of a Mobile Identification Number (MIN), Equipment Serial Number (ESN) and a Personal Identification Number (PIN) were meant to prevent spoofers (or unauthorised users) from gaining access to a subscriber's phone account. Please see Figure 4.9 for more illustration. These security measures are deficient due to unlawful methods employed [19], such as eavesdropping, cracking, shoulder surfing, and social engineering [61]. To worsen the situation, the MIN/ESN numbers for digital phones (e.g. GSM) that were initially well encrypted during transmission, were later cracked (made simpler) because of the decoding disparity from one country network to another [61]. As a result, internetwork linking countries have lax security which is easy to breach. This insecurity allows the spoofers or hackers to make anomalous calls at the expense of the lawful phone owners.

The term 'anomaly' refers to an outlier and to suspicious data that can easily be spotted in small datasets, but is hidden and requires intelligent detection in the massive amounts of call data in

telecommunications environments. Common inconsistencies in call data are caused by customer churn or attrition, failure in networks, potential fraud, deliberate or unintended expensive mistakes made by telecommunications' workers, bad debt risk, or even improper disconnection of communication lines [56] [62]. Malicious behaviour has been noted as one of the key causes for such anomalies, which have consequently led to unquantifiable loss of revenue to many telecommunication networks world-wide [60]. The most common cause of such malicious behaviour is potential fraud, because it is committed intentionally and it is difficult to combat. To complicate the situation further, the network carriers often do not want to admit that fraud exists as an anomaly in their systems, so that their subscribers do not suspect that fraud is a significant problem. If they do acknowledge it as a significant problem, it might cause churn or cause more subscribers to try to commit fraud. Instead, they prefer to solve this problem intelligently. If the subscribers suspect fraud but nonetheless keep paying for debts that they did not incur and for services that they did not receive, and if the Telecommunication Service Provider (TSP) does not find a solution to the problem, these customers may decide to seek alternative competing service providers. Also, service providers are greatly challenged when subscribers change to competing carriers, due to feeling uncomfortable with the service provider's services, including for example, over-charging, not answering queries promptly, or general bad customer service. These changes contribute to suspicious and sudden changes in call patterns that are reflected in their call data. It results in the loss of revenue, which is mostly attributed to anomalies. This could quickly put a telecommunication company out of business. For these reasons, it is obvious that anomaly prevention is defeated. This necessitates the use of immediate detection techniques enhanced by our improved HGA for existing subscribers who experience anomalies.

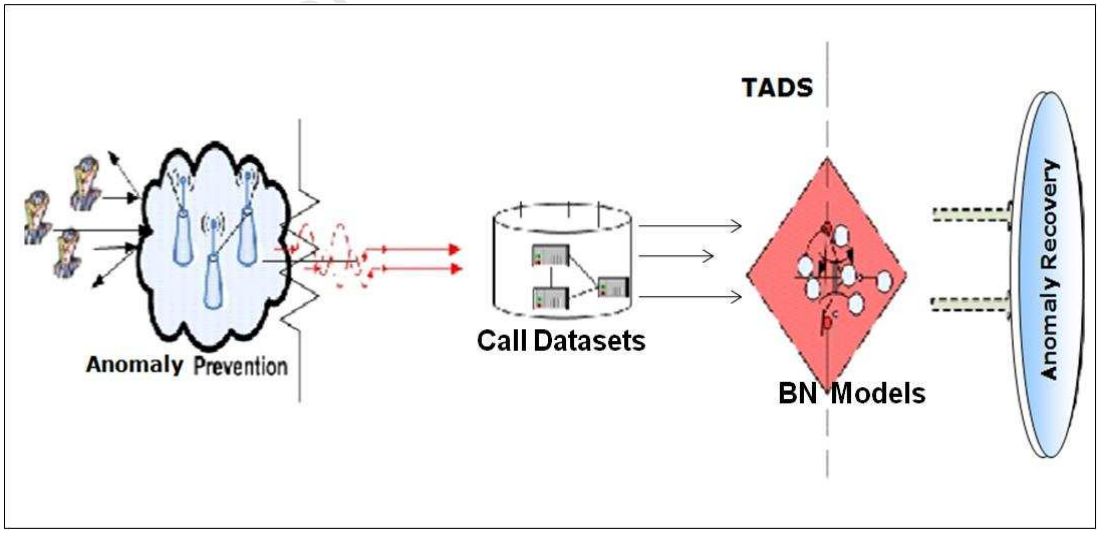


Figure 4.9: A Descriptive Model with Quick Overview of How Intentional Anomalous Call is Committed and Detected Intelligently.

As shown in Figure 4.9, a reduced lag-time (difference between time of call and detection time) will enable service providers to detect and act upon anomalies faster, which will reduce the damage caused by these anomalous behaviours. Also, an anomaly detection method can work well for a finite number of known anomalies, but due to the difference in behaviours of subscribers as a result of uncertainty in the use of phone services, a system for detecting an infinite number of unknown anomalies is preferable. Accomplishing this objective is very challenging and requires continuous improvement in detection approaches. Therefore, a TSP (Telecommunication Service Provider) that can detect the trustworthiness of any call record immediately after a call session is ended will flourish and create a level of confidence between itself and its subscribers.

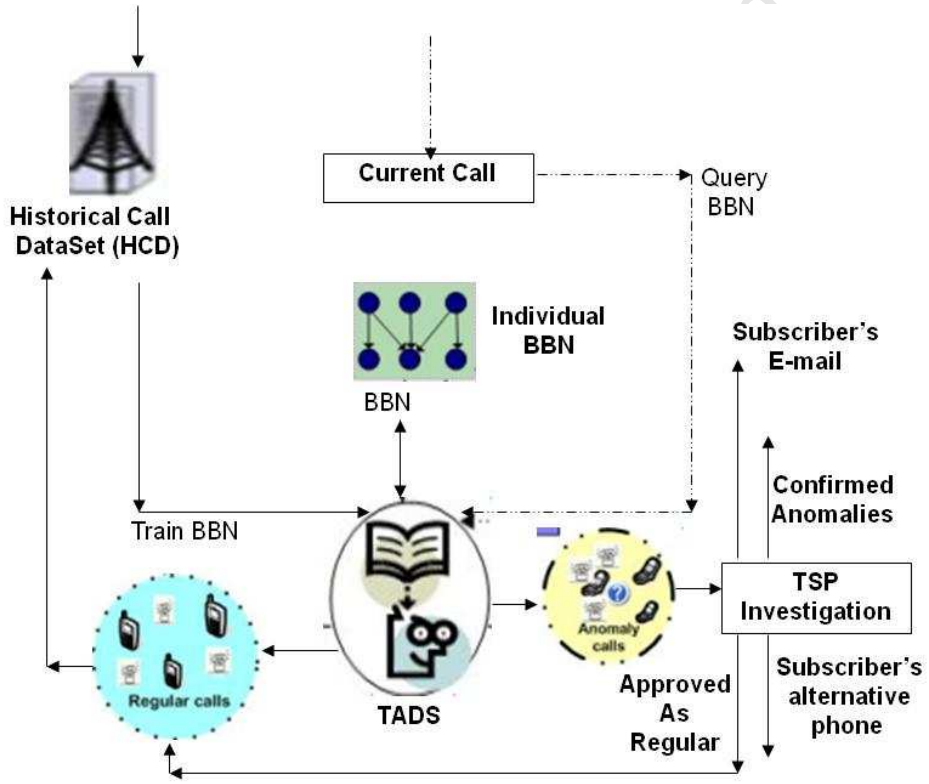


Figure 4.10: Our Individual Modelling System with Integration of Alternative Rapid Response Contacts of Subscribers.

By improving on the speed of our individual modelling approach as shown in Figure 4.10, the HGA improves the *response* rate of the TADS that immediately detects any anomalies and minimises false alarms to an acceptable level. The TADS comprises the following four components: Call Sampling Probability, Call Prediction, Call Detection, and Degrees of Detection [63]. Distinguishing between calls made to new destination numbers and detection of anomalous calls is difficult but the TADS uses

anomaly indicators [63] to address this problem effectively. An anomaly indicator is the most significant attribute in call detail records (CDR). In principle, the Call Detail Records (CDR) that describe subscribers' user profiles generally contain relevant attributes, such as a caller's number, location, duration, destination number, date and time of call. Assignment of equal weights to all these attributes, even though some carry more information than others, is an approximate assumption. Unequal weighting is determined as indicators, which improve the appropriateness of TADS, and telecommunication revenue losses will be significantly reduced. To realise this constructive idea and to facilitate a trusted relationship, a warning notification through alternative phone, sms, or e-mail can be sent by the TSP to subscribers whose account reflects abnormal call patterns. These alternative contacts are also a means of making rapid response as described on the individual modelling system shown in Figure 4.10.

4.6.2 Modelling Approaches to Address Anomalies

A number of methods used for anomaly detection include: rule-based approaches, statistical techniques, neural networks, distance-based approaches, and Bayesian networks. It was presented in recent studies that unsupervised learning of Bayesian networks for anomaly detection, which is optimized in this section, is better than most detection methods. This is because it does not need to preset types of fraudulent activities and can detect new anomalies from call data. Improvement on Bayesian network modelling is an ongoing research effort and it is our focus in this section to apply the improved HGA to enhance the *response* rate of anomaly detection.

The *rule-based* techniques work best with user profiles containing explicit information, where anomaly criteria are encoded as rules [64]. This technique is difficult to manage because it requires explicit rules which are labour-intensive and time-consuming, and involves programming for every imaginable possible anomaly. The *statistical* methods of anomaly detection work well with univariate distributions, where the average call duration, the longest call duration and the average number of calls per day are usually compared with a pre-determined threshold [65]. If the number of calls for a day exceeds its normal behaviour (threshold), it is considered as anomalies which may not be true. The popular *neural network* technology [66] employs feature extraction, where summarised statistics are usually used to train models with Call Detail Records (CDR). The extracted features from CDRs are pre-classified as either a regular or an anomalous call. This technique can detect known anomalous calls in the training data. Hollmen et al. [56] presented the use of a Bayesian Network in anomaly detection by profiling users' behaviours and used known anomalous scenarios, but did not address new types of anomalous calls. Also, the *Distance-based* approach is described in [62]. In this method, an absolute distance between a dataset and a defined point as a distance outlier is used. This approach focuses

primarily on continuous datasets. The approach of TADS using the HGA differs from theirs as we can also process mixed datasets. This section describes how individual models are learnt from data using the HGA which are then used to detect anomalies immediately after a call session ends. This reduces the time lag with degrees of detection. Also, the differential threshold makes the TADS adaptive and minimises false alarms to an acceptable level.

4.6.3 Experiments

This section shows the immediate call detection results for individual subscribers using the HGA and the TADS. The objective here is to know the impact of the improved HGA on the response rate of the TADS. We obtained real-world land-line call data for over 160 TSP subscribers, and more than 73,000 calls with nine attributes. The results of Figures 4.11 to 4.15 are examples of individual subscribers models obtained from every historical call profile using the HGA. The quicker the models are emerged, the better the response rates to achieve immediate detection are. By masking the subscribers' phone numbers, we have ensured the confidentiality of network carriers based on the policies to protect telecommunication customers.

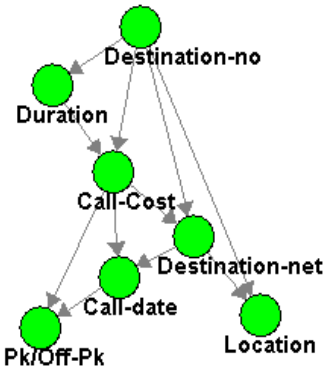


Figure 4.11: Subscriber 145521137 Call Behaviour.

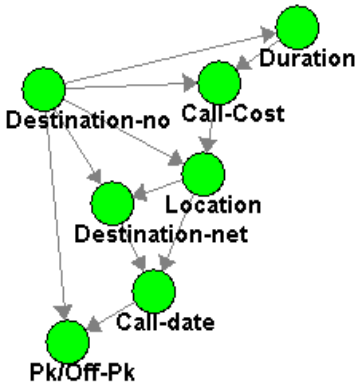


Figure 4.12: Subscriber 145521198 Call Behaviour.

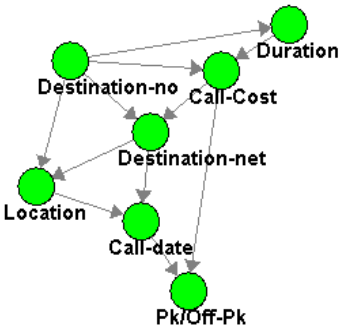


Figure 4.13: Subscriber 145571179 Call Behaviour.

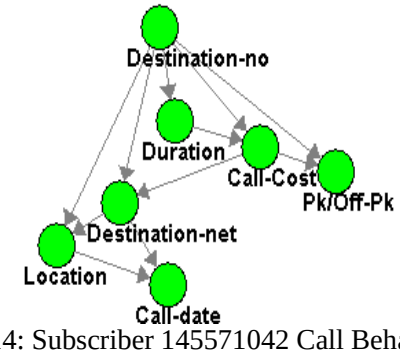


Figure 4.14: Subscriber 145571042 Call Behaviour.

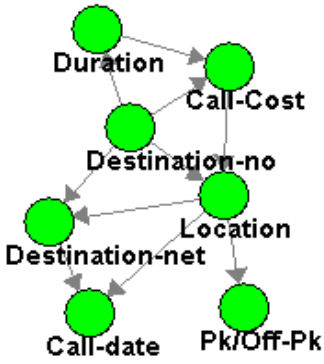


Figure 4.15: Subscriber 145571030 Call Behaviour.

The TADS uses the mathematical models in equations (4.1) and (4.2) for determining the degrees of anomaly detection for any instance of call record. The two components here are call sampling probability $\Pr_{E}^{\Xi}(\vec{e})$ determined from the training sets, and call prediction $\Pr_{E}^{BN}(\vec{e})$, which is the probability of a call record inferred from the individual Bayesian networks using the inference algorithms. This is where the optimized HGA learns models faster to support the rapid response rate of the TADS to achieve immediate detection. A deviation *below* the sample probability is computed using equation (4.1). A deviation *above* the sample probability is computed using equation (4.2). The components of equations (4.1) and (4.2) use differential analysis due to their regular update in input values. The results imply that we are **X.xx**% sure that a call is *anomalous* or *regular*.

$$\left(\frac{\left|\Pr_{E}^{\Xi}(\vec{e}) - \Pr_{E}^{BN}(\vec{e})\right|}{\Pr_{E}^{\Xi}(\vec{e})} \times 100\right)\% = \textbf{X.xx}\% \qquad (4.1)$$

$$\left(\frac{\left|\Pr_{E}^{BN}(\vec{e}) - \Pr_{E}^{\Xi}(\vec{e})\right|}{\Pr_{E}^{BN}(\vec{e})} \times 100\right)\% = \textbf{X.xx}\% \qquad (4.2)$$

For a test case, 90% of the historical call records were used as training sets to model the individual networks and 10% of anomalous test data (noise) was randomly generated. The noise records were generated as known anomalies using the properties of the real dataset and it was repeated for other subscribers. Since all the test records are known anomalies, 100% detected anomalies are expected. Subsequent call records of subscribers are immediately expected to be detected as either anomalous or regular events. The regular call events are used to re-train the old model to adapt new knowledge. In this

test case, the calls detected as anomalies are regarded as true (correct) detection, while the calls detected as regular calls are false (incorrect) alarms. The accuracies of the detection are summarised and presented in Figure 4.16. In this case, the average accuracy of correctly detected anomalies is 74.754%, while incorrect (error) detection is 25.246%. One can see in Figure 4.16 that, for every subscriber, the true positive rate is greater than the error rate. We cannot have a group model for subscribers who behave in a similar way. If we do not have individual models to separate subscribers' behaviours so that detection is immediate, the false detection rates may be greater than the true positive rates.

The average *response* rate of detection on a subscriber is now 0.42 secs [63] as compared to its initial exponential rate whose focus was a proof of concept [58]. The constrained BN learning time is optimized. This implies an improved *response* rate but the quality of detection is still maintained as presented in Figure 4.16. One can see that an optimized learning algorithm like the HGA is necessary to support the enhancement of *response* rates of a TADS.

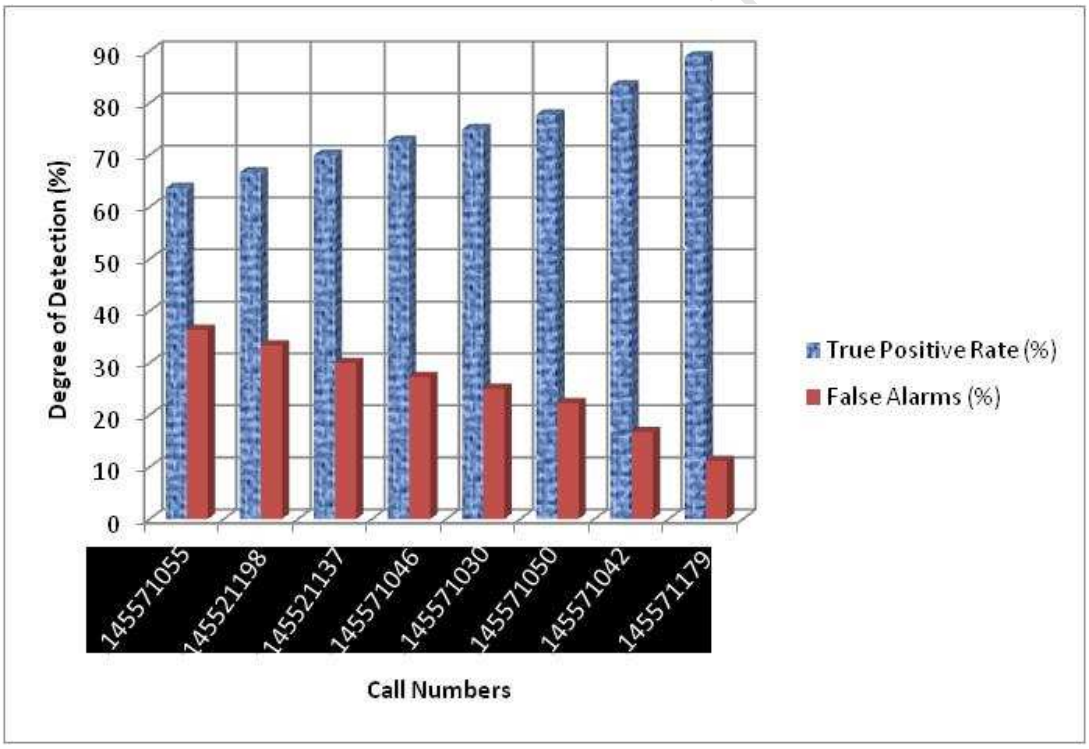


Figure 4.16: Anomaly Detection Accuracies with Improved Modelling Speed for Eight Subscribers Maximising True Positive Rates and Minimising False Alarms.

4.7 Conclusions

We have identified and emphasized in this chapter that backtracking in the emergence of optimal Bayesian Networks is an overhead problem which needs to be avoided to minimise computational intensity of learning. This optimizes and standardises the development of HGA. This strategy is achieved by emphasizing the inner-loops, and also integrating the ADTree to speed up the learning process. This is our preliminary solution for addressing the computational intensity (or NP-hard) problems of learning.

The decomposability of the HGA makes it simple to understand and easy to implement. We presented our results structurally as an obvious evaluation method. The efficiency and proven performance of our experiments make the approach of the HGA suitable to interesting applications in various research and industrial areas, such as telecommunications networks, upgradable for learning in Dynamic Bayesian Networks, etc. We have established a proof of concept in this chapter, and subsequent chapters in this thesis handle the emergence of the network models from massive datasets using sophisticated technologies.

We have demonstrated how immediate detection of telecommunication anomalies can be achieved, as optimized learning algorithms like the HGA can learn individual Bayesian Networks to enhance the response rates of detection models (e.g. TADS). In our detection results in Figure 4.16, the TADS aimed at and succeeded in maximising true positive rates and minimise false alarms with average rapid *response* rate of 0.42 secs of detection, much better than our initial exponential rate [58].

With this impressive performance of our HGA, we want to further improve our modelling approach to make the *response* rate a gold standard, such that detection will be carried out before communication lines are opened. We intend to apply it to other interesting application areas such as robotics used in human daily activities (e.g. robotic soccer, robotic oil exploration, etc.), internet banking, business pattern recognition, etc. As presented in the next chapters, we have a vested interest in upgrading and integrating the HGA into the architecture of Dynamic Bayesian Networks and frameworks to evolve models from small datasets up to massive datasets.

Chapter Five

The Proposed Evolving Dynamic Bayesian Networks (EDBN)

5.1 Introduction

There are observations of moderate to a massive multivariate time series (MTS) obtained from dynamical systems in the professional world of industrial practices and research such as oil exploration, environmental management, business activities, etc. The ever-changing professional environment extensively increases complex hidden relationships concerning the activities embedded therein, and the development of sophisticated technologies that can be evolved according to how the environments are changing, in order to capture hidden and useful information, is strongly required.

One of the key components of revealing hidden and useful information is to model such an environment with Dynamic Bayesian Networks (DBN) for easy prediction and/or monitoring of the events unfolding. A variety of such models have recently been developed to address this objective, mainly using hidden Markov model (HMM) representations such as coupled HMM [5], hierarchical HMM [3], Factorial HMM [6], etc. For the same purpose, DBN for audio-visual recognition [1], BBDBN (brain and body DBN) [2], structural DBN (SDBN) [7] are variants of DBN which do not concentrate only on achieving efficient predictions, but also face the challenges of presenting their network structures due to rigidity of the models. A DBN model consists of two components of knowledge – the network structure and the probability distributions [3] [8]. Like the others, a Partial DBN (PDBN) [36], for example, learns a network structure with many domain assumptions, repeating the same network structure over time, and explicitly connects to a target response node. It interconnects multiple instances of same Bayesian Networks (BNs) with temporal relation into the DBNs. The PDBN identifies a minimum set of state variables to separate evidence nodes (or variables) between two time steps. On the other hand, the HMM, which other advanced models often use as a baseline of comparisons, is the simplest form of explicit representations of DBNs. Here, same network structures and conditional probability tables (CPTs) can be repeated over time while a mathematical function (e.g. Poisson, Gaussian distributions, etc.) can be used to handle the probabilistic process.

These existing models have contributed to the baseline of temporal modelling but they are, however, restricted (or rigid) with assumptions and are limited in their expressive power, because they often require the intervention of domain experts. So, because of the challenges to directly and completely learn the two components of a DBN model from datasets, the capability and power of DBNs have not

been fully exploited, and DBN remains the most popular to use for reasoning about cause-effect relationships over time. Modelling requirements and various assumptions adopted by the varieties of the existing DBNs are strongly domain-dependent, such as stationarity assumptions (repeating same network knowledge over time). This is mostly handled by experts. Thus, domain-independent DBN modelling approaches that can learn completely and directly from any environment in the absence of domain experts still remain an open challenge. Several modelling approaches that have been proposed in the past, and which we have presented in this chapter, are major motivations to classify DBNs into three well-defined research niche areas shown in Figure 5.1.

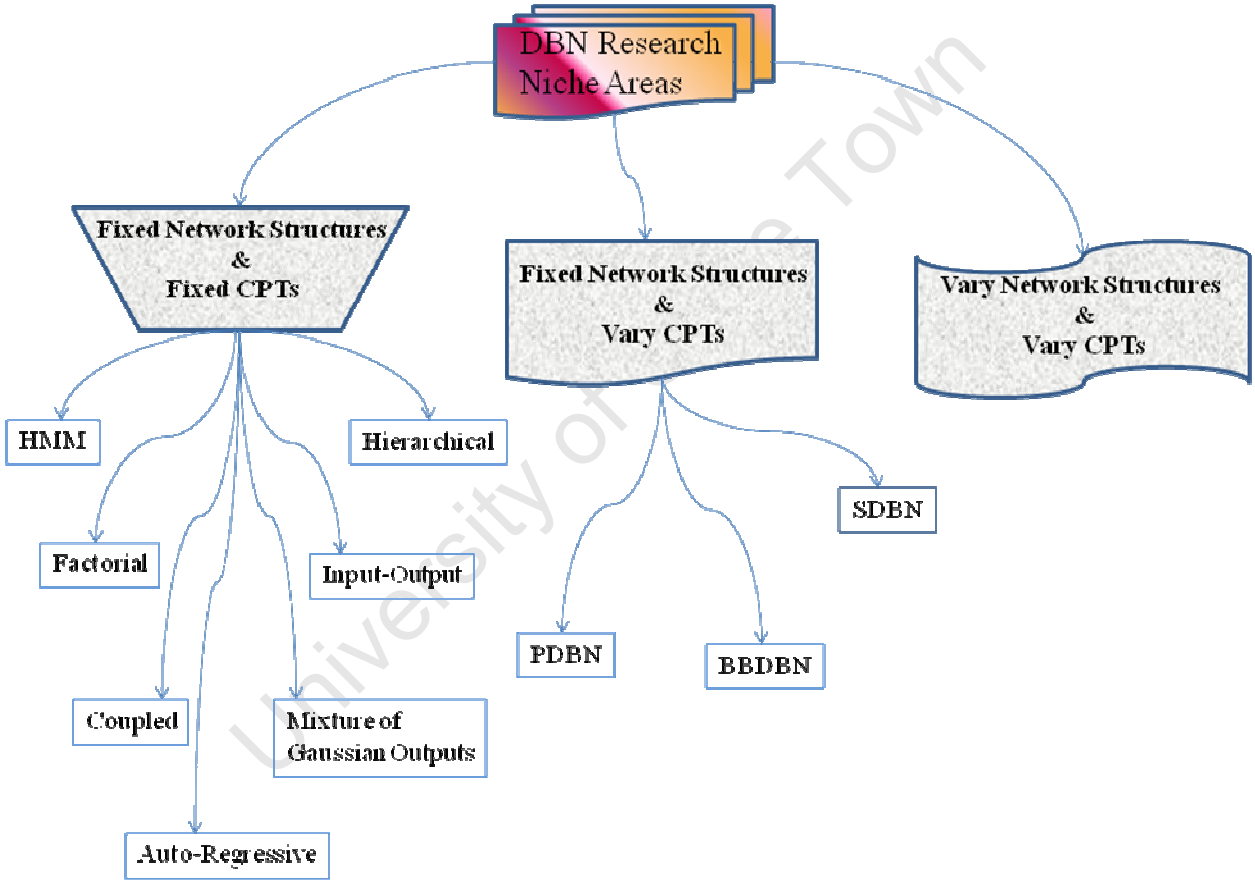


Figure 5.1: Three Well-Defined DBN Research Niche Areas.

From Figure 5.1, the first niche area develops fixed networks and fixed CPTs, which varies its probabilistic process with some defined mathematical and statistical functions [8] [67] such as Poisson and Gaussian distributions, etc. The parameters of these functions can be estimated from the datasets by intelligent systems engineers. It is worth noting that these functions have their known behaviour assuming

situation in a domain of interest is smooth or in a controlled environment, but environmental domains are not always smooth in real life. The second niche area develops fixed networks but learns and sometimes varies its CPTs directly from datasets. The third niche area is expected to learn both network structures and their associated CPTs directly from datasets and reflects change in knowledge as the environment varies. This relaxes the rigidity of niche areas one and two. We do not know any DBN models in literature for the third niche area. Since the first and second niche areas as shown in Figure 5.1 have their own drawbacks, such as approximated knowledge and the problem of choosing which DBN models are suitable for application by non-experts, in this chapter we propose for the third niche area, which has not received adequate attention, a brief overview of a sophisticated architecture called the EDBN. Part of this architecture has been applied to wider real-life areas such as business intelligence [68], project profitability analysis [69], etc. The EDBN integrates two orthogonal deliberative and reactive frameworks – temporal probabilistic modelling and DMMAL. The first framework addresses the challenges above, as it models DBNs completely and directly from MTS in the absence of domain experts. It uses collective knowledge to create awareness about hidden situations embedded in environments and guides users on timely and effective decision-making. Dynamic memory management in adaptive learning (DMMAL) is an economical scalable framework for Bayesian learning. Since researchers and practitioners have, however, stressed that learning (or emerging) such models from massive datasets is computationally intensive [20] [21], intelligent system engineers do not want to wait too long to make the reasoning component ready for use. The major contributions of this chapter are as follows:

- Development of the new EDBN architecture to resolve two orthogonal issues using temporal probabilistic modelling and economical scalable Bayesian learning frameworks.

- Derivation and presentation of new class of DBNs achieving situation awareness technologies with minimal technicalities to accommodate skilled users and non-expert practitioners.

- A new way of presenting three well-defined DBN research niche areas and rigorous evaluation of the EDBN modelling with other popular DBN models.

The rest of this chapter is organized as follows. In Section 5.2, we present the EDBN architecture. The rationales behind the temporal probabilistic framework are presented in Section 5.3. In Section 5.4, the rudiments surrounding the economic scalable framework of Bayesian learning, which include an agent architecture and dynamic memory management scheme, are presented. Section 5.5 describes the comparisons between the EDBN models and other popular DBN models. This chapter concludes in Section 5.6.

5.2 The EDBN Architecture

A brief overview of the proposed EDBN architecture is given in this section, leaving out individual technical details of algorithms from this chapter to accommodate the understanding of all readers. The architecture of practical intelligence which integrates two orthogonal deliberative and reactive frameworks (temporal probabilistic modelling and DMMAL respectively) is shown in Figure 5.2. The ideas in these frameworks are new modelling approaches with respect to the literature.

The first part, illustrated at the top of Figure 5.2, is the temporal probabilistic modelling framework which develops two new technologies – Emergent Situation Awareness (ESA) and Emergent Future Situation Awareness (EFSA). The ESA technology monitors any complex environment by predicting and revealing hidden situations over current time steps. The EFSA, a variant of the ESA, projects the environment into the future when current situations are well understood. These technologies have been demonstrated to be effective when the ESA monitors complex environments, such as: (i) the assessment of rainfall onsets in complex weather patterns to determine farmers' planting dates [70], and (ii) finding drivable routes for autonomous robotic vehicles [90]. In the same way, these technologies were presented in [71] where the EFSA was used to project the measurement of regular insulin doses that would be required by a patient for the next 12 months.

The descriptions in the following paragraphs are involved in applying the architecture. In the absence of domain experts, the temporal probabilistic modelling framework captures any domain of complex environment as MTS. An MTS is a set of observations recorded at specified time intervals about the attributes describing a domain of interest. There is essential transitional knowledge embedded in the MTS as the descriptions (or causal relationships) of the attributes differ over time. This is the environment containing datasets, where our DBNs evolve over time t_n. The time interval could be yearly, monthly, daily and hourly. The framework marshals the MTS into frames t_0 to t_n without changing the originality of the data. By creating awareness of situations in the MTS, temporal frame models are evolved and are interlinked as a DBN using learning algorithms, as shown in the framework. If any frame *t* contains massive dataset, the DMMAL framework at the lower part of Figure 5.2 is activated and the massive frame is selected for modelling using the scalable algorithms. DMMAL returns the emerged frame model back to the frame number of the temporal framework.

Observe in Figure 5.2 that DMMAL is not tightly coupled together with the temporal modelling framework due to reusability of frameworks. One important advantage of this architecture is the independence and/or dependence of its two frameworks in terms of operations. This implies that existing Bayesian learning algorithms can use DMMAL to mitigate their computational intensity (or NP-hard)

problems without any interference with the temporal probabilistic modelling framework. The frameworks are further emphasized better in sections 5.3 and 5.4.

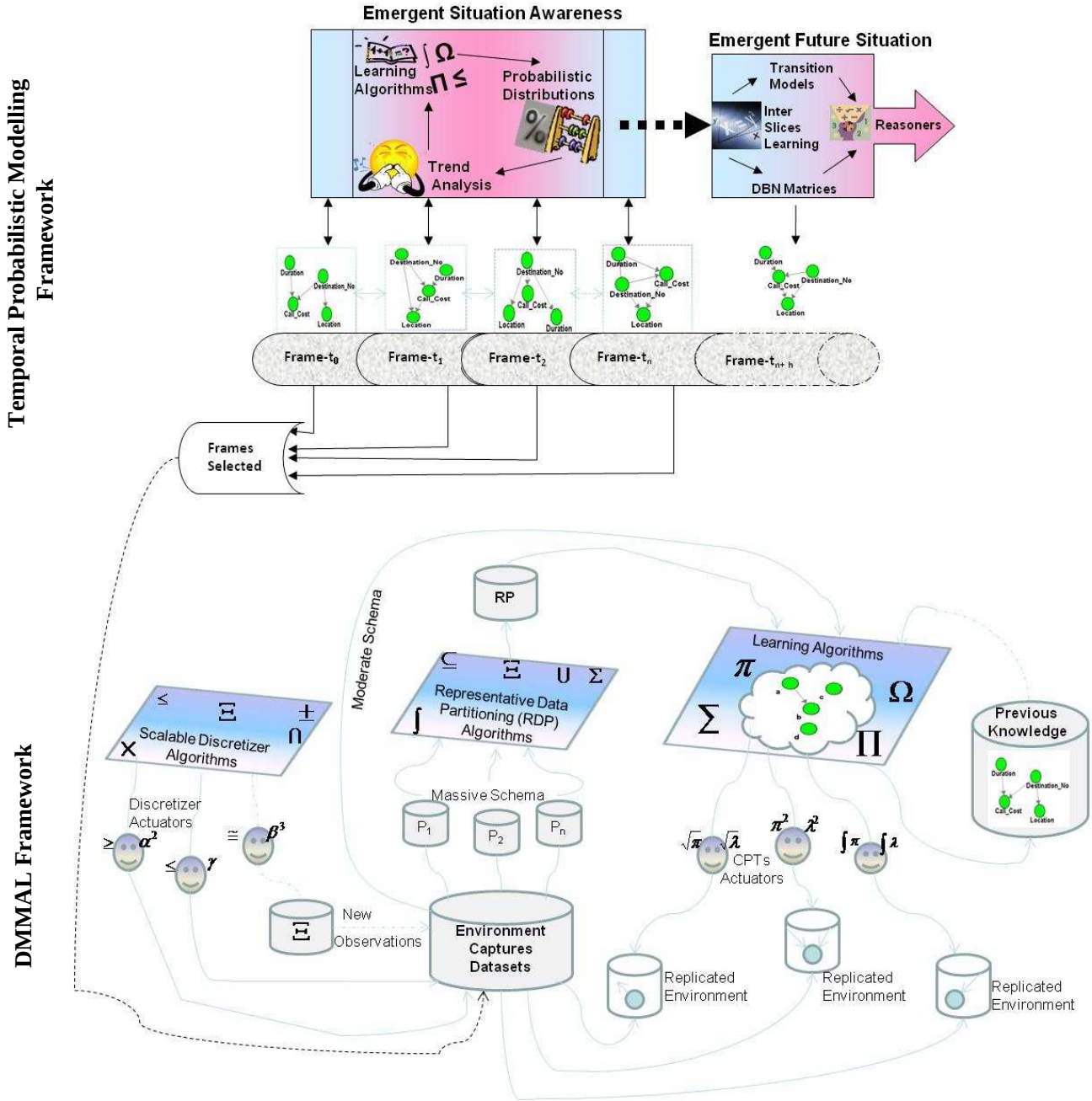


Figure 5.2: The EDBN Architecture

5.3 The Temporal Probabilistic Modelling Framework

We believe that an environmental event captured in multivariate time series (MTS) continually changes in unforeseen situations differently from how some other situations change another set of environmental

events. For instance, in retail, situations driving sales of toiletries are definitely different from the driving situations of beverages. These hidden and multifold situations, which are continually changing the events, mingle together as they are embedded in the MTS. The basic idea here is to clarify and understand the situations well in order to facilitate efficient decision-making and drastically reduce possible risks. Achieving this will enable decision-makers to carry out anticipatory planning for good management strategies.

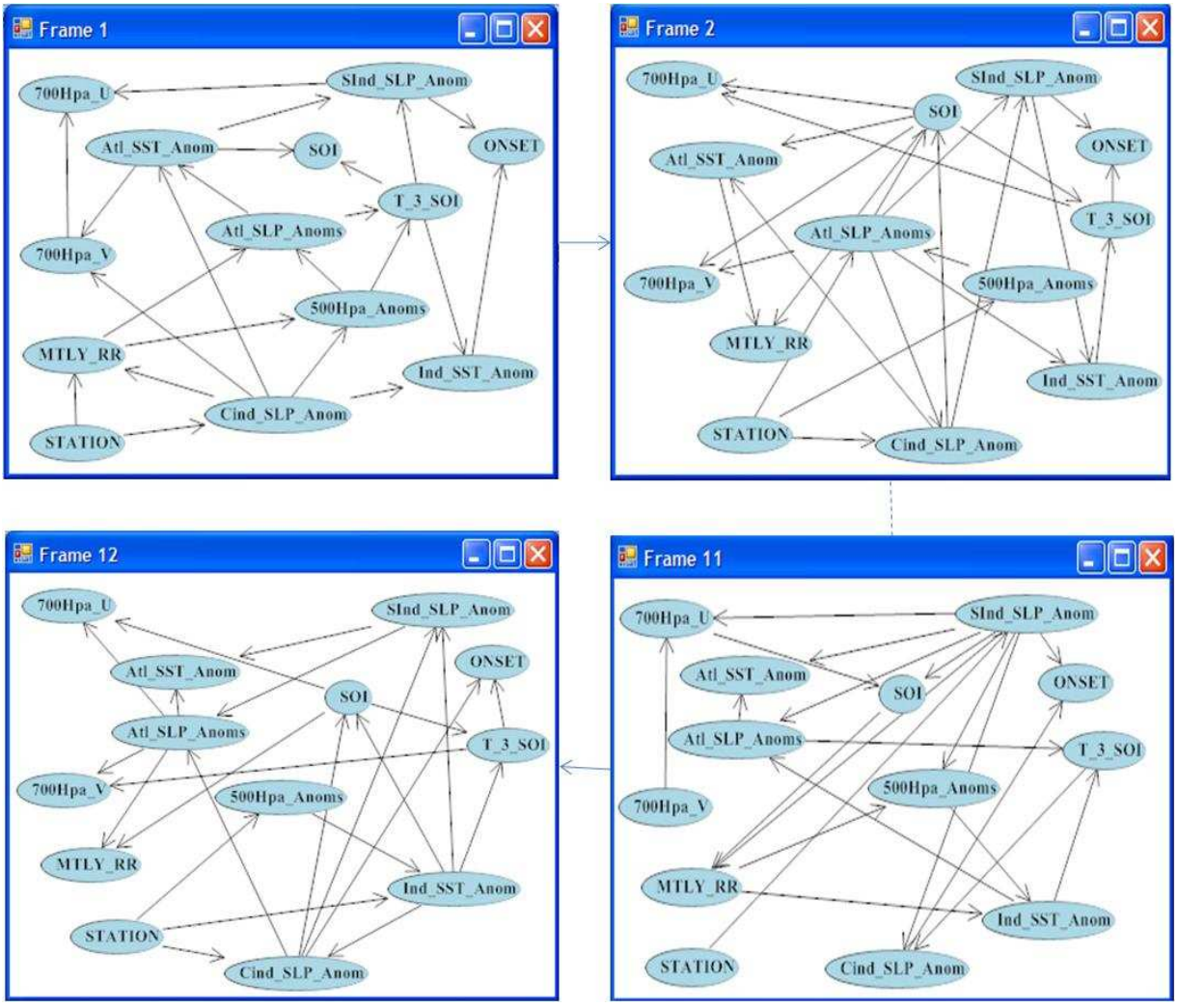


Figure 5.3: An Example of Temporal Probabilistic Model (or Global Dynamics) for Botswana Rainfall Distribution over Time Steps January (or Frame 1) to December (Frame 12) for a Current Year 2000 using the EDBN Architecture.

Since a DBN that is first evolved from an MTS is a temporal probabilistic model or global dynamics of knowledge, the hidden situations revealed from it is the local dynamics, which gives the smallest pieces of information required to understand the environments. Consider a rainfall distribution described by

weather and climate attributes, which is captured locally as an MTS from the Botswana environment. It is used by the EDBN to reveal the onsets of rainfall to determine farmers' planting dates. Figures 5.3 and 5.4 are examples of the global and local dynamics (a revealed hidden situation from the complex rainfall environment) respectively. Some of these attributes presented as nodes on the model in Figure 5.3 are: Atl_SST_Anom - Atlantic Ocean Sea Surface Temperature Anomalies (^{0}C), Ind_SST_Anom -Indian Ocean Sea Surface Temperature Anomalies (^{0}C), Mthly_RR - Monthly Rainfall (in mm), Onset types - false, early, normal, late or failed, Station of rainfall observations, etc.

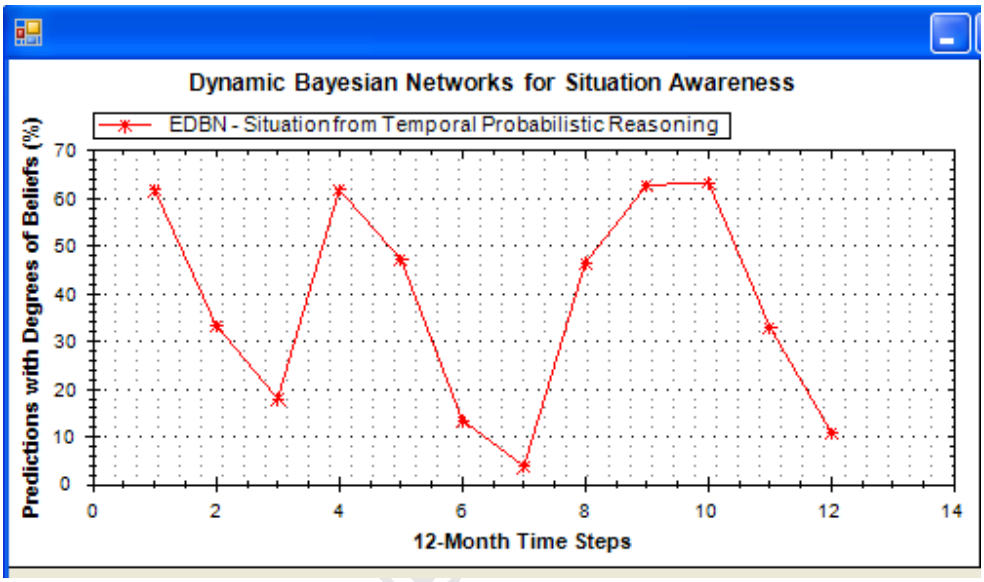


Figure 5.4: A Revealed Hidden Situation (or local dynamics) for normal onsets of the Rainfall Distribution for a Current Year 2000 at Station 69, using the EDBN Architecture. The situational awareness result of this normal onset is conditioned on parameters: Atl_SST_Anom and Ind_SST_Anom.

For the purpose of interpreting the complex relationships within the attributes of the model, an example of possible situations affecting the ever-changing *normal* onset event in the model is first predicted over current time steps, as shown in Figure 5.4. The temporal pattern of this situation shows that there are irregular ***normal*** onsets in the year 2000 which may not be an excellent signal for farmers' planting dates. It is obvious that good anticipatory planning is associated with a detailed understanding of situations that are currently occurring in one's domain of interest. By first understanding and separating current situations from any anticipation, there is a good contribution to the efficiency of the temporal probabilistic framework. Having established the current situations of events, their future behaviour is then projected to assess the entire behaviour of such complex environment. Before describing the second framework of the

EDBN, the ESA and the EFSA technologies derived from the temporal probabilistic framework are first introduced with minimal technicalities.

The ESA is an innovative technology, which completely evolves temporal models and reveals the hidden behaviour of what is currently happening over time in any domain of interest. One of its powerful strategies is its emergence from multivariate time series (MTS) data in the absence of domain experts. Formally, let $\{V^t, E^t\}$ represent the set of state and observed DBN variables in ESA at time *t.* The DBNs are emerged over all the non-negative current time steps $t \in T$, such that $T = \{t_1, t_2 \ldots t_n\}$ and the interlinked probabilistic relationships at each time step *t* is shown as an example in Figure 5.3. ESA extends the algorithms of ordinary Bayesian networks to truly evolve dynamically as it changes its network and the probabilistic distributions with time. The part of the temporal probabilistic framework shown in Figure 5.2 has three components: the learning algorithms, the probabilistic reasoner and the trend analyzer. The learning component uses genetic algorithms [51] to evolve temporal Bayesian Network models, called frames, over the time steps from the MTS environments. A number of other BN learning algorithms such as [23] [38] can also be adopted herein, as long as they can evolve dynamic models across the time steps. The probabilistic reasoner is the Bayesian inference engine, which handles the necessary possible forward and backward propagations through the links of the frames and generates probable results through collective intelligence. The trend analyzer is an interface engine that generates *n*-dimensional transition matrices of knowledge, where *n* corresponds to the pieces of hidden knowledge to be revealed, e.g. a transition matrix of target probabilities of situations, a transition matrix of target parameter values of events, etc.

Once the ESA technology has automatically modelled, revealed, and guided users to being well acquainted with the current situations of a complex environment, the problems of how to choose the most suitable model for specific real-life applications, especially by non-expert practitioners, are almost solved. The EFSA completes the process of finding a solution as it takes this technology further, projecting the situations of the environment into the future and avoiding convergence problems when predicting over wider time steps. Its prediction strategy is based on the automatic emergence of temporal models which spans two-dimensional (2D) orthogonal time space from historical Multivariate Time Series (MTS), using some mathematical and algorithmic developments not presented in this chapter. However, models that are to be evolved from massive datasets are sent to the DMMAL framework for scalability.

5.4 An Economic Scalable Framework of Bayesian Learning

This section presents the basic agent architecture and the memory management scheme as part of the building blocks of our economic DMMAL (dynamic memory management in adaptive learning) framework for scalable Bayesian learning.

Among the classes of agents used in intelligent systems, the software agent, as related to this work, perceives from the *environment* through *sensors* and acts upon the environment through *actuators* [8]. The fundamental components of any agent, as highlighted above, are illustrated in the basic architecture shown in Figure 5.5.

5.4.1 A Basic Agent Architecture: DMMAL Strategy

According to Russell et al. [8], a software agent can sense its environment using file content or network packets and also uses writing files or packets as actuators to act on the environment. Figure 5.5 shows that when environment is perceived, some forms of machine learning algorithms (e.g. MLE) are used to interpret the percept. They consequently generate the instructions required by the actuators to carry out actions on the environment.

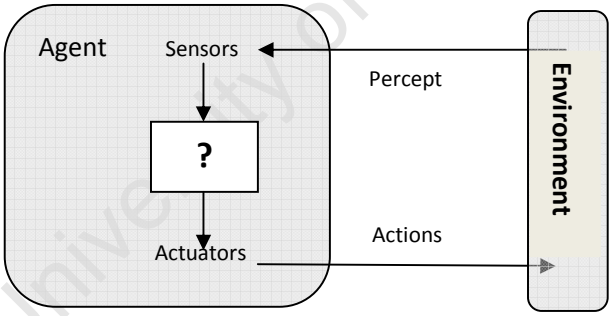


Figure 5.5: A Basic Agent Architecture [5]

Observe that the separate positions of the agent and the environment possess some distributed properties. It illustrates that agents can be sent over a network to carry out specific tasks and can provide services to other components (e.g. storage environment) on a given machine. It is deduced from here that CPT agent actuators, as described in the next section, can be characterized with mobility as they include their required information in their description. Their independence influences the design of components for distributed agents which motivate the design of the CPT agent actuators in this study. The next subsection describes the memory management scheme adopted in a DMMAL framework.

5.4.2 Dynamic Memory Management Scheme: DMMAL Strategy

Our framework borrows its dynamic memory management scheme from the loci framework [29]. It is adopted to address the out-of-memory problems of learning the networks from massive datasets. It is known that datasets are stored as data structures, but when they are large, they are the major source of memory consumption. The memory is also totally consumed during execution (e.g. learning of models from massive datasets) when memory used varies in an unpredictable way [72]. Therefore, this scheme is an economical solution which manages the memory by allocation and de-allocation of data structures based on the lifetime of data structures. Thus, in order to accommodate learning models from large datasets within a limited memory, we extracted and interpreted the scheme from [29] [72] as follows:

The Scheme: *Memory Management*

[I] Pre-allocation of memory to data structures
[II] Incorporate relevant memory management operations
[III] Invoke loop scheduling techniques
[IV] Recycle memory from data structures

Pre-allocation with partitioning of the entire memory alone in Step I does not benefit space saving until the other steps in the sequence are involved. Many parallel algorithms exploit Step I as a trade-off that optimizes speed but which suffers from peak memory requirement. A possible relevant management operation in Step II uses remote memory or secondary storage devices, for example. These management operations are generalized concepts of virtual memory. A virtual memory is a multilevel store which gives a large process an impression that it has more primary memory to itself, while it actually uses external disk devices as a supplement [72]. Examples of loop scheduling technique constituents in Step III are multiple nested iterations, recursions, synchronizations, etc. Also in Step IV, at the end of every schedule or lifetime, memory is recovered from data structures after its execution. Thus, this scheme empowers a system to accommodate massive datasets within a limited memory without a halting problem.

5.4.3 The Descriptive Approach of DMMAL Framework

We study the computational intensity problems (memory failure and slow execution) of learning Bayesian network models from massive datasets. As confirmed in this research, it is well known that the intensity is a result of exponential time to learn the network structures and train the structures to capture knowledge

as CPTs. Most of the previous approaches in the literature such as the predominant discretization, the incremental learning and the use of ADtree [30], which have attempted to address the intensity challenges, do not guarantee complete (memory and time) scalability, and the distributed solution on networked machines is expensive for single users. In the Bayesian learning research of this chapter, the challenge was to achieve economic and complete scalability to alleviate the intensity problems. We now present the DMMAL framework that addresses the underlying challenges, as illustrated at the bottom part of Figure 5.2.

The DMMAL framework was engineered and has the following principal components: representative data partitioning (RDP) algorithms, genetic algorithms (GA), configurable CPT actuators and the adapter operator. They are used to evolve optimal Bayesian networks faster and safely. The mathematical and algorithmic developments of these components are not presented in this chapter. Figure 5.6 provides a high-level description of the main components involved in the DMMAL framework at the bottom part of Figure 5.2.

INPUT: **(D_s: Dataset Schema)**

---

1. While $\mathbf{D_s}$ = *new-schema*
   i) If D_{sj} = *Numeric,* for j = 0, 1, 2. . . m.
      $\mathbf{D_s'}$**:=** Discretizer-Agent-Actuators($\mathbf{D_s}$)
   ii) Generate Rp($\vec{y}$) := RDP(D_s, $\mathbf{D_s'}$)
   iii) Emerge **G** := Genetic-Algorithm(Rp($\vec{y}$))
   iv) $\mathbf{B_n}$ **=** CPTs-Agent-Actuators**(G,** D_s, $\mathbf{D_s'}$**)**
2. Else
   i) Load previous parameters, B_o
   ii) Emerge $\mathbf{B_n}$:= Adapter-operator(B_o, d_s)
3. Return emerged Bayesian network, $\mathbf{B_n}$

Figure 5.6: General Outlines for DMMAL Framework

The DMMAL framework starts by learning a Bayesian network, $\mathbf{B_n}$ or a frame model with input dataset, $\mathbf{D_s}$, which may be a massive schema captured from environments. If $\mathbf{D_s}$ is a new schema, DMMAL executes Steps 1 and 3 in Figure 5.6, otherwise, Steps 2 and 3 are executed. With respect to users' requirements, the scalable discretizer algorithms [73] in Figure 5.2 pre-process $\mathbf{D_s}$ to generate learning set $\mathbf{D_s'}$**.** They concurrently distribute discretization processes by classifying numerical data into its corresponding interval values relative to the patterns in the data attributes. To shed load from limited memory without loss of information, a representative partition of $\mathbf{D_s}$, or $\mathbf{D_s'}$ if discretized, is generated as

Rp($\vec{y}$) using a representative partitioning algorithm. A schema $\mathbf{D_s}$ could otherwise be equivalent to Rp($\vec{y}$) and be used for learning, if moderate in size. The optimal network structure **G**, of the $\mathbf{B_n}$ is emerged from the Rp($\vec{y}$) using genetic algorithms such as [51]. In order to achieve complete scalability, the configurable CPT actuators dynamically manage the limited memory and concurrently train G by optionally using $\mathbf{D_s}$, $\mathbf{D_s'}$, or Rp($\vec{y}$), depending on the capacity of the environments.

For new observations in schema, the previous parameters such as old Bayesian network, B_o, which may include previous discretized parameters, are loaded to update new knowledge using adaptive operators. Thus, the optimal $\mathbf{B_n}$ reliably emerges from the massive schema $\mathbf{D_s}$ using the DMMAL framework.

5.5 Comparing the New EDBN Modelling Architecture with Other Popular DBN Models

The objective here is to compare some very favourable benefits of our EDBN modelling approach with those of other popular DBN models. The resulting characteristics depicted in Table 5.1 are a summary of our experience concerning existing DBN models in literature and our practical knowledge of the EDBN models in terms of network structures, probability distribution, complexity, expressive power and our inferences drawn from all of these.

In all the cases, the resulting features reveal that our EDBN model potentially varies its knowledge completely with respect to changes in the environment, and with no restrictions whatsoever in the modelling approach. Its capability of learning models directly from MTS makes it ready for use by experts and accommodates non-expert practitioners, while the other models suffer from approximated knowledge by exhibiting network 'remains fixed' and/or a probability distribution of 'variability not always true'. For convenient modelling, they exhibit a stationarity dynamics but miss out possible useful information. The 'variability not always true' is a fixed CPT using a probabilistic process with some defined mathematical and statistical functions, such as Poisson and Gaussian distributions, etc. Once again, it is worth noting that these functions have their known behaviour assuming all things being equal, but things are not always equal in real life. That is, these functions perform well in smooth or controlled environments but real life situation is not smooth. Their domain rules of modelling are the restricted assumptions which make them challenging to represent and they can best be modelled by skilled users.

Observe in Table 5.1 that the variability of the EDBN models and their capability to learn directly from MTS makes it possible for them to be utilized/applied in the absence of domain experts. These

resulting features suggest that using the EDBN architecture to model DBNs will enable users to capture the ever-changing complex environments adequately.

Table 5.1: A Summary of Comparisons between the EDBN Models and Other Popular Models

DBN Models	**Network Structures (over time)**	**Probability Distributions (over time)**	**Model Complexity**	**Expressive Power**	**Our Inference**
EDBN	Varies	Varies	Learn completely from datasets	No restrictions	For both skilled and non-expert users
HMM	Remains fixed	Variability not always true	Not difficult to represent	Very restricted with assumptions	For skilled users
Auto-Regressive	Remains fixed	Variability not always true	Not difficult to represent	Very restricted with assumptions	For skilled users
Input-Output	Remains fixed	Variability not always true	Not difficult to represent	Restricted with assumptions	For skilled users
Factorial	Remains fixed	Variability not always true	Challenging to represent	Restricted with assumptions	For skilled users
Coupled	Remains fixed	Variability not always true	Challenging to represent	Restricted with assumptions	For highly skilled users
Mixture of Gaussian Outputs	Remains fixed	Variability not always true	Challenging to represent	Restricted with assumptions	For highly skilled users
Hierarchical	Remains fixed	Variability not always true	Challenging to represent	Restricted with assumptions	For highly skilled users
PDBN	Remains fixed	Varies	Challenging to represent	Restricted with assumptions	For skilled users
BBDBN	Remains fixed	Varies	Not difficult to represent	Very restricted with assumptions	For skilled users
SDBN	Remains fixed	Varies	Challenging to represent	Restricted with assumptions	For skilled users
Intrusion detection DBN [15]	Remains fixed	Variability not always true	Not difficult to represent	Restricted with assumptions	For skilled users
Activity recognition DBN [11]	Remains fixed	Variability not always true	Not difficult to represent	Restricted with assumptions	For skilled users
Air Combat DBN [12]	Remains fixed	Variability not always true	Challenging to represent	Restricted with assumptions	For highly skilled users
Click model DBN [10]	Remains fixed	Varies	Challenging to represent	Very restricted with assumptions	For highly skilled users

5.6 Conclusions

We have proposed in this chapter the development of EDBN architecture for temporal probabilistic modelling from MTS, and as an alternative optimization solution to the computational intensity problems arising in intelligent systems. The temporal probabilistic modelling framework derived the ESA and EFSA technologies, where the ESA monitors complex environments over current time steps and the EFSA projects the behaviour of the environments into the future. The theoretical descriptions potentially showed that DBNs can now be evolved in the absence of domain experts and that the use of DMMAL is an economically scalable solution to the problem, as it does not require purchasing expensive hardware.

Our architecture was rigorously subjected to a number of evaluations as it relaxes the rigidity of existing models and accommodates both skilled and non-expert practitioners. This study shows that the EDBN has the potential to become a more powerful deliberative and reactive architecture that finally puts an end to the worries of non-expert practitioners in choosing the appropriate model among DBN types and also ends the computational problems raised in various research efforts. If Bayesian model researchers and the intelligent system engineers can integrate either or both of the frameworks in EDBN into their developments, there will be more contributive solutions to mitigate computational intensity, and accommodate skilled and non-experts of DBNs. This greatly eases anticipatory planning for good management strategies.

Chapter Six

Emergent Situation Awareness (ESA) Technology

– Understanding Complex Systems

6.1 Introduction

Situation Awareness (SA) is becoming popular among decision-makers to a notable extent. SA has gained its popularity in, for example, the areas of air traffic control, emergency responders and surgical teams [16]. Instances of application areas where taking correct rapid-response decisions is needed are disaster management, business intelligence, robotics, even sport (e.g. robotic soccer often called Robocop) where players have to make instant decisions in a constantly changing environment. Most notably, there is an ongoing demand for SA technologies and their variants in environmental water management problems, business problems, areas presenting human health risks, and robotic agents [8] [74] [75] [76]. Each of these problem areas is complex to understand due to the uncertainties embedded therein, but Bayesian Network (BN) models are effective in handling complexities.

Bayesian Networks (BNs) are probabilistic models which are gaining popularity in decision-making. The major shortcomings of their current implementations include the inaccurate complex modelling despite expert intervention and the absence of complete temporal pattern modelling capabilities. The available DBNs (Dynamic Bayesian Networks) with temporal modeling, such as Factorial HMM (Hidden Markov Models) [6], Coupled HMM [5], Input-Output HMM [4], and PDBN (partial Dynamic Bayesian Network) [36] have contributed to temporal modelling up to the baseline but they are explicitly represented by skilled users, therefore are limited in their expressive power. Decision-makers, such as non-expert practitioners, struggle to interpret them in order to take correct action. This has consequently led to ongoing information gap problems (e.g. between complex models and correct interpretation by decision-makers) [16]. The information gap problem is a result of decision-makers not being well acquainted with the patterns currently occurring in their various domains. Bridging this gap is a challenge and the difference between poor and good decision-makers lies in their pattern understanding or SA ability.

In this chapter, we achieve emergent situational awareness by evolving actual local dynamics from global emergent behaviours. The local dynamics are the smallest pieces of information needed for making correct decisions easily, interactively and economically. For instance, declaring discounts on a

number of selected, rather than all, manufactured products within a space of time, is economical. This chapter aims to empower decision-makers using the new ESA technology to make the best possible decision from any recognized pattern at any point in time. Our ESA technology completely evolves DBNs (changing network structures and CPTs) from MTS (multivariate time series) in the absence of domain experts; views knowledge as situational patterns over time; and provides a suitable platform to guide decision-making processes. We do not know of any existing DBN implementations that completely evolve Bayesian networks dynamically but rather, they are approximated and they do not guarantee true response to evolving current situations of the real life application domains. The major contributions in this chapter are as follows:

- The theoretical development of the ESA technology, which establishes a new class of DBN models and guides on decision-making processes.

- The development of the ESA algorithms facilitates direct learning of DBN models from environments and reasoning about current situations over time without expert interventions.

- Achieving wider real-life applicability such as water quality management and business intelligence, other than the popular speech recognition areas, using our new DBN technology.

The rest of this chapter is arranged as follows: in section 6.2, we present the theoretical development of the ESA as a new class of DBN which includes reasoning within these temporal models. The theoretical description to apply the ESA technology, which includes a system model extracted from our EDBN architecture (not presented in this chapter) and ESA algorithms, is presented in section 6.3. Section 6.4 rigorously presents various experimental applications of the ESA in environmental water quality management and business intelligence. We conclude the chapter in section 6.5.

6.2 Theoretical Developments of the ESA

This section formally presents the theoretical developments of the ESA technology from the third research niche area of DBNs.

6.2.1 Theoretical Derivation of Research Niche Areas on Dynamic Bayesian Networks

The joint probability density distribution of a Bayesian Network for discrete random variables $X_1, \ldots, X_n$ with each having a set of some values $x_1, \ldots, x_n$, is defined in equation (6.1).

$$pr(X_1,...,X_n) = \prod_{i=0}^{n} pr(X_i \mid \pi(X_i)) \tag{6.1}$$

where $\pi(X_i)$ represents a set of probabilistic parent(s) of child X_i [37]. Let V_i^t represent DBN variables at time *t*; we derive the following equation (6.2) from equation (6.1), over all the non-negative time steps *t* ϵ *T*.

$$pr(V_1^t, V_2^t ..., V_n^t) = \prod_{i=0}^{n} pr(V_i^t \mid \pi(V_i^t)) \qquad \forall \text{ t } \epsilon \text{ T} \tag{6.2}$$

The system of equation (6.2) forms the temporal dependency relations between the time slices for the proposed ESA, which generates a matrix of transition knowledge embedded in the environments captured as MTS. Using the concept in this section, coupled with the various DBN reviews in literature such as the examples at the introduction of this chapter, we therefore derive three well-defined research niche areas of DBNs as equations (6.3) – (6.5).

$$pr(V_1^1, V_2^1 ..., V_n^1) = pr(V_1^2, V_2^2 ..., V_n^2) = pr(V_1^t, V_2^t ..., V_n^t) \tag{6.3}$$

In equation (6.3), the probabilistic distribution (CPTs) and the DAGs (or network structures) do not vary, but the representation of the probabilistic process changes using some mathematical distributions such as Gaussian or exponential as fitness functions. This is otherwise referred to as stationary dynamics, and most DBNs based on HMM representations fall into this class. Examples are HMMs with Mixture of Gaussian Outputs, Input-Output HMM, etc. [3].

$$pr(V_1^1, V_2^1 ..., V_n^1) \approx pr(V_1^2, V_2^2 ..., V_n^2) \approx pr(V_1^t, V_2^t ..., V_n^t) \tag{6.4}$$

Equation (6.4) is a DBN that varies its probabilistic distributions (or CPTs) but its DAG remains fixed. They sometimes learn the CPTs from datasets to capture the variability embedded therein. Examples are PDBN (partial Dynamic Bayesian Network) [36], SDBN (structural Dynamic Bayesian Network) [7], etc.

$$pr(V_1^1, V_2^1 ..., V_n^1) \overset{\Delta}{\equiv} pr(V_1^2, V_2^2 ..., V_n^2) \overset{\Delta}{\equiv} pr(V_1^t, V_2^t ..., V_n^t) \tag{6.5}$$

$\overset{\Delta}{\equiv}$ implies equivalence is *not* true generally.

Equation (6.5) is a DBN that varies both its probabilistic distributions and its temporal DAGs by learning directly from MTS. This is the most challenging niche area that has not received adequate attention because the uncertainties in the DBNs involve a large number of relationships between the attributes. The relationships here are of greater value to situation awareness, as presented in this chapter, because hidden patterns are revealed over time. This implies that the local dynamics are evolved from the global behaviour. Existing temporal modelling efforts have been approximating their DBNs for convenience.

6.2.2 Theoretical Illustration of Reasoning in ESA Technology

Probability distribution is an important component of the ESA technology. Bayesian inference is well fitted and integrated into the ESA since it is capable of projecting the next likely event(s). It reasons over the interconnections of the system of equation (6.2), and the projected likely events over time are generated as a transition matrix of knowledge. Bayesian inference algorithms are characterized by information propagation (forward and backward), through the entire network and are therefore good for reasoning purposes. The best approach for illustrating Bayesian inference is to use a sample network.

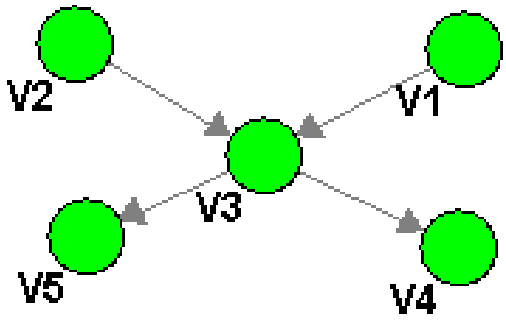


Figure 6.1: A Sampled Bayesian Network Illustrating Reasoning

Suppose a Bayesian Network is represented with random variables $\{V_1, \ldots, V_5\}$ as shown in Figure 6.1, then equation (6.6) is a variable elimination algorithm which works well with regard to its Bayesian inference used for reasoning. Variable elimination has been certified by Russell et al. [8] as suitable for DBNs. It is thus presented in equation (6.6) as it investigates the probability of V_1 given V_4 and V_5. We therefore derived equation (6.8) from (6.6) for reasoning over the system of equations (6.2). We will first represent similar Bayesian inference to equation (6.6) for DBN at any time step *t*, as shown in equation (6.7).

$$pr(V_1 \mid V_4, V_5) = \alpha pr(V_1) \times \sum_{V_2} pr(V_2) \sum_{V_3} pr(V_3 \mid V_1, V_2)\, pr(V_4 \mid V_3)\, pr(V_5 \mid V_3) \tag{6.6}$$

$$\rho^t = pr(V_\ell^t \mid V_i^t, V_{i \pm j}^t), \quad \forall\, t \in T, \tag{6.7}$$

Likewise, from basic probabilistic calculus, the probability of an event happening at random in an experiment is known as the probability that the event is likely to occur at the *next* experiment [8] [67]. Such an experiment being repeated in this study follows situations happening over the time steps. We can similarly say that the probability of the event happening at random in the next time step *t*, is likely to follow the knowledge in the previous time step *t -1*. Consequently, and as a relation to equation (6.6), we equate equation (6.7) to equation (6.2), which therefore gives equation (6.8). This is used for propagating reasoning over all the time steps depending on the temporal BN, evolved dynamically.

$$pr(V_\ell^t \mid V_i^t, V_{i \pm j}^t) = \alpha \sum_{V_\delta} \sum_{V_\gamma} pr(V_\ell^t) \prod_{i=0}^{n} pr(V_i^t \mid \pi(V_i^t)) \tag{6.8}$$

ℓ identifies the target variable, while *i* and *j* identify the evidence variables. V_δ and V_γ are the variables without parents evolved with time. Equation (6.8) is therefore an important component in the ESA to generate the transition of knowledge. This reveals in-depth awareness of patterns for current situations over time and gives insight into the next likely event. As a proof of concept for our ESA, we integrated the implemented variable elimination algorithm of JavaBayes [77]. It is open-source software. Having presented the mathematical developments, the next section describes the derived ESA paradigm.

6.3 Development of ESA Components

Besides our mathematical analysis in the previous sections, this section presents the ESA paradigm. With reference to the SA (situation awareness) theory [16], which describes a mental model at three hierarchical levels, overall SA can be defined as shown in equation (6.9).

$$\text{OSA} = e_3\ [+ e_4 + e_5] \tag{6.9}$$

$$\text{where} \quad e_3 = g\,(f\,(e_1), f\,(e_2))$$

$\boldsymbol{e_3}$ is the *active* SA component obtained as function g(.) from the integration of system knowledge of function f($\boldsymbol{e_1}$) and interface knowledge of function f($\boldsymbol{e_2}$). It is the crucial aspect of SA that is often prone to

errors, as discussed by [16]. Direct observation from real world, $\boldsymbol{e_4}$, and distant observer, $\boldsymbol{e_5}$, provides other sources of information that may probably affect SA and consequently have an impact on decision-making. To interpret these theories, Figure 6.2 shows and simplifies a system model to achieve $\boldsymbol{e_3}$ using the ESA.

6.3.1 The System Model for the ESA

The system model in Figure 6.2, which comprises three essential components, was extracted from our EDBN architecture, which is beyond the scope of this chapter. They are learning algorithms, probabilistic distributions and the trend analysis. The first two components collectively generate the system knowledge f (e_1). This is integrated into the third component, which is the interface knowledge f (e_2). This is presented in equation (6.9) as it accomplishes e_3, the crucial aspect of SA theory. The e_4 and e_5 are subjective knowledge that may affect decision-making by answering the four 'W' questions of SA: *What is happening? Why is it happening? What can I do about it? What will happen next?* As shown in Figure 6.2, the three components formulate the engine used to reveal specific situations (local dynamics) from frame model (global behaviour) over time, which accurately improves decision-making. There are numerous hidden situational patterns embedded in any DBN. Real-life applications of these situations are presented in the experimental results of section 6.4 as the ESA bridges information gap problems.

In Figure 6.2, the Learning Algorithms dynamically evolve temporal models from multivariate time series (MTS) environments. The time intervals could be weekly, monthly, daily or hourly, observed in the fields of water management, retail, oil production, telecommunications, etc. The MTS is observed as frames and serves as input for learning. The algorithms evolve interlink temporal models from *frames 0* to *n*. The existing learning algorithms, such as hill-climbing and genetic algorithms (GA) [23] [34] [38] used for learning ordinary BNs from datasets, can fit into Figure 6.2, if upgraded to work over time steps. Our optimized GA [51] is upgraded to evolve over time and is used as a proof of concept in this system model. The algorithm uses information-theoretic measures (e.g. Minimum Description Length) and mathematical components (e.g. PowerSet in set theory) as genetic operators and as a means of balancing between efficiency and decomposability. The GA is used due to its efficiency as it performed very well when used to learn models from the environments of numerical, nominal and mixed datasets.

The probabilistic distribution integrated into Figure 6.2 is a Bayesian inference of the Variable elimination algorithm, which is used to reason over time. Finding what is needed when it is needed is the result of a probabilistic reasoning from a time slice about a situation. This is otherwise a state of knowledge as described in the SA theory. The component of trend analysis in Figure 6.2 is an interface that constructs a transition matrix of knowledge over time. The nature of knowledge in the patterns

generated can periodically determine the likely action to be taken on any situation *n* to arrive at (or avoid) the next situation *n+1*.

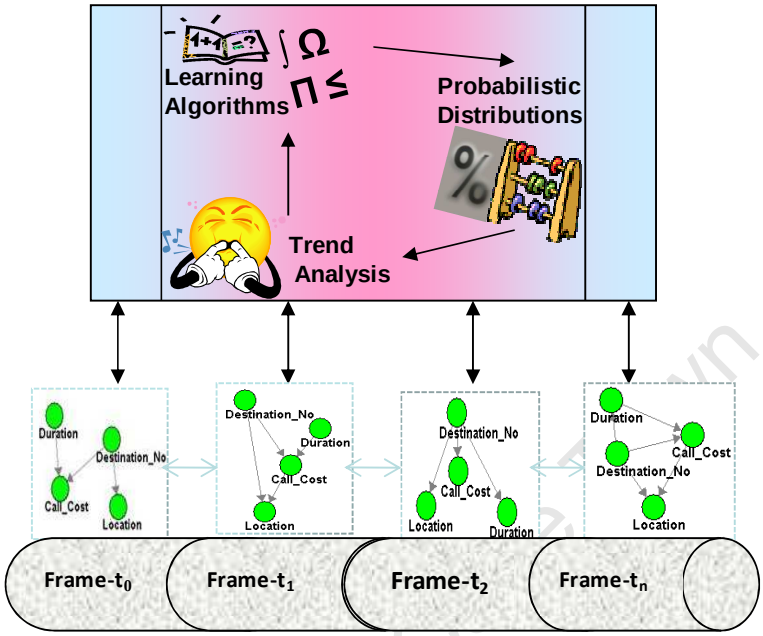


Figure 6.2: A System Model for the ESA

In essence, the three components handshake or interchange their outputs and the current situations are made interactive for real-time decision-making. These are basically used to simplify the answers to the Four 'W' questions about situations. In applying this technology, we formally develop the ESA algorithm as shown in Figure 6.3.

6.3.2 The ESA Algorithm

An MTS serves as the required schema to Figure 6.2 but the additional capability of the ESA algorithm in Figure 6.3 serves to generate MTS from domain datasets without changing their originalities. Its development is based on the theories, algorithms, models and mathematical analysis, which are used as subroutines as described in the previous sections.

In Figure 6.3, the D_{sj} is a column of the schema, d_t is a frame dataset and b_t is a temporal Bayesian Network emerged at time *t*. As shown in step 1[*i*], discretization classifies numerical datasets into their corresponding interval values relative to the patterns in the data attributes [40] [42]. Due to the predominance of computational complexity during data-preprocessing, the ESA introduces scalability into the discretization processes by adopting our DMMAL framework. In this scheme, space is shared and

every used memory is cleared for the next processes. In step 1[*v*], the Bayesian learners are the algorithms that were recently mentioned [23] [25] [51], whose functionalities are to carry out intra-slice learning over time. They evolve temporal optimal BN at each time step. Likewise, the Bayesian inference, as described in equation (6.8), generates several situational trends as a transition matrix of knowledge, which is consequently used by decision-makers.

INPUT (D_s : Dataset Schema)

---

1. While D_s = MTS,
   - [i] If D_{sj} = *Numeric*, for j = 0, 1, 2. . . m.
     - Call Scalable_Discretizer (D_{sj}).
   - [ii] Perform ordering on D_s using t key.
   - [iii] Set t, the frame count, to *0*.
   - [iv] Let $d_t \in D_s$, $\forall$ t = 0, 1, 2, . . ., n.
   - [v] For each $t <= n$,
     - Select frame d_t for emergence.
     - Call Bayesian_Learner (d_t).
     - Store the emerged temporal BN in matrix B.
     - Increment t by 1.
   - [vi] For Situational Trends, Call Probabilistic Distributions, $\forall$ $b_t \in B$.
   - [vii] Return the dynamic BNs in B as the frames' situations, then exit.
2. While D_s <> MTS,
   - [viii] If D_{sj} = *Date*,
     - Select t.
     - Generate MTS from D_s using t.
   - [ix] Repeat step 1.

Figure 6.3: Emergent Situation Awareness Algorithm

Decision-making is made simpler with the ESA, as users can now be well acquainted with their current complex domains before projecting into the future. Most of the existing DBN frameworks focus on projecting only into the future with explicit representations. Since the ESA is domain-independent, it not only accommodates highly skilled users, but also allows non-expert practitioners to benefit from DBNs.

6.4 Experimental Applications and Evaluations

One of the objectives of our ESA technology is to bring theory to practice with an emphasis on applications and practical work. Subsections 6.4.1 – 6.4.3 present extensive experimental applications of the ESA in water quality management, business intelligence and wireless sensor networks respectively.

An empirical way to justify the universality of our ESA technology and to assure that our modelling design is reproducible is to use publicly available datasets to test our theories and implementations. The common publicly available datasets are in the university of California (UCI) repository [31], which most machine learning researchers use in order to experiment their techniques.

6.4.1 Applications in Water Quality Management

A multivariate water quality dataset was used as a typical example of time series observed for a period of one year, which corresponds to four seasons. Each season consists of an average of 45 sample observations taken from sites on different European rivers. These river samples are described with the *season*, *the river size*, and *the fluid velocity*. According to the UCI repository, the quality of the water samples are also analysed using the following chemical substances: *Nitrates, Nitrites, Ammonia, Phosphate, pH, Oxygen* and *Chloride*. The data includes Algae population distribution, but this is not required in this experiment since the details are not provided. This is an integration of chemical and biological processes, which require advanced technology like the ESA.

The ESA focuses on revealing current hidden patterns by understanding the variability of chemical concentrations over the seasons to project the next likely situation. A major concern of water quality managers is to protect rivers and streams from toxic waste and reduce the impact of man on his environment. The toxic wastes are generated from *industrial processes*, *run-off* from farmland activities and from *sewage water treatments*. The pollution of surface and underground water such as lakes and seas results in the reduction of oxygen levels, which consequently leads to environmental hazards such as the death of river fish and poor water quality. We are therefore motivated to use ESA to monitor and understand the variability of the water concentrations using about 200 training datasets supplied and evaluated with 140 test set samples provided.

The training set is thus fed as MTS into the ESA algorithm, using seasons as time steps. The interactive pattern results obtained encourage better real-time decision-making. Since the sample in every season leads to a frame model in our technology, Figure 6.4 is the result of the temporal probabilistic model evolved over time. The visible changes in the frame models are evident and show that, in reality, an action is executed on frame *t* to evolve the next frames. This causes changes in situations over time. It is worth noting that the four temporal frame models are inseparable because they are from the same data stream (multivariate time series). They are pre-processed together as current frame is dependent on previous frame and unobservable temporal probability is revealed from the frames interlinks.

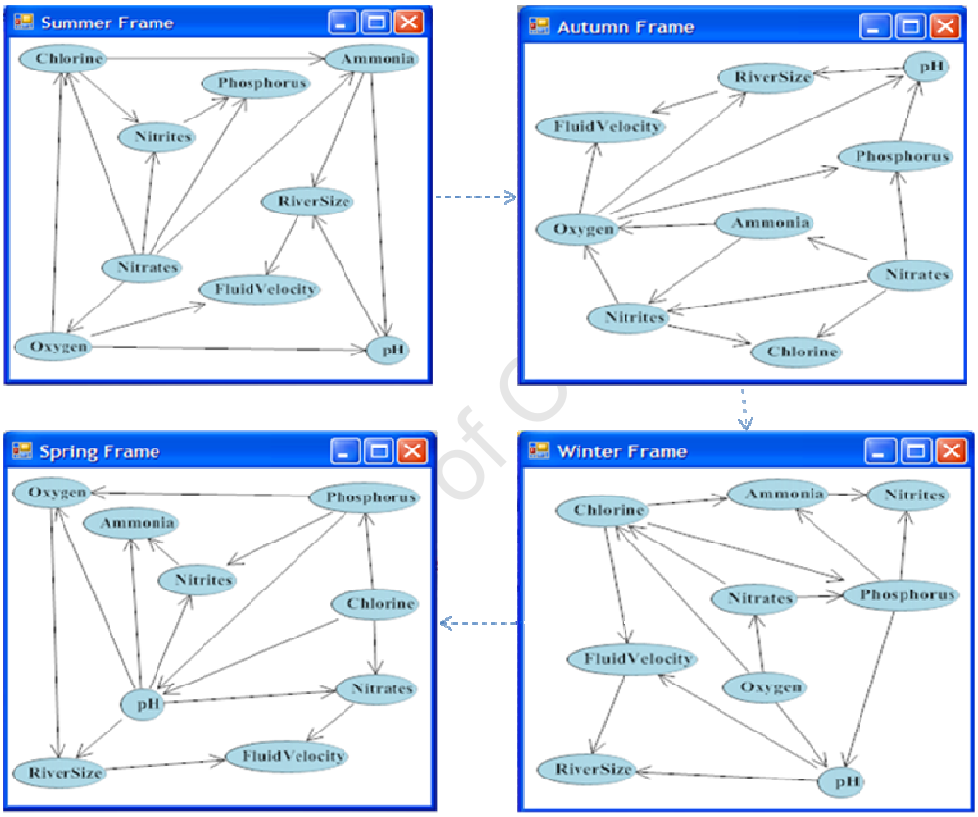


Figure 6.4: An Evolved DBN for Water Quality from the ESA Technology

Figure 6.5 shows the variability of the probability distribution tables visualised with JavaBayes for sampled *Nitrate* nodes from summer to spring frames of Figure 6.4. As if the frame changes in Figure 6.4 were not enough, visible changes in its probability distributions are obvious pieces of evidence that qualitative or quantitative information is lost if an appropriate temporal frame model is not evolved at every time step. This confirms equation (6.5) and the minimization of approximations, which has been identified as leading to increased accuracy in scientific research. Therefore, this enables decision-makers to sufficiently have better understanding about hidden situations over time. The performance evaluation

of our ESA technology is evident with the variability of the results shown, which are not guaranteed as may be seen when they are compared with the existing DBN frameworks presented at the introduction to this chapter.

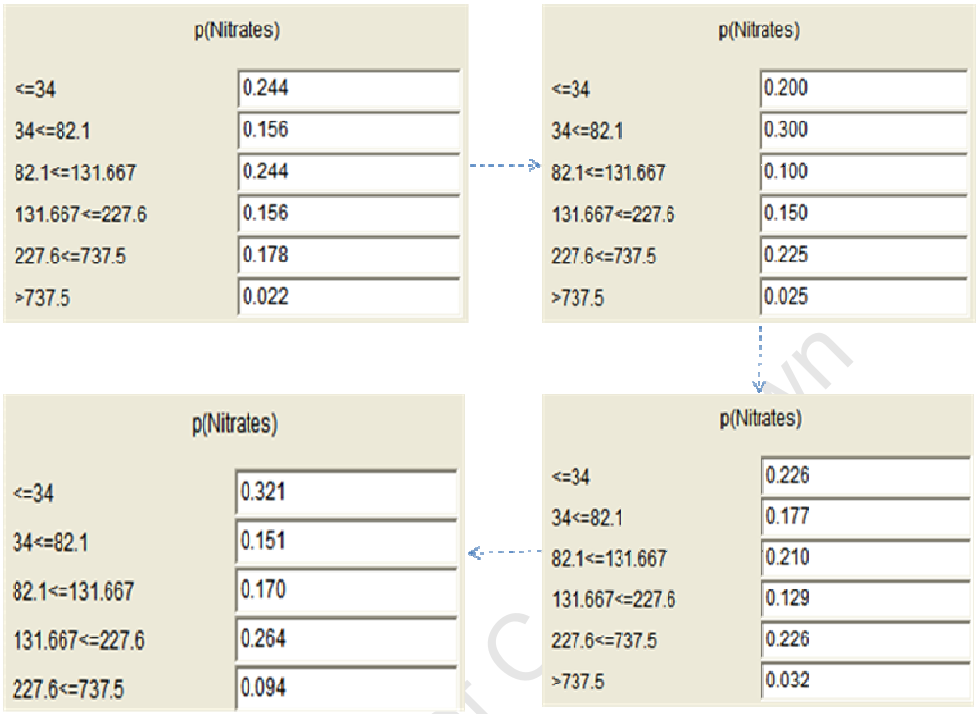


Figure 6.5: Marginal Probability Distribution Tables of Nitrates From Summer to Spring Frames

The next paragraphs specifically experiment reasoning, which facilitates decision making with examples of situation awareness. Table I and Figure 6.10 also evaluate the consistency of the ESA and once again it demonstrates the validity of the ESA technology. It therefore becomes imperative for more research developments in the 3rd class of DBN to capture variability in any domains of interests.

A. Sampled Results of Situation Awareness

The awareness of hidden situations over time on the DBN in Figure 6.4 comes as a result of acting using sampled probabilistic queries, shown as a set of equations (6.10). This reveals the local dynamics from the global behaviour. If the ESA can reveal changes in *pH*, then it is worth knowing the variability for the concentrations of chemical substances such as *Chlorine* (Cl) and *Ammonia* (NH_3). These interactively reveal the transitional knowledge about *pH, Chlorine* and *Ammonia* from the domain of water quality management. It shows the likelihood of the next situations, which give insight into the actions to be taken by the decision-makers (or water quality managers).

$$pr\ (pH^{t}\ ?\ |\ RiverSize^{t} = small, FluidVelocity^{t} = high),$$

$$Pr(NH_3{}^{t}\ ?|\ RiverSize^{t} = small, FluidVelocity^{t} = high),$$

$$pr\ (pH^{t}\ ?|\ RiverSize^{t} = large, FluidVelocity^{t} = high)$$

$$pr\ (Cl^{t}\ ?|\ RiverSize^{t} = large, FluidVelocity^{t} = high) \qquad \forall\ t\ \epsilon\ T \tag{6.10}$$

In the first situation of equations (6.10), we want to know the *most* probable values of *pH* over time when the river size is small and when the fluid velocity is high. This is similarly repeated for the rest of the situations. The revealed states of knowledge for all hidden situations in equations (6.10) are therefore shown in Figures (6.6) – (6.9) accordingly. Having seen the situations over seasons, decisions can be made easily after knowing what was not known by answering the four questions of the SA theory about the chemical water substances.

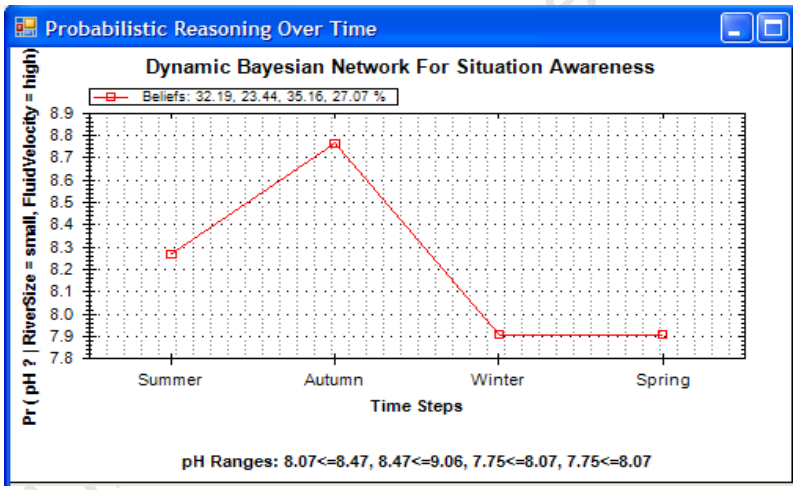


Figure 6.6: Emergent Situation of **pH** from *small* rivers flowing at *high velocity*.

Detailed understanding and decision-making process about the pattern in Figure 6.6:

Q1: What is happening?

A1: The *pH* is highly alkaline before *Winter* but uniformly tends to neutral from *Winter* to *Spring*.

Q2: Why is it happening?

A2: The rise in alkalinity is a true reflection of the presence of toxic chemicals, which reduces concentrations at *Winter* perhaps due to effects of more rain.

Q3: What will happen next?

A3: Supposing we are currently in *Spring* and since this is a seasonal variation, the *pH* will rise to about *8.29* in the next *Summer*. Similar reasoning is consistent as shown for other seasons.

Q4: What can I do about it?

A4: Before the next *Summer*, one can identify the actual chemical substances and their sources, which affect the rise of pH and moderate or prevent them from getting into the rivers.

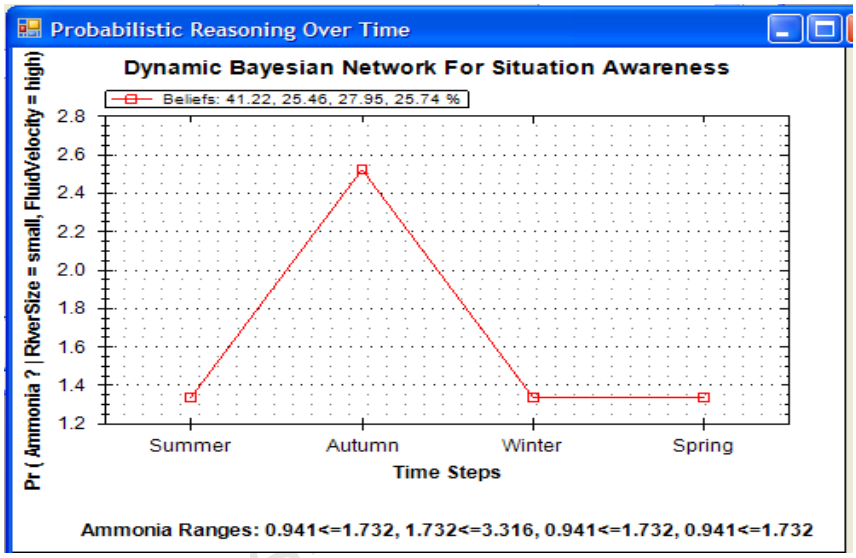


Figure 6.7: Emergent Situation of **Ammonia** from *small* rivers flowing at *high velocity*.

Detailed understanding and decision-making process derived from the pattern in Figure 6.7:

Q1: What is happening?

A1: Ammonia concentration over the seasons is about 1.3 and it increases drastically in *Autumn* where toxicity suspiciously increases the *pH* in Figure 6.6, and is harmful to life in water.

Q2: Why is it happening?

A2: This might result from farmland activities (e.g. fertilizer application) around this type of river since Autumn is regarded as the planting season. Also, more fish metabolism and deaths suspiciously produce Ammonia over the seasons.

Q3: What will happen next?

A3: Supposing we are currently in *Summer*, the Ammonia is projected to about 2.57 in *Autumn* but this is more than the water standards of about 0.5. Similar reasoning is consistent as shown for other seasons.

Q4: What can I do about it?

A4: Before Summer, farmers must be well educated and enforce pollution control measures to save lives in rivers.

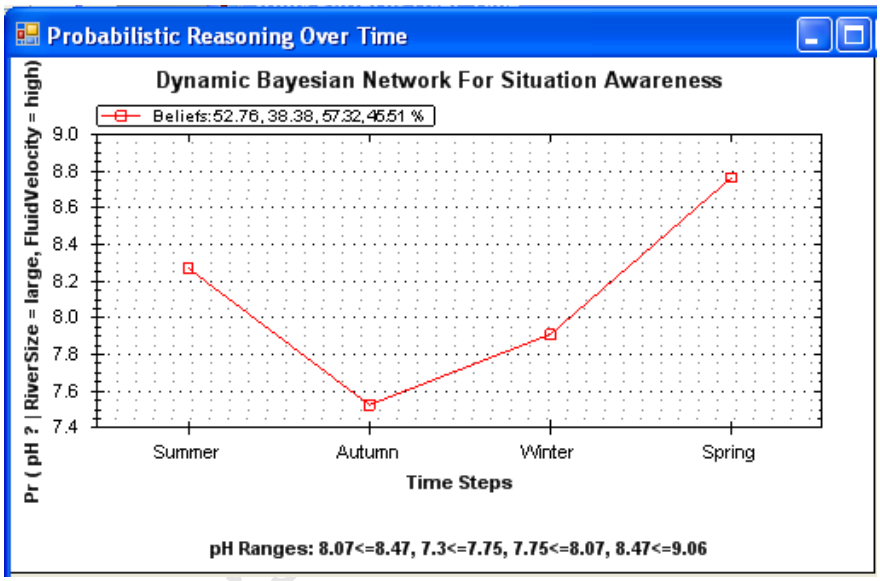


Figure 6.8: Emergent Situation of **pH** from *large* rivers flowing at *high velocity*.

Detailed understanding and decision-making process derived from the pattern in Figure 6.8:

Q1: What is happening?

A1: The *pH* is very unstable over the seasons. It is alkaline in *Summer* but tends to be neutral in *Autumn* and gradually becomes highly alkaline in *Spring*.

Q2: Why is it happening?

A2: There is presence of more toxic chemicals suspected over the seasons except in Autumn, which shows a reflection of acidic chemicals.

Q3: What will happen next?

A3: Supposing we are currently in *Winter*, about 8.8 is anticipated as the *pH* in *Spring*. Similar reasoning is consistent as shown for other seasons.

Q4: What can I do about it?

A4: Similarly to Figure 6.7, the ESA provides the need for the early identification of the toxic chemical substances, which causes high *pH* in *Spring*.

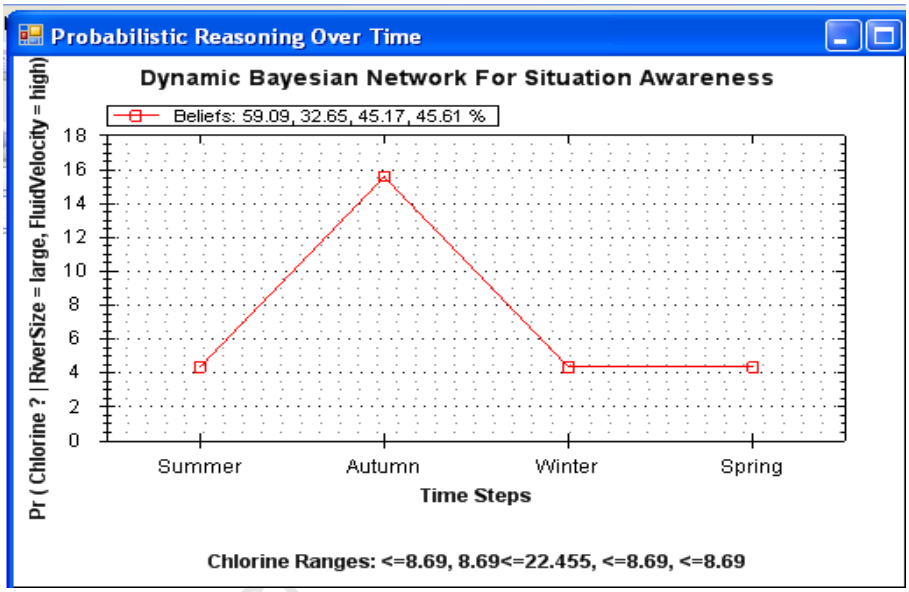


Figure 6.9: Emergent Situation of **Chlorine** from *large* rivers flowing at *high velocity*.

Detailed understanding and decision-making process derived from the pattern in Figure 6.9:

Q1: What is happening?

A1: Chlorine is an acceptable 4.3 over the seasons but it dramatically increases to 16 in *Autumn* when its concentration suspiciously tends to make *pH* acidic. This excess Chlorine can lead to breathing difficulties for the aquatic animals, such as fish.

Q2: Why is it happening?

A2: Autumn has high chlorine since the season is dominated by agricultural practices, much of which is suspected to consist of high sewage treatment that dumps toxic chlorine into the river. Chlorine level is stable in other seasons probably because the intensity of sunlight is routinely measured.

Q3: What will happen next?

A3: Supposing there is still continued abusive sewage treatment which is harmful to the environment, anticipated value of Chlorine in Autumn will be around 16.

Q4: What can I do about it?

A4: The controlling agencies must enforce laws on land practices around this type of river to maintain Chlorine concentration with the maximum of 5.0.

The awareness can similarly be applied to other situations as required. Thus, the ESA suitably showed the results for variations in chemical concentrations of rivers in European countries. It is tremendously useful for understanding hidden patterns in various domains of interest.

B. Performance Evaluations of the ESA

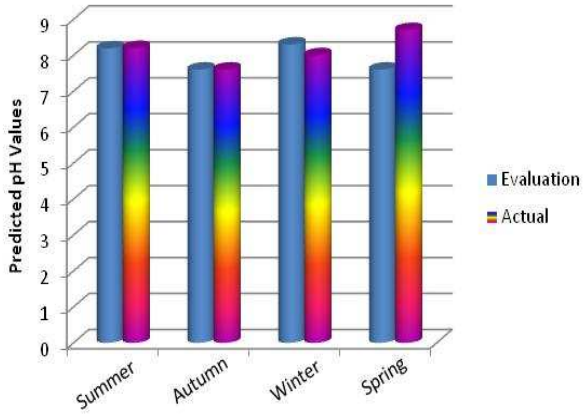
Table 6.1 and Figure 6.10 present the evaluation results of our ESA with its capability of understanding the next step ahead, using the situational patterns generated. The objective here is a measure of consistency of technology performance as Endsley [16] asserted, that the volume of data is not equal to the amount of information revealed. This implies that varying data sizes can reveal the same amount of information, if any technology developed as an active component of SA is well designed. A cross validation technique [8] was therefore used in this evaluation, where 60% of the dataset was used for training and 40% was used for evaluation. The evaluation results are therefore emerged from the evaluation data and the actual results are revealed from the training dataset which generate situational patterns in Figures 6.6 to 6.9. A DBN model is first evolved from the evaluation data and two situations are chosen at random from the same equations (6.10), which act upon the evolved DBN. The prediction results are shown in the table as 3rd and 4th situations respectively. This can be further viewed as shown in Figure 6.10.

Observe that the results of the *evaluation* and the *actual* are approximately equal which thereby show consistency. A percentage error deviation here is the absolute difference between actual value predicted from the actual training data and the evaluation value predicted from the evaluation data, divided by the evaluation value. That is, deviation = absolute (Actual value predicted – Evaluation value predicted) / Evaluation value predicted. Observe that the deviations in each evaluation and actual results for the four seasons are insignificant, except for the Spring value in the 3rd situation. This makes the percentage error of that particular Spring value 14%. However, since the rest of the deviations are less than 5%, the average performance of the ESA is approximately 87.5%. This once again shows that the

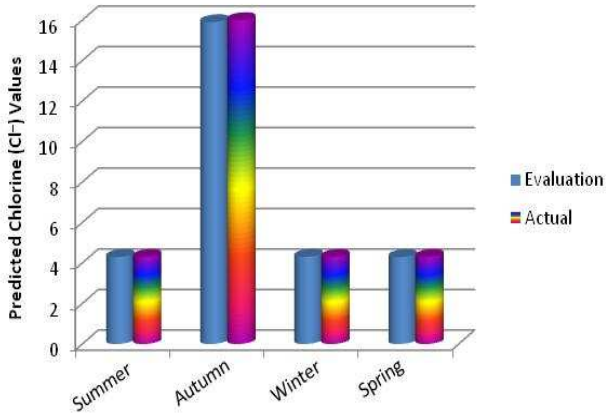
proposed ESA technology is useful for capturing or recognising dynamically changing patterns and performs reliably well.

Table 6.1: Seasonal Evaluated Situations For pH And Chlorine Respectively Using The ESA.

Time Steps	3rd Situation		4th Situation	
Seasons	**Evaluation**	**Actual**	**Evaluation**	**Actual**
Summer	**8.19**	**8.20**	**4.30**	**4.30**
Autumn	**7.60**	**7.60**	**15.90**	**16.00**
Winter	**8.30**	**8.00**	**4.31**	**4.30**
Spring	**7.60**	**8.70**	**4.29**	**4.30**
87.5%	**75%**		**100%**	



(a) pH: from large river flowing with high velocity



(b) Chlorine (Cl^-): from large river flowing with high velocity

Figure 6.10: Evaluation of Prediction Consistency of the ESA on Revealing Seasonal Situations for pH and Chlorine Respectively from Evaluation and Actual Datasets

C. The ESA, Related Techniques and Discussion

This section further evaluates the ESA with the existing DBN models discussed at the introduction of this chapter, together with some related techniques. None of these models is developed as SA technology.

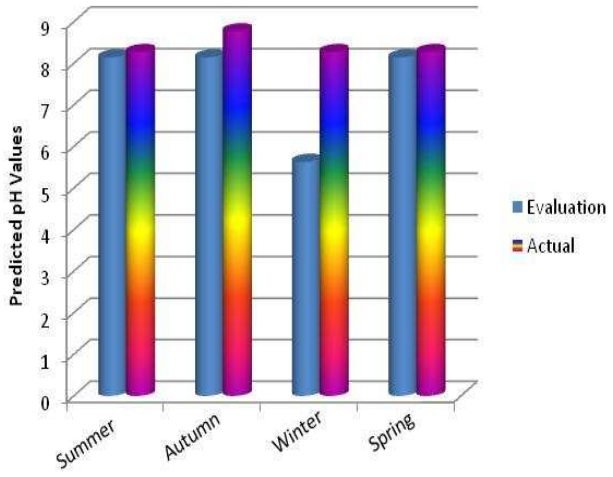
Bayesian Network technologies were compared with neural networks and ruled-based techniques in recent scientific research. Bayesian Networks showed better performance over these techniques in water analysis [78] [79]. A widely used statistical logistic regression model [80] can be mentioned, but it has lesser capability for handling complexities as compared with the ESA. The regression model is a linear function of dependent and independent variables, which has a limited time window. That is, during projection, its limitation has an exponential characteristic nature, which goes to infinity, converges, or vanishes after some time.

Decision-makers cannot project if there is a shallow understanding (information gap) of what is currently occurring in a domain of interest. One can therefore recall from the DBN literature at the introduction and in section 6.2 that most DBNs are based on the popular HMM (simplest DBN), which are often used by new technologies as a baseline for comparisons [2] [3] [81]. This chapter also follows the trends as we managed to train the simplest DBN. This means the focus is on awareness of current patterns, but not yet on projecting into the future. A similar evaluation to Table 6.1 is carried out using the water quality data. Table 6.2 is a result of the simplest DBN with the same network, but with changing conditional probability tables (recall the 2nd class of DBN in section 6.2). The evaluation results are further visualised as shown in Figure 6.11.

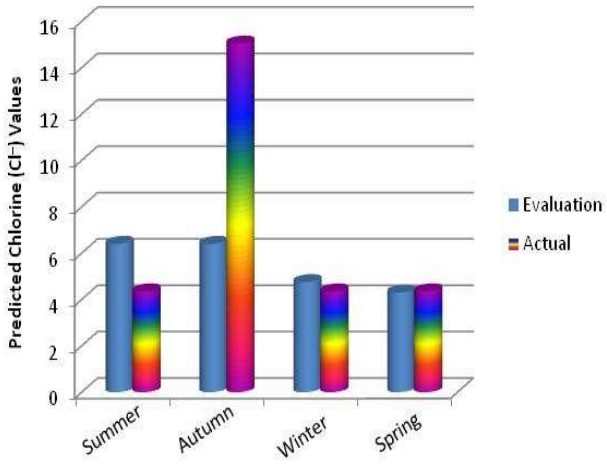
The average performance in Table 6.2 is 62.5%. One can see obvious differences between Figures 6.10 and 6.11 or Table 6.1 and Table 6.2. This is as a result of the improved differences between the DBN 3rd research niche area and others, as described in detail in section 6.2. Besides the measure of consistency between the evaluation and the actual data, one may observe many results over the season in each column of Table 6.2 being repeated. This recurrence problem does not help decision-makers to have detailed awareness of the variability of domains of interest. The 1st class of DBN (fixed network and probabilities) would give poorer results because there may be no change in the four seasons. Our ESA is based on the theory of the 3rd class, while most existing DBNs are based on the other two classes. Therefore, if the existing DBN models can integrate the ESA, there will be more SA technologies available, which would provide a better pattern understanding of real-life problems.

Table 6.2: Seasonal Evaluated Situations For pH and Chlorine Respectively Using the Simplest DBN (HMM)

Time Steps	3rd Situation		4th Situation	
Seasons	**Evaluation**	**Actual**	**Evaluation**	**Actual**
Summer	**8.14**	**8.27**	**6.40**	**4.35**
Autumn	**8.14**	**8.77**	**6.40**	**15.04**
Winter	**5.63**	**8.27**	**4.75**	**4.35**
Spring	**8.14**	**8.27**	**4.30**	**4.35**
62.5%	**75%**		**50%**	



(a) pH: from large river flowing with high velocity



(b) Chlorine (Cl^-): from large river flowing with high velocity

Figure 6.11: Evaluation of Prediction Consistency of the HMM on Revealing Seasonal Situations for pH and Chlorine Respectively from Evaluation and Actual Datasets

6.4.2 Applications in Business Intelligence

Business researchers and practitioners have stressed that Alignment of Strategic Businesses and Information Technology (SBITA) is still an ongoing key challenge for management decision-makers [17] [18]; and it often has an adverse effect on business deliveries. Researchers have made it clear that businesses with a high degree of alignment are associated with better business decision making; but there are not enough documented theories on SBITA, and competitive business intelligence technologies are even less investigated [18].

The misalignment hinders economical (or astute) and effective decision-making, which is a daily exercise that every successful management must carry out in order to avoid losses. Declaring discounts on a number of selected manufactured products – rather than all, within a space of time is economical, for instance. Often, the misalignment of business and information technology occurs as a result of information gaps between the complexity of business environments, complex models, and the business analysts. Business environments are captured as massive heterogeneous datasets and uncertainties about meaningful knowledge are embedded therein. A business environment is inevitably accountable for complexity if a percentage of its dependent variable Y has effects on a set of its independent variables $\{X_1, X_2,\ldots, X_n\}$. In practice, the degree in the relationship between the *sales* of products and historical *holidays* is an example of this complexity. Also, in reality it is not always true to account for all possible combinations of a dependent business attribute Y, given the independent business attributes X_i. This is an example of uncertainty that contributes to the SBITA problem, which affects the making of correct business decisions. The challenge is to solve this problem by bridging the information gap. The best approach to illustrate how to bridge this gap to solve the misalignment problems of SBITA and easily make qualitative decisions is using a sampled sophisticated intelligent technology.

A number of business and assessment models have been proposed to address SBITA; yet competitive business intelligence technologies like the ESA are much less investigated as applications to a concrete business example. For instance, a business model ontology was proposed as a fitness function towards business and information systems [82]. Their ontology has four pillars, which are: product innovation, customer relationship, infrastructure management, and financial aspects. Their goal was focused on the use of ontology to share a company's business model with a view to understanding the business logic. They intend to put the ontology into real practice. A framework for developing business metrics, which also describes the use of fuzzy cognitive maps to model and analyze business performances, was proposed [83]. Their framework was particularly considered for internal and external organizational assessments. We do not know any comprehensive procedure given on how to represent the business metrics for strategic levels. A theoretical framework was derived for the assessment of a

technology environment using its properties - complexity, uncertainty and disruptiveness [18]. They acknowledged that information technology that supports competitive business intelligence, which is very important for organizations, has been much less investigated. A prioritized theory diagram was proposed for assessing the level of SBITA in the frame of enterprise architecture [17]. They indicated different levels of alignment for companies, such as communication levels between IT and business, the service level agreement performance, and the level of human resources skills.

From investigations, it is observed that computational burdens or constraints and poor knowledge interpretations are challenges for business analysts when aligning with IT. For instance, managers see it as a burden to understand IT technologies implemented as generalised business applications. It probably shows that IT researchers and practitioners care more about machines in their innovations. If they put more consideration for human beings or users (decision-makers) into practice, businesses will have one-to-one alignment with IT. Customised IT solutions with concrete examples in business to include understandable presentation and elegant sample interpretations will help decision makers do better.

Furthermore, many seasoned sales managers rely heavily on classical statistical methods implemented in spreadsheets for business activities. These are good in their own frequentist way, but they cannot handle the dynamics of uncertainty about knowledge typically embedded in business environments. Also, the classical methods have limited information for capturing the complexities of business attributes. It is evident that sophisticated competitive intelligent technology is required to reason with the uncertainties and reveal meaningful information for correct decision-making.

This chapter also applies innovative ESA technology in order to address these problems. The proposed ESA technology is a temporal probabilistic reasoning over time which saves users from computational burdens. An understanding of local dynamics from emergent global behaviour makes the ESA evolve with time. It automatically discovers meaningful hidden patterns from the business environments and reveals uncertainties over time for better understanding of hidden knowledge. This provides a contributive solution to SBITA problems. Our ESA uses concrete real-life examples to illustrate to managers how to achieve competitive intelligence. Our approach is more focused on real-life applications to show business managers how to align with IT. ESA takes decision makers into consideration when interpreting results, so as to achieve the one-to-one alignment objective. In a number of comparative evaluations in handling uncertainties embedded in business, ESA outperforms the classical statistical methods implemented in spreadsheets. This excellent result validates the claim that temporal probabilistic reasoning can efficiently reveal the complexities in business environments, as a strategy that aligns the business with the IT in order to make timely and economical decisions. The major contributions of this section are:

- The application of the temporal probabilistic reasoning to SBITA as competitive business intelligence.

- Illustrating to business researchers and practitioners how to use concrete examples to align businesses with IT through presentations.

- Accounting for inevitable complexities and uncertainties embedded in business environments as a strategy for making timely, correct, and astute business decisions.

A. ***Data Analytics And Business Expectations***

This section presents the analysis of real-life experiments from validation studies of our ESA technology using meat-packer data obtained from a local South African butchery company. This analysis is carried out with the intention of acquiring competitive business intelligence regarding the butchery market, which requires taking appropriate responsive actions on any anticipatory planning and risk situations. We therefore monitored the production and the distribution of four different meat types to 17 sales outlets, based on customers' demand and supply in the local butchery enterprise.

More than 45,000 MTS sales observations were captured over the period of time January to December for the year 2005, where the ESA evolves. Each month consists of about 3,500 observations obtained from the 17 South African sales outlets. The multivariate time series dataset is described using the following attributes: *Sales-Outlet*, *Product-type*, *Quantity*, and *Sales*. These attributes are likely influenced by external factors such as public *holidays* and possibly affected by climatic conditions such as: *Temperature*, *Wind and Rainfall*. New future factors or conditions are considered when necessary. The main objectives of sales managers are to improve the performance of their business, as follows:

- maintaining TQM (total quality management) especially among customers [83],
- understanding the strengths and weaknesses of their sales-outlets,
- minimising risk of revenue losses in sales and productions and,
- being aware of the situations with their business competitors.

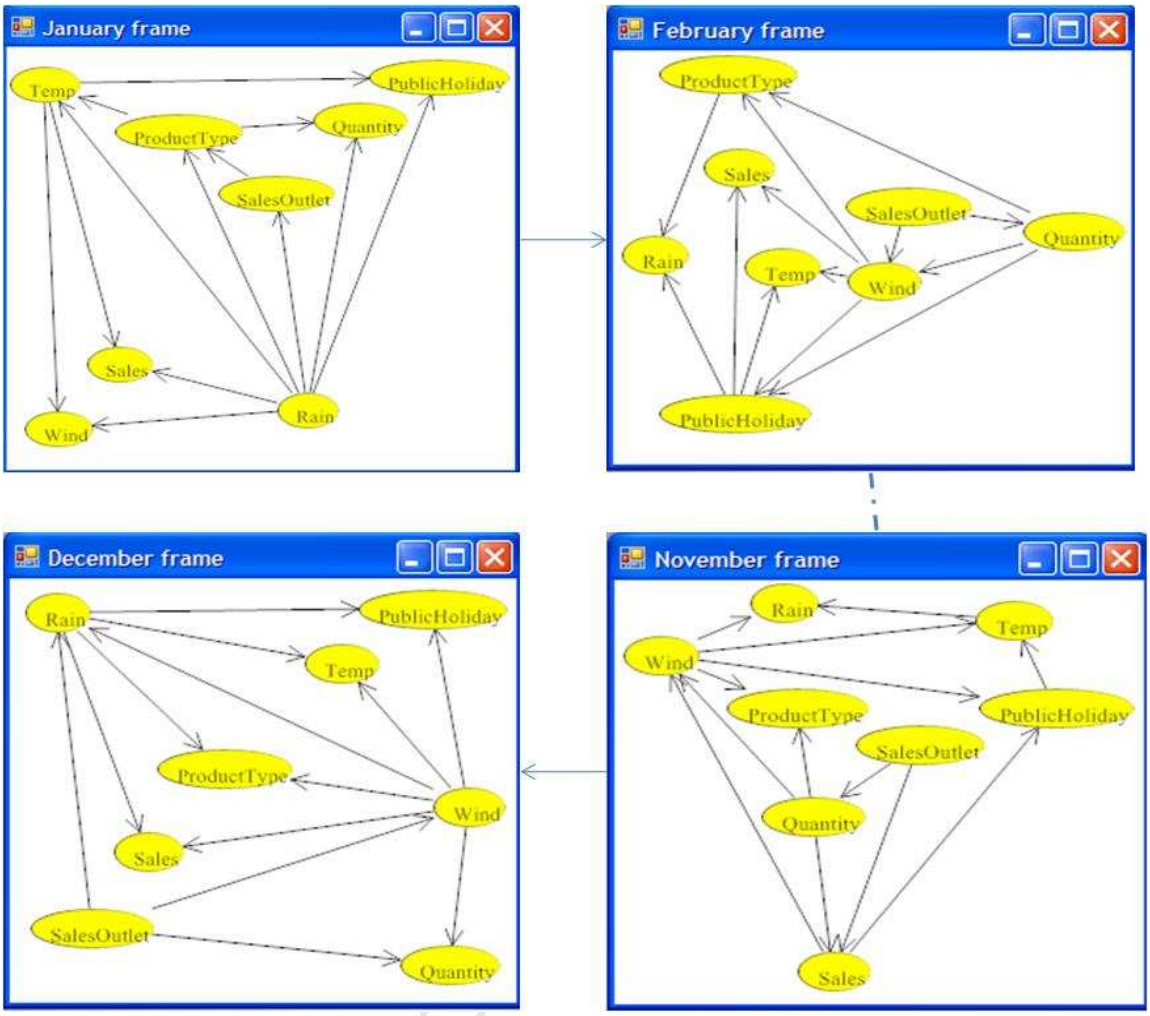


Figure 6.12: An Evolved DBN Model for Sales Management using the ESA

With ESA, the managers can, for instance, identify products with similar sales patterns, the best-selling outlet, the fast-, and slow-moving products, and will know how environments impact on their market. These real-world economic results reveal the uncertainties inherent in the butchery business, which consequently facilitates better defensive and astute decisions. ESA thus learns from the meat-packers' MTS and evolves the DBN shown in Figure 6.12. Observe that this is a global behaviour (or knowledge) of the business environment, as the ESA truly evolves temporal networks (frames) over time.

The true evolvement or variations is a result of changes in probability distributions and networks of ever-changing sales over periods of months. Based on our theory of situation calculus in the preceding chapters, especially Figure 3.4, some actions act upon a situation at a time step t to evolve next situation t+1. In practice, hidden current sales patterns in January gives rise to the frame model revealed for the month. In February, there have been some actions such as: less/more holidays, excessive rainfall, or increase/decrease in weather temperature, which impact significantly upon the sales. The environmental

changes are captured and revealed with a varied frame model for February and this occurs over the rest of the months. The next paragraphs illustrate the awareness of sales situations as intelligence to make astute and correct decisions.

B. Intelligence In Customer Buying Patterns

The ability of sales analysts to understand customer buying patterns becomes essential in order to keep pace with ever-changing customer preferences. An analyst may issue any probabilistic queries of interest similar to the samples in equations (6.11) and (6.12) which act upon Figure 6.12 and create awareness about business situations, as shown in Figure 6.13.

$$pr\ (Product\text{-}Type^{\,t} = Coldmeats \mid Sales\text{-}Outlet^{\,t} = Paarl,\ PublicHoliday^{\,t} = Holiday)? \quad (6.11)$$

$$pr\ (Product\text{-}Type^{\,t} = Lamb \mid Sales\text{-}Outlet^{\,t} = Paarl,\ PublicHoliday^{\,t} = Holiday)? \quad (6.12)$$

In equations (6.11) and (6.12), we want to understand the similarity of sales patterns between Lamb and Coldmeat in the Paarl outlet during public *holidays* with respect to customers' preferences. The revealed business situation results by the ESA are interactive, as shown in Figure 6.13. For simplicity, an interesting characteristic of the ESA is that it enables decision-makers to reason about one situation at a time.

January to December (time steps or x-axis) indicates the total monthly sales for the situations in the first and second results of Figure 6.13. The values on the left (y-axis) or on top of each situation indicate the corresponding percentage sales of each meat out of the four meat types. By answering the four guideline questions of the ESA technology that follow Figure 6.13 using the situational knowledge revealed over the months, it will facilitate better anticipatory planning and decision-making. This strategically aligns business and IT.

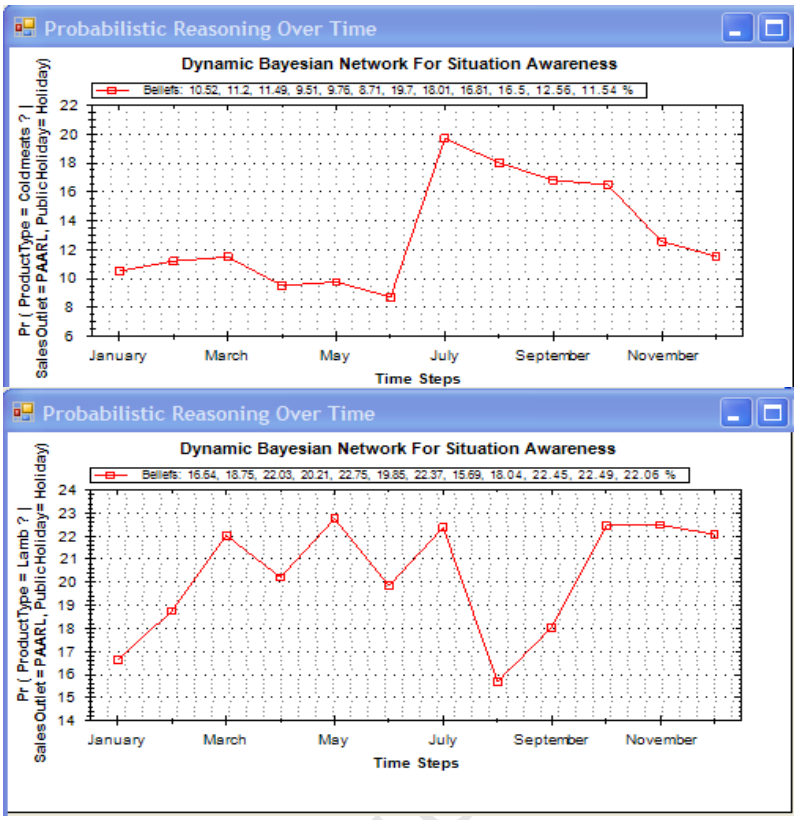


Figure 6.13: Business Situations of ***Coldmeats*** and ***Lamb*** in ***PAARL*** outlet during public ***holidays***.

Detailed understanding and decision-making process of the situations in Figure 6.13:

Q1: What is happening?

A1: By comparing the pattern of sales of coldmeats in the first situation results with the corresponding sales patterns of Lamb in the second situation results, one can observe nine similarities out of 11 patterns. This is 82% level of agreement or correlation between the two meat types. This implies that irrespective of the percentage of sales, there are increases in sales for both meats from January to March, and drops in April, for instance. These similar patterns continue until December.

Q2: Why is it happening?

A2: In the Paarl outlet, it is evident that customers who buy Lamb are likely to buy Coldmeats or vice versa. This might be due to competitors' prices, cultural beliefs, and social influences etc., in the context in which the outlet is situated.

Q3: What will happen next?

A3: Similar patterns are likely to reoccur since these business situations are revealed from sales of two years, unless there is a promotion declaration on either of the two meat types.

Q4: What can I do about it?

A4: It is a clear indication that the Paarl outlet must ensure that they stock both Lamb and Coldmeats, especially during public holidays. It is an astute decision to declare discounts on both meat types at the same time so as to influence customer acquisition and retention.

C. Intelligence in Production Planning

Using IT to plan for the production of products and services is also a challenge for business analysts. A sales analyst may also issue any probabilistic queries of interest similar to the sample in equation (6.13), which acts upon the model in Figure 6.12 and creates awareness about the business situation.

$$pr\ (Product\text{-}Type^{\,t} = Lamb \mid Sales\text{-}Outlet^{\,t} = Roeland\text{-}Street, PublicHoliday^{\,t} = holiday) \qquad (6.13)$$

From the business situation of equation (6.13), we also want to know the pattern of Lamb sales in a Roeland Street outlet and understand the effect that public holidays have on the meat sales. This gives business analysts insight into units of Lamb to produce for this outlet in order to minimise losses. Similarly, answering the four guideline questions of the ESA technology that follows Figure 6.14 using the situational knowledge revealed over the months, facilitates better anticipatory production planning and making astute decisions. This strategically once again aligns business managers with IT.

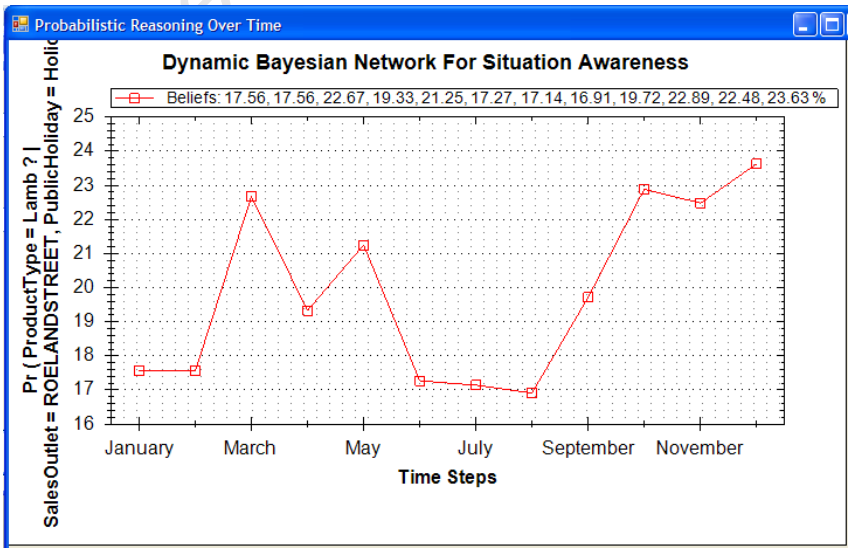


Figure 6.14: Business Situation of *Lamb* in Roeland-Street Sales-outlet during public *holidays*

Detailed understanding and decision-making process of the situation in Figure 6.14:

Q1: What is happening?

A1: With the x-axis and y-axis retaining their usual meanings, observe sales growth around *March* to *May* and *October* to *December,* while Lamb sales in other months are below 18%.

Q2: Why is it happening?

A2: The sales growth months are dominated with festive periods (e.g. Easter, Christmas, etc.). This shows that sales of Lamb in the Roeland Street outlet have sharp responses to public holidays.

Q3: What will happen next?

A3: For instance, to avoid losses on Lamb with similar business environment, about 18% of the total meat types sold should be produced or supplied for the next January to February and June to August.

Q4: What can I do about it?

A4: With respect to information obtained from competitors, it is an astute decision to routinely declare promotions or discounts only in those months with sales less than 18%. This minimizes wastages in revenue.

D. Evaluating the ESA with the Classical Statistical Methods

Bayesian network modelling technologies have been compared with other AI (artificial intelligence) techniques such as neural networks and ruled-based techniques in recent scientific research. The probabilistic networks showed better performance over these techniques in water analysis [78] [79]. This experiment therefore evaluates our ESA with the classical statistical methods as a baseline of comparisons in handling uncertainties.

Once again, the classical statistical methods use the frequency concept of probability [8] [67] to handle uncertainties, while the temporal probabilistic modelling of the ESA uses Bayes' theorem [34] over time. The uncertainties imply that if a set of unobserved sales situations arises, how will the decision-makers handle it? This type of situation enables them to be prepared for unforeseen sales contingencies. If an IT technology cannot provide a meaningful solution, then the information gap problem which leads to the misalignment is created for the business analysts. We therefore reasoned with a sampled selected set of probabilistic situations shown in Table 6.3 using both approaches, but found that the classical statistical methods fall short or are limited to capturing the uncertainties embedded in the business environment (the meat packer's sales data).

Observe zero percentages (or probabilities) for all situations chosen at random on the statistical methods column in Table 6.3. We understand that this is because the classical statistical methods assume that all possible data of sales situations must be observed [8], but this is not true in the real life of uncertainty, especially in business. For the five sales situations shown in the other column, the temporal probabilistic model of the ESA outperforms the classical methods in handling uncertainty. Out of the unobserved sales data, the temporal probabilistic model finds the most likely percentages of the meat sales, given the respective situations in Table 6.3. These results are compared as shown in Figure 6.15. This shows that the ESA could reason with the underlying uncertainties and provides meaningful information to the sales managers. This carries the decision-makers along with IT technologies, rather than being inaccessible or unfriendly, and the non-zero results contribute likely ideas to SBITA.

Table 6.3: Comparing reasoning with uncertainties embedded in business situations using both classical statistical methods and the ESA.

Probabilistic Business Situations	**Frequency Concept of Probability**	**Temporal Probabilistic Reasoning –ESA**
pr(Product-Type ^{t} = Chicken | Sales-Outlet ^{t} = Brackenfell, PublicHoliday ^{t} = Holiday)	0.0% t = *January*	18.04% t = *January*
pr(Product-Type ^{t} = Coldmeats | Sales-Outlet ^{t} = Tableview, PublicHoliday ^{t} = Holiday)	0.0% t = *February*	18.97% t = *February*
pr(Sales-Outlet ^{t} =BRUMA| Product-Type ^{t} = Lamb, PublicHoliday ^{t} = None)	0.0% t = *March*	13.01% t = *March*
pr(Sales-Outlet ^{t} =BRUMA| Product-Type ^{t} = Beef, PublicHoliday ^{t} = None)	0.0% t = *April*	14.67% t = *April*
pr(Sales-Outlet ^{t} =TOKAI| Product-Type ^{t} = Beef, PublicHoliday ^{t} = None)	0.0% t = *January*	17.04% t = *January*

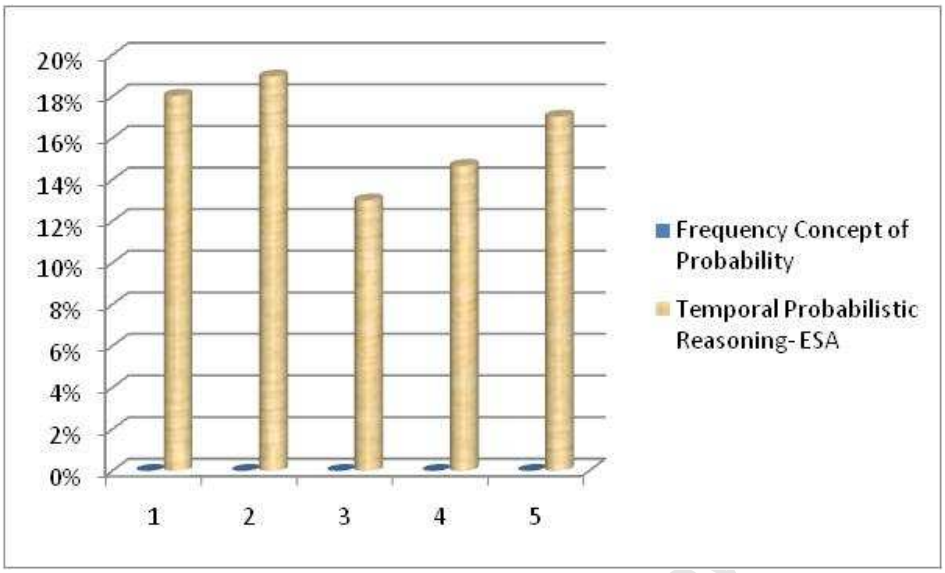


Figure 6.15: Comparing Capability of Handling Unforeseen (Uncertainty) Situations between Classical Statistical Methods and the ESA.

6.5 Conclusions

In this chapter, we have described the development of a new temporal probabilistic modelling technology, ESA, from the domain of well-defined DBN research areas. By interactively revealing local dynamics from global behaviour over time, ESA has shown itself to be suitable for better understanding of patterns in any domain of interest. The ESA is rigorously subjected to a number of experimental evaluations and applications to show that it can reveal variability in order to supply a detailed pattern understanding of real-life problems. This consequently bridges the information gap and therefore facilitates the making of accurate and economical decisions. We have shown that DBN needs to be evolved with variability in the temporal frames of the time steps. We carried out empirical evaluations on the ESA to measure its reliability as 87.5% in water quality. Recall the risk variability reflected in the situation awareness results in Figures 6.6 – 6.9, which approximately follow the theoretical understanding of water quality. The ESA contributes illustrative solutions to SBITA problems by making correct and astute decisions. It was shown that hidden knowledge of products that are likely to be sold together can be revealed to decision-makers from ever-changing customer buying patterns, through using the temporal probabilistic reasoning. We have also illustrated with our technology that revealing hidden knowledge about the percentage of product sales in festive periods can make production and distribution planning easier.

In the water quality experiment, the ESA is further compared with related techniques, such as the widely used smallest DBN, and found that the ESA guarantees wider applicability to science, engineering and industry. We also carried out performance evaluations of our ESA and the classical statistical

methods often implemented in spreadsheets regarding capability of handling uncertainties embedded in sales environments (multivariate time series). The results show from the five probabilistic situations selected in Table III and Figure 6.15 that the temporal probabilistic reasoning of the ESA outperforms the classical methods in handling uncertainties embedded in business environments. From the water quality results, this chapter shows that the ESA can potentially become a powerful pattern recognition technology to operate safely by avoiding the appearance of critical or risk situations for the real-life problems. This chapter also shows that the temporal probabilistic reasoning of the ESA can potentially become a more powerful technology for completely resolving the misalignment problems of business and IT. With reference to the complexity of the business environment, recall the hidden knowledge revealed over time about the 82% customer buying patterns and the decision guidelines including production planning. If the current implementations of classical statistical methods such as spreadsheets can integrate temporal probabilistic reasoning for decision-makers, there will be more contributive solutions to SBITA problems. Thus, interestingly, the ESA accommodates more users (e.g. managers, professionals, etc.), unlike most existing DBNs that require highly skilled users.

Chapter Seven

Emergent Future Situation Awareness (EFSA) Technology

– Predicting Future of Complex Systems

7.1 Introduction

Prediction into the future for making anticipatory planning is the cornerstone of every successful business. Industrial practitioners and researchers observe multivariate time series (MTS) from their daily business activities or dynamical systems (e.g. medical systems, retail, sensor networks, etc.). Complex hidden relationships (or patterns) are often embedded among the variables that describe such activities within and across the time steps. Some classical methods such as neural networks and statistical logistic regression models (LRM) have been applied to predict such hidden patterns, but they fall short of proving their prediction capabilities [81] [84]. The regression models struggle to predict accurately from MTS due to complex dependencies of one variable over the other [3]. Since statistics assume everything must be observed, which cannot easily be done in real-life, regression models cannot adequately capture all these dependencies [8]. Reasoning on neural networks is rigid in one direction (e.g. feed-forward), because neural network reasoning process must be considered in modelling. Also, the perceptrones used in the hidden layers of neural networks do not often have direct meaning to users in real life. Using more sophisticated approaches, these hidden patterns can be revealed from the historical MTS to predict risks, or guide actions to be taken at particular future times. Since anticipatory planning is the soul of every successful business, any retail enterprise may, for instance, intend to know which products require declaration of discounts among selected outlets in specific months for the years to come. This is an example of future prediction over time within a multitude of complex situations where Dynamic Bayesian Networks are well suited for reasoning.

A number of good Hidden Markov Model (HMM) representations of Dynamic Bayesian Networks with different characteristics have been presented, such as Auto-regressive HMM [3], Factorial HMM [6], Coupled HMM [5], etc. However, the varieties of the DBNs have opened up challenging problems to non-expert practitioners, such as managers and decision-makers, as to how to choose the most suitable model for various real-life applications. Some prediction models also suffer from convergence or exponential problems over wider time steps [8]. That is, the prediction steps get stuck towards zero or tend towards infinity. Most of these models observe stationary assumptions which lead to a DBN with repeated network structure at each time step. In reality, however, situations in some time

steps may change. It is evident that most of these existing DBNs approximate their models. That is, they do not truly evolve the knowledge in network structures and the knowledge in probability distributions, whereas the basis of DBN requires complete modelling of both types of knowledge over time [1] [8]. Murphy [3] confirms that the HMM, which is the basis of most of these representations of the existing DBNs, are limited in their expressive power. To complicate the situation further, the challenges have made technologies such as the DBNs too complex for non-experts, including practitioners [17] [18] (seasoned software programmers, managers, etc.). Despite the progress made, more work is still required in order to model DBNs automatically for all users (e.g. skilled and non-experts). Finding solutions to these challenges is difficult, especially for complete emergence to predict future trends, to accommodate all users and facilitate wider applicability.

This chapter presents a variant of the ESA [68] [69] called the Emergent Future Situation Awareness (EFSA) technology. This is another technology which predicts the future of situations from complex environments. The EFSA is emerged from the temporal probabilistic modelling of our EDBN architecture as it predicts trends over finite future time steps. The finite time steps can consist of a short or a long window, depending on the user requirements at the domain of the MTS. The EFSA eliminates the worries of choosing a good DBN model type with its automatic and complete emergence of temporal models over time from historical MTS. The EFSA's modelling spans two-dimensional (2D) orthogonal time space as a prediction strategy to avoid convergence problems. The major contributions of this chapter are as follows:

- The theoretical derivation of a temporal probabilistic theory for the new EFSA technology which predicts trends over future time steps in the absence of domain experts.

- The development of the EFSA algorithm which facilitates the mitigation of the worries of practitioners and researchers in choosing a type of DBN from the multitude of model varieties for specific applications.

- The avoidance of convergence problems when predicting longer time steps using the 2D strategy guarantees EFSA wider applicability to accommodate users in the areas of continuous monitoring of diabetic patients, sensor networks and rainfall onsets.

The rest of this chapter is organized as follows. In Section 7.2, we present the theoretical development of the EFSA. The details of the EFSA technology are presented in Section 7.3. In Section 7.4, we evaluate the performance of the EFSA's prediction consistency and accuracy when applied to three diverse areas –

future situation about rainfall onsets for farmers, the treatment of a medical patient, and counting people for possible future events. This chapter concludes in Section 7.5.

7.2 Theoretical Development

Having presented the theoretical backgrounds of Dynamic Bayesian Networks as the models that monitor and predict over time, of situation awareness as a mental model at three hierarchical levels (perception, comprehension and projection), and the details of temporal probabilistic modelling of our ESA as the technology which predicts over current time steps, we now present the analytical derivation of the EFSA in subsection 7.2.1.

7.2.1 Theoretical Derivation of the EFSA

Researchers [8] assert that prediction too far into the future (wider time lag) converges to a fixed point (i.e. remains constant for all time). In order to minimize the convergence problem, the EFSA predicts future trends using a-two step process that includes (1) the emergence of the temporal probabilistic model and (2) prediction with reasoning. The emergence of this temporal model uses the strategy of two-dimensional (2D) orthogonal time space. The first dimensional space of time steps $\{t_1, t_2 \ldots t_n\}$ monitors the behavioural current patterns in a complex environment as used in the ESA. For instance, environmentalists often like to know over four seasons (summer, autumn, winter, spring) of any observed year, the hidden patterns of pH in rivers when polluted through human activities or affected by changes in the weather. The second dimensional space of time steps $\{T_1,\ldots,T_m\}$ observes the historical patterns for each of the time steps $\{t_1, t_2 \ldots t_n\}$. For example, from the history of every season over previous years, the anticipatory pattern of the pH can be revealed over future seasons of coming years. This is a new concept as an extension of any period T in the ESA. Therefore in practice, at the end of the t_n of T_m, the EFSA updates further future trends to ensure consistency and accuracy.

Let the DBN variables V^t span the space of 2D time steps represented in the system of equations (7.1).

$$
\begin{aligned}
&T_1 = \{ V_i^{t_1}, V_j^{t_2} \ldots, V_\alpha^{t_n} \}, \text{ for each } i, j, \alpha = 1,2,\ldots\ell \\
&T_2 = \{ V_i^{t_1}, V_j^{t_2} \ldots, V_\alpha^{t_n} \} \\
&\cdot \qquad \cdot \qquad \cdot \\
&T_m = \{ V_i^{t_1}, V_j^{t_2} \ldots, V_\alpha^{t_n} \}
\end{aligned} \tag{7.1}
$$

ℓ is the length of the DBN variables and m is the length of the history. All the changing parameterizations (the DAGs and the probability distributions) of the DBN in the EFSA are now carried out across the historical time steps $T_1 \ldots T_m$. That is, the emergence (or learning of the temporal models) takes place across the links:

$$\{ V_i^{t_1}, V_i^{t_1} \ldots, V_i^{t_1} \}, \ldots, \{ V_\alpha^{t_n}, V_\alpha^{t_n} \ldots, V_\alpha^{t_n} \}.$$

Once the temporal probabilistic model evolves, prediction with reasoning now acts on the model. From the Markovian principle [8] which states that next states of a system depend on the finite history of the previous states, we can now have multiple n predictions from the space of 2D time space in equation (7.1) into the future, as follows in the set of equations (7.2).

$$T_{m+\lambda} = \{ V_i^{t_{1+\lambda}}, V_j^{t_{2+\lambda}} \ldots, V_\alpha^{t_{n+\lambda}} \}, \text{ for some } \lambda > 0$$

$$\begin{aligned} \Rightarrow \Pr(V_i^{t_{1+\lambda}}) &= \Pr(V_i^{t_{1+\lambda}} \mid E_i^{1:t_1}) \\ \Pr(V_j^{t_{2+\lambda}}) &= \Pr(V_j^{t_{2+\lambda}} \mid E_j^{1:t_2}) \\ \vdots & \qquad \vdots \\ \Pr(V_\alpha^{t_{n+\lambda}}) &= \Pr(V_\alpha^{t_{n+\lambda}} \mid E_\alpha^{1:t_n}) \end{aligned} \tag{7.2}$$

In equation (7.2), E_i, E_j,…E_α are the set of evidences or observations of V_i, V_j,…V_α respectively made so far within the space of time steps. Equation (7.2) is therefore the set of predictions that can be computed by the Bayesian inference algorithms, such as Junction tree, Variable elimination, etc. [8] [49]. The variable elimination implemented in [77] was integrated as the inference engine in the EFSA due to its efficiency. Therefore, the EFSA performs a multiple range of future predictions from the space of time.

7.3 The Proposed EFSA Technology

To simplify the theory in the previous section, this section describes the system model and the algorithm of the EFSA.

7.3.1 The System Model for the EFSA

The system model that describes the EFSA as a variant of ESA is shown in Figure 7.1. The emergence of DBNs (or temporal probabilistic models) of the EFSA is often a task of BN learning algorithms, provided they can also learn across the historical time steps $T_1 \ldots T_m$. The learning component uses our genetic

algorithms in [51] to evolve temporal Bayesian Network models, called frames, over the time steps from the MTS environments. For instance, the frames shown in Figure 7.1 belong to a first dimension T_i and the EFSA considers the history of every frame as second dimension in the previous time T_{i-1} to evolve a DBN where i = 1, 2, 3 .., n. The EFSA acts on the emerged DBN and reasons into the future. As confirmed by [3], a number of other BN learning algorithms, such as [23] [25] [38], can also be adopted herein as long as they can evolve dynamic model across the time steps. The probabilistic reasoner is the task of the Bayesian inference engine, which handles the necessary possible forward and backward propagations through the links of the frames and generates probable predicted results using collective intelligence.

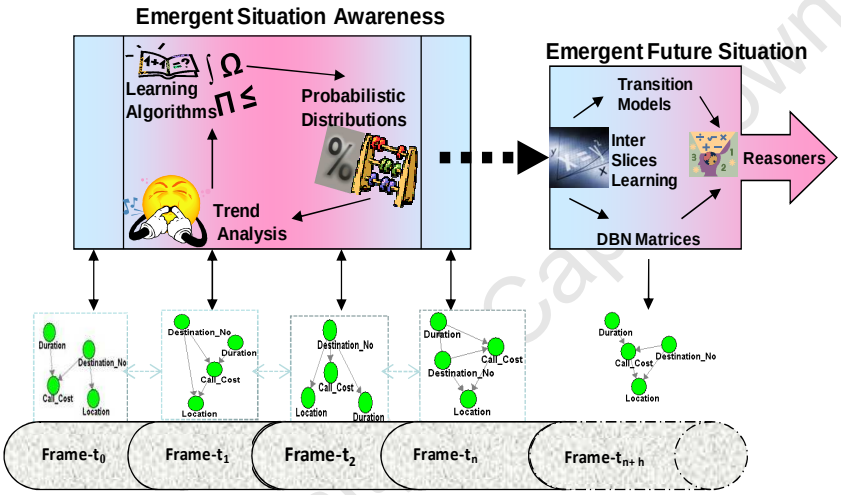


Figure 7.1: System Model For The Emergent Future Situation Awareness (EFSA)

7.3.2 The EFSA Algorithm

To further simplify this technology, the outlines of learning DBNs automatically from the EFSA algorithm are refined from our previous work in [69] [85], as shown in Figure 7.2.

All parameters retain their usual meanings and d_t is a frame of dataset at time t. It is selected into set $\{d_t\}$ over T for learning interlinked frames of the DBN. Once again, any Bayesian learning algorithms as presented earlier can be integrated as a subroutine. The algorithms carry out the intra-slice and inter-slice learning over time. Each variable in step T must have parents in step T-1. As a proof of concept, we integrated our genetic algorithms in [51] to learn the DBNs due to its efficiency.

INPUT (D_s : Multivariate Time Series - MTS)

1. While D_s = MTS,
   [i] Set *t*, the frame count, to 1.
   [ii] Set T, the historical time step, to 1, 2, …, m
   [iii] Let $d_t \in D_s$, $\forall$ $t = 1, 2, \ldots, n$.
   [iv] For each $t <= n$,
   [v] For each $T <= m$,
   - Select frame d_t into $\{ d_t \}$

   [vi] Increment T by 1.
   - Invoke Learning_Algorithms ($\{d_t\}$).
   - Store the emerged frame in *n by m* matrix *B.*

   [vii] Increment *t* by 1.
2. Return the DBN in B
3. Predict the next *n* time steps using inference engine, then exit.

Figure 7.2: Emergent Future Situation Awareness (EFSA) Algorithm

7.4 Performance Evaluations

One of the objectives of our proposed EFSA technology is to bring theory to practice (implementation) with an emphasis on applications and practical work. Subsection 7.4.1 presents the experimental prediction results of the EFSA, together with the HMM and the LRM which researchers [81] often use as a baseline of comparisons to new techniques. A number of experiments have been carried out to compare the other popular AI techniques such as neural networks with LRM for reasoning. However, the neural network approach does not prove better [81] [84]. We follow the practice by evaluating the EFSA with the HMM and the LRM as a baseline for comparison. The LRM is a function of dependent variable over the independent variables [81]. In subsection 7.4.2, we present the evaluation results of the EFSA prediction accuracy when compared with the other popular models.

We used a local southern African (Botswana) real-life dataset and two publicly available MTS datasets from the UCI [31] machine learning repository, which most intelligent-systems researchers use to validate their techniques. We carried out the evaluations of the three models on the three datasets – DIABETES and SENSOR [31] and real-life RAINFALL data obtained from a meteorological station in Botswana. Some of the sample data models are shown in appendix B. The treatment records of the behaviour of a diabetes patient were captured electronically as MTS. It has over 29,000 instances and 6 attribute columns which contain several treatment measurement codes such as 33 (regular insulin dose), 48 (unspecified blood glucose), etc. The sensor dataset captures the traffic of people flowing in and out of the main door of a CalIt2 building at UCI. It has 10,080 records and 4 attribute columns which contain the

date, time and count of people that flow in and out the building entrance. The rainfall dataset assesses onsets of rainfall for farmers to understand their varying planting dates. The sample data have over 5,000 records and 14 attribute columns which include monthly rainfall, the station where rainfall was recorded, national onsets, wind and anomalies from the sea, to name a few.

7.4.1 Comparing the EFSA Prediction Consistency With Other Popular Models

Our intention here is to determine whether the EFSA can predict multiple n-time steps consistently in terms of prediction directions. For diabetes, the models predict treatment measurements required in future for that patient, based on their historical behavioural patterns. For instance, equation (7.3) describes a situation which predicts how much of the minimum (about 7 units) measurement of regular insulin dose will be required for the next 12 months in the future year 1991.

$$Pr(Measurement^{t+\lambda} <= 7\ units \mid Code^{t} = 33) \tag{7.3}$$

for all t $\subset$ T, where in diabetes MTS, $t = \{Jan.\ldots Dec\}$ and $T = \{1988, 1989, 1990\}$.

A common empirical technique to evaluate the performance of Bayesian network technologies is to use a basic cross validation [8]. We adopted the cross validation approach by setting the 1991 time step as actual test data and learned (or trained) the DBN model across 1988 to 1990 time steps. The EFSA reasons with the temporal model and acts by predicting over time, as described in equation (7.3). This experiment was repeated using basic HMM which is dynamically constructed on the fly using GeNIe Bayesian software [27]. Similarly, equation (7.3) was also repeated using the LRM implemented in R statistical software [86]. The actual and the predicted results were recorded in each experiment and are shown in Table 7.1.

The objective in the sensor is to be able to predict counts of people for every half-hour over future weeks based on the historical behavioural patterns of traffic. This is used to determine if an event takes place in the building. For instance, in equation (7.4), we want to predict the possibility of counting the average number (between 8 and 17) of people that will flow out of the building for the next 5 weeks.

$$Pr(Count^{t+\lambda} = \text{`8<=17' people} \mid Flow^{t} = \text{`outflow'}) \tag{7.4}$$

for all t $\subset$ T, where in Sensor MTS, $t = \{Week\text{-}1\ldots Week\text{-}5\}$ and $T = \{July, Aug, Sept\}$.

We also adopted the cross validation approach by setting October time step as actual test data and learned (or trained) the DBN model across 15 weeks of July to September time steps. The EFSA reasons with the temporal model and acts by predicting over time, as described in equation (7.4). This experiment was also

repeated using other models. The actual and the predicted results were recorded in each experiment and are shown in Table 7.1.

Table 7.1: Performance Comparison of Future Prediction Directions on Three Situations using the EFSA, the HMM and the LRM

Data Sets	Time Steps	Actual (%)	EFSA (%)	HMM (%)	LRM (%)
Diabetes	January	40.15	73.26	70.43	87.30
	February	50.79	58.7	72.45	87.31
	March	74.64	90.99	79.45	87.41
	April	65.42	80.44	89.69	87.42
	May	72.51	69.08	89.07	87.46
	June	60.14	59.69	90.16	87.48
	July	69.43	76.54	93.28	87.51
	August	61.09	75.69	95.31	87.53
	September	55.17	85.48	95.36	87.56
Sensor	Week-1	19.55	12.46	24.13	25.29
	Week-2	20.88	13.39	23.07	26.18
	Week-3	24.44	18.69	33.17	37.07
	Week-4	27.11	18.81	33.41	37.95
	Week-5	3.11	2.76	21.01	38.01
Rainfall	January	70.22	53.57	58.24	45.88
	February	72.75	56.67	51.12	45.62
	March	72.39	55.48	44.09	45.33
	April	61.95	55.29	41.20	45.04
	May	62.34	56.67	40.02	44.72
	June	78.21	62.67	39.59	44.38
	July	85.73	59.81	38.90	44.03
	August	78.85	58.38	38.89	43.65
	September	83.45	58.00	36.87	43.25
	October	80.30	60.05	25.65	42.82
	November	89.21	62.31	25.14	42.38
	December	91.26	65.05	24.69	41.90

The objective of the rainfall is to be able to predict normal onset at any station over future months in every coming year. For instance, equation (7.5) predicts the normal onset of rainfall over future months for a given station number 2. This may include more complex conditions, like considering how sea anomalies and wind affect the onsets, as shown in equation (7.6), which other methods such as regression model struggle to handle [3]. For the purpose of comparison, we keep it simpler as equation (7.5).

$$Pr(Onset^{t+\lambda} = \text{'normal'} \mid Station^{t} = 2) \tag{7.5}$$

$$Pr(Onset^{t+\lambda} = \text{'normal'} \mid Station^{t} = 2, Sea\_Anom > 0.5, wind < 7.7units) \tag{7.6}$$

for all t $\subset$ T, where in Rainfall MTS, *t = {Jan.…Dec}* and *T = {1971,…2000}.*

We also adopted the cross validation approach by setting the year 2001 time step as actual test data and learned (or trained) the DBN model across the 1971 to 2000 time steps. The EFSA also reasons with the temporal model and acts by predicting over time, as described in equation (7.5). This experiment was also repeated using HMM and LRM. The actual and the predicted results were recorded in each experiment, as shown in Table 7.1.

Observe the consistencies or how each model captures the direction of predictive patterns in Table 7.1. That is, observe the increase or decrease in predictions from one time step to the next when compared with the actual results. For instance, one can see sensor results in Table 7.1 as the EFSA prediction increases from 12.46 in Week-1 to 13.39 in Week-2 and this corresponds to a rise in the actual results. The predictive patterns of the sensor networks can be visualized as shown in Figure 7.3. The rest of diabetes and rainfall results can be viewed as shown in appendix B. In view of this, one can see generally in Table 7.1 that the EFSA has the best prediction directions (consistency) of 50%, 100% and 70% for the Diabetes, Sensor and the Rainfall datasets respectively. This is as a result of the EFSA that truly evolves its network and probability distribution, and with the aid of its 2D strategy of the predictions.

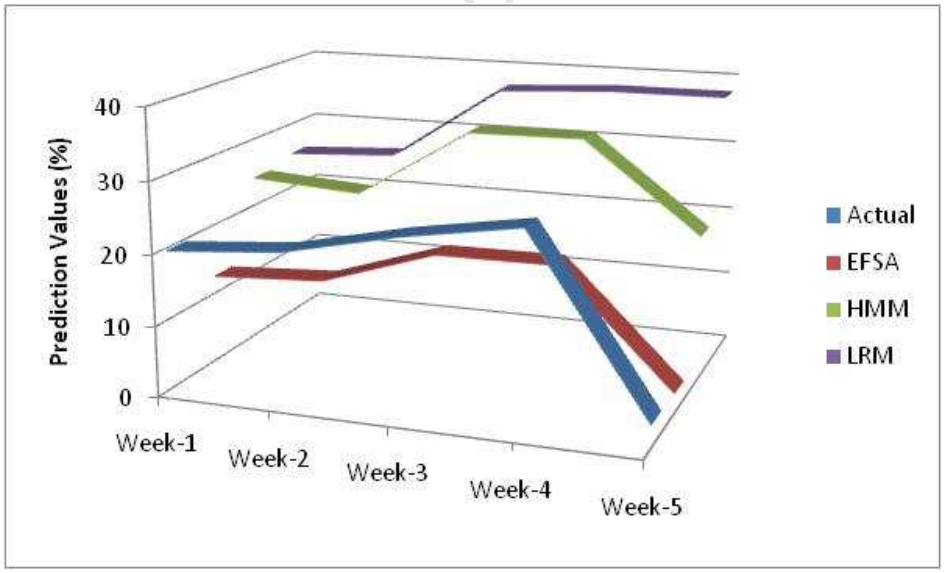


Figure 7.3: Predictive Directions of the EFSA have a very High Correlation with the Patterns of the Actual Results on Sensor Networks better than other Models.

7.4.2 Evaluation of the EFSA Accuracy Compared With Other Popular Models

Table 7.2: Evaluation of Accuracy for Future Predictions on Three Situations using the EFSA, the HMM and the LRM

Data Sets	**Time Steps**	**EFSA (%)**	**HMM (%)**	**LRM (%)**
Diabetes	January	17.53	24.58	17.90
	February	84.43	57.35	28.11
	March	78.09	93.56	82.89
	April	77.04	62.90	66.37
	May	95.27	77.16	79.38
	June	99.25	50.08	54.54
	July	89.76	65.65	73.96
	August	76.10	43.98	56.72
	September	45.45	27.15	41.29
Average Accuracy		**73.65**	**55.82**	**55.68**
Sensor	Week-1	63.73	76.57	70.64
	Week-2	64.13	89.51	74.62
	Week-3	76.47	64.28	48.32
	Week-4	69.38	76.76	60.01
	Week-5	88.75	4.76	0.11
Average Accuracy		**72.49**	**62.38**	**50.74**
Rainfall	January	76.29	82.94	65.34
	February	77.90	70.27	62.71
	March	76.14	60.91	62.62
	April	89.25	66.51	72.70
	May	90.90	64.20	71.74
	June	80.13	50.62	56.74
	July	69.77	45.38	48.64
	August	75.31	49.32	55.36
	September	69.50	44.18	51.83
	October	74.78	31.94	53.33
	November	69.85	28.18	47.51
	December	71.28	27.05	45.91
Average Accuracy		**76.76**	**51.79**	**57.87**
Overall Accuracy		**74.3**	**56.66**	**54.76**

The objective here is to measure how accurate the predictions of the models are when compared with the actual results. The accuracy is simply computed as the difference between a 100% and the percentage

error deviation, where the error is the absolute difference between actual value from the test data and the predicted value from the models, divided by the actual value [67]. This is represented in equation (7.7).

$$\text{Prediction Accuracy (\%)} = 100 - \frac{|(actual - predicted)|}{actual} * 100 \tag{7.7}$$

The accuracies are computed from Table 7.1 while the results of every time step and the average accuracy for each dataset are recorded in Table 7.2 accordingly.

Figures 7.4 and 7.5 therefore compare the performance accuracies of the EFSA with other models on Diabetes and Rainfall results in Table 7.2. The objective here is to improve the prediction accuracy. One can observe in Figures 7.4 and 7.5 that the predictions from the EFSA have higher accuracies than other models, due to smaller error deviations. The improved accuracy of the EFSA is mainly as a result of minimizing approximation by completely changing networks and probability distributions, and the capability of Bayesian technologies in handling complex situations. For instance, observe the predictions of the regression model (LRM) on diabetes and rainfall in Table 7.1 as it tends towards a convergence problem (insignificant variations). Murphy [3] also confirms that the regression models often struggle to handle multivariate complex problems. Table 7.2 shows the overall accuracy of the EFSA as 74.3%. By comparing the performance of the EFSA with other models within the scenarios of the three datasets used, the EFSA improves prediction accuracy over others with about 20%.

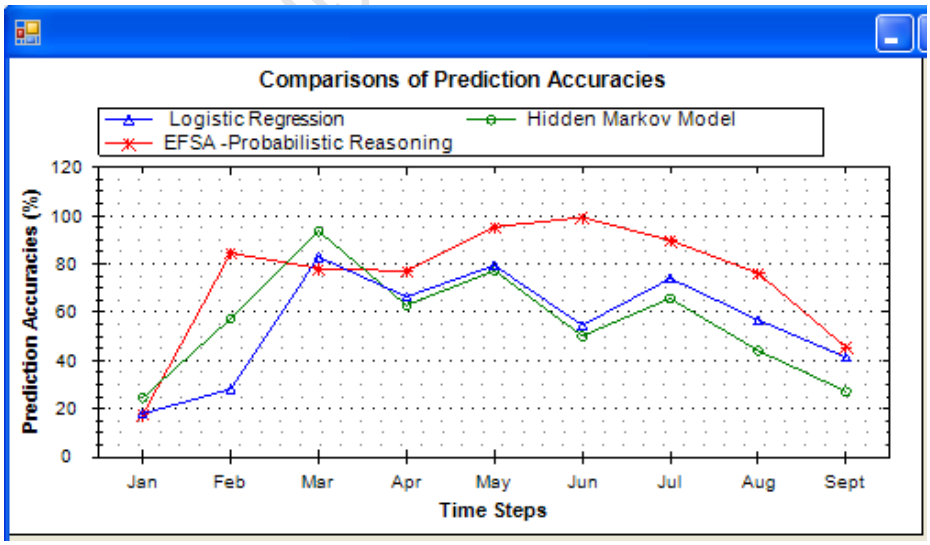


Figure 7.4: Temporal probabilistic reasoning of the EFSA improves prediction accuracies on Diabetes better than the HMM and the LRM.

The EFSA also has 89.5% performance when evaluated on some massive real-life datasets obtained from a South African local retail company whose results cannot be published due to a confidentiality agreement. Thus, the EFSA is more consistent and performs better with future predictions within the multivariate time series.

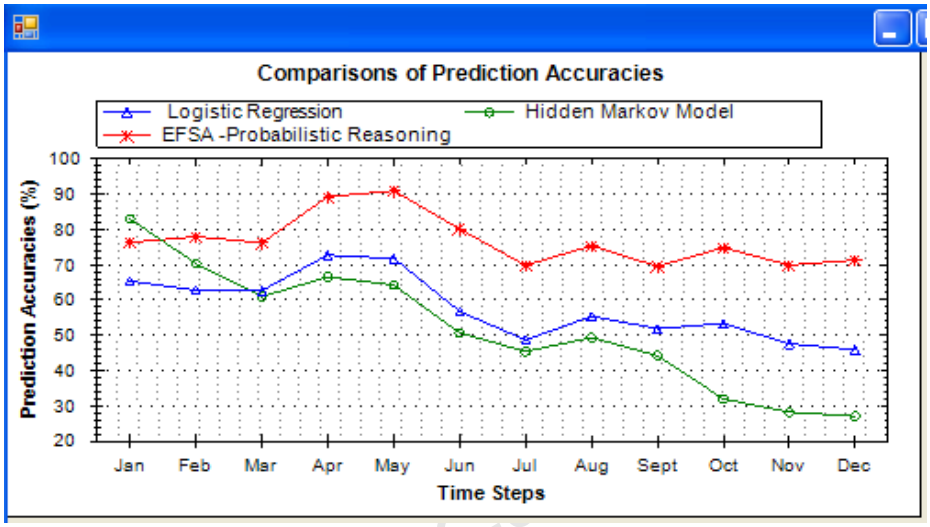


Figure 7.5: Temporal probabilistic reasoning of the EFSA improves prediction accuracies on Rainfall better than the HMM and the LRM.

7.5 Conclusions

In this chapter, we developed and presented the EFSA technology as a new temporal probabilistic reasoning for consistent multiple predictions into the future in the absence of domain experts.

The EFSA is rigorously subjected to a number of prediction evaluations using publicly available MTS in the UCI repository and the real-life datasets obtained locally. Since we have sufficiently shown how to automatically evolve DBNs with theories and the EFSA algorithm, it shows that non-experts now have fewer worries in choosing from the multitude of DBN types for real-life applications.

As shown in Table 7.1 and Figure 7.3, we methodically presented the experimental results of the three models and they predict months and weeks time steps into the future using a locally obtained Rainfall dataset, and the Diabetes and Sensor MTS obtained from the UCI repository. The EFSA proved more consistent in prediction directions than the HMM and the LRM that researchers often use as a baseline for comparisons. Using the prediction results in Table 7.1, the accuracy of the EFSA is further evaluated with other models as shown in Table 7.2. The results in Figures 7.4 and 7.5 show that our EFSA completely emerged DBNs, as it predicted 74.3% more accurately into the future than other

popular models. This improved performance of the EFSA guarantees its wider scientific and industrial applications which will attract more users (experts and non-experts).

This study shows that the EFSA can potentially become a more powerful temporal probabilistic model to become operational for all users predicting future trends to make anticipatory planning. Within the multitude of all DBN types that create problems of choice for users, recall that the EFSA simply requires implementation on any platform and starts predicting in any domain of interest. The improved overall 74.3% accuracy of the EFSA over the 56.66% of HMM and 54.76% of LRM when evaluated on the domains of the three datasets guarantees the reliability of the EFSA in many diverse areas. In respect of the good performance, we have a vested interest in applying the EFSA to other wider applications to make DBNs much simpler for use by researchers and in industry. Our subsequent work will present the details of our economic scalability framework for handling massive MTS for the EFSA.

Chapter Eight

Achieving Economic Scalability in Bayesian Learning

– Handling Massive Models

8.1 Introduction

Dynamic Bayesian Networks (DBNs) are temporal probabilistic models, which monitor uncertainties in environments and reason over time. DBNs have been applied for reasoning in intelligent systems such as audio-visual speaker detection [1], traffic monitoring [3], project profitability analysis [69], etc. A DBN is an extension of a Bayesian Network (BN) model, which comprises two components of knowledge – network structures and their associated conditional probability tables (CPTs). The basis of modelling DBNs directly from an environment captured as massive datasets is therefore a Bayesian learning problem. Researchers and practitioners have stressed that learning Bayesian networks from massive datasets is a computationally intensive problem [20] [21]. Previous research efforts such as data discretization [41] [42] and distributed learning on networks [47] [48] have contributed their own quota, up to the baseline, to this problem. Yet, very little research shows how popular single-user machines (e.g. desktops, laptops, etc.) can find solutions to learn Bayesian networks from massive datasets without a trade-off. That is, learning Bayesian networks from massive datasets by minimizing computational time, without experiencing out-of-memory problems and re-scanning old data for new observations, has been much less investigated. Finding scalable solutions to the tasks associated with the learning process is also challenging.

This chapter presents an economical solution primarily to enable single-user machines to learn good Bayesian networks from massive datasets, by using a new scalable framework called dynamic memory management in adaptive learning (**DMMAL**). DMMAL is based on concurrent distributions of agent actuators, uses dynamic memory management schemes and the representative data partitioning (RDP) algorithms to provide a complete scalable basis for an optimization strategy. DMMAL specifically prevents limited memory from crashing and optimizes learning time. Using different learning algorithms on publicly available massive datasets, we conducted a number of experiments which showed that the use of the framework results in better performance compared to using conventional learning algorithms such as in Weka and GeNIe in terms of memory and computational resources. By showing that models can be emerged from massive datasets within limited memory and time, these results suggest the integration of

our framework into the learning process of Bayesian networks to reduce the computational intensity (or NP-hard) to a minimum acceptable level. The major contributions of this chapter are as follows:

- The development of the new DMMAL framework which mitigates computational intensity of Bayesian learning especially on single-user machine to a minimum acceptable level.

- The engineering of scalable machine-learning algorithms such as the RDP and the design of the configurable actuators as a combined strategy for the success of DMMAL framework.

- The evaluation of our framework on publicly available massive datasets using different learning algorithms implemented in Weka and GeNIe systems.

The rest of this chapter is arranged as follows: in section 8.2, we present the details of DMMAL framework and its components, which include the algebraic understanding of discretizer and CPT agent actuators, and the design of configuration for the actuators. It also includes the representative data partitioning algorithms and the adaptive operator. Section 8.3 presents four experimental evaluations which include the computational time efficiency and memory scalability using publicly available massive datasets from the UCI [31] (University of California Irvine), the department of transportation in the USA [32], and the Toronto delve repository [33]. This includes a comparison with the conventional learning algorithms implemented in Weka [26] and GeNIe [27]. We conclude the chapter in section 8.4. The chapter is organized as shown in Figure 8.0.

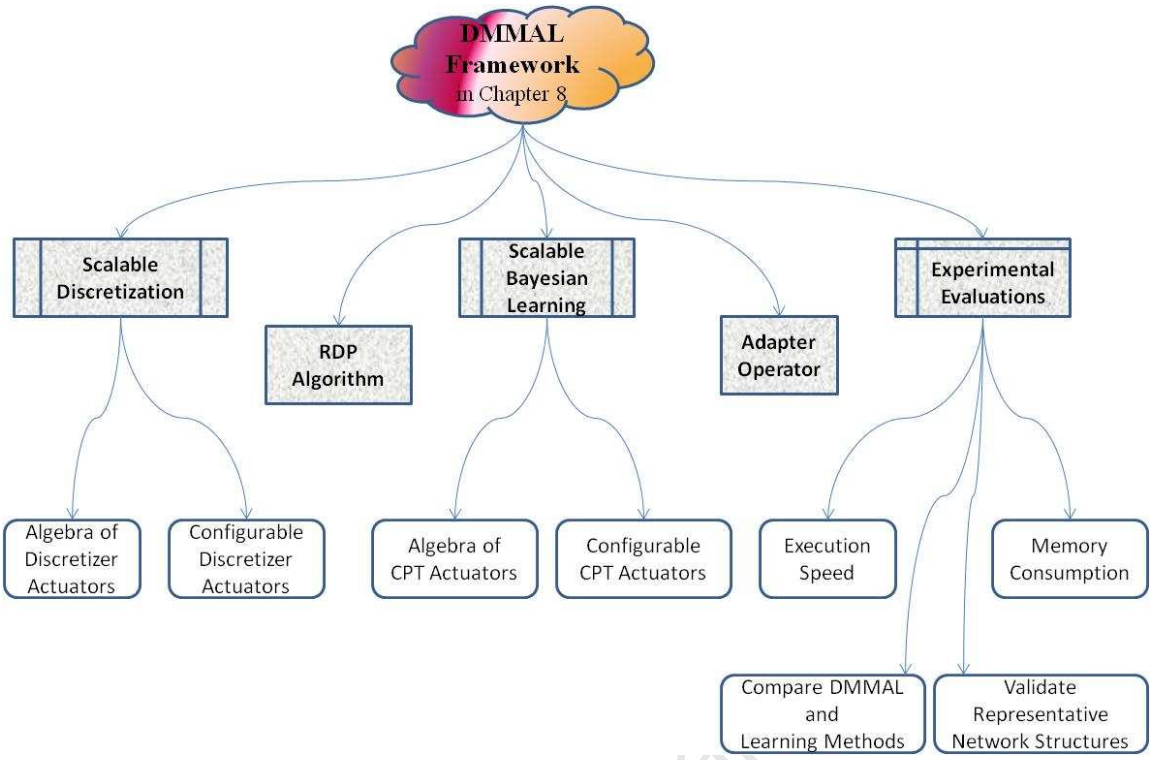


Figure 8.0: Organization of Chapter Eight

8.2 The Components of DMMAL Framework

This section presents the components of DMMAL as the basic building blocks of our framework, applied in scalable discretization [73]. The discretization process classifies numerical datasets into their corresponding interval values relative to the patterns in the data attributes. Prior to scalable Bayesian learning, scalable discretization is first presented as a supportive preprocessing strategy to optimize learning. From our practical knowledge, improving the performance of discretization serves as a sound basis for optimizing Bayesian network learning.

8.2.1 Scalable Discretizations

The DMMAL framework is a paradigm that we applied to accomplish complete scalable discretization. Complete scalability is measured by balancing between space and time as the discretizer agent actuators are concurrently distributed across various locations. If either space or time is optimized, it is an incomplete scalability as the trade-off is not beneficial to the networks used in intelligent systems. DMMAL strategy combines a memory management scheme and an agent architecture in [8]. In this strategy, an actuator is dedicated and sent to discretize values of one or more attributes. For balancing purposes, a number of actuators, rather than all, are heuristically set by users and concurrently distributed

at one time. Discretization time is faster as the actuators act on more than one attribute at a time. As the actuators complete discretization of some attributes in a pass, they are returned to the symmetric processors that reschedule them for subsequent attributes. With this, memory is continually and dynamically allocated, which then recycles each time there is scheduling of actuators for discretization.

We now define the major discretization components of DMMAL as follows: discretizer *actuators*, discretization *algorithm*, *massive environment* (or datasets), storage of previous discretized *parameters* and subsequent *observations* made after discretizing the massive datasets. The algorithm which resides on the limited memory of a machine generates tuples of intervals for the actuators to discretize values of the attributes remotely. Any of the conventional algorithms, such as [40] [41] [42], can be integrated into DMMAL, but we adopted the Hellinger-based algorithm [43] as a proof of concept because of its interval fitness measurement. The component of the massive dataset (or environment) is kept away from the limited memory and its attributes' values are acted upon concurrently in a secondary storage or across a network. This provides a competitive advantage in developing countries where the discretization process can be accidentally suspended, possibly due to electricity power failure, but modelling continues where the process stops.

Also, the previous parameters are used adaptively to discretize subsequent observations instead of re-scanning the entire old massive dataset. The last tuples of intervals of the data patterns remain the same, the data types for all attributes, etc. are examples of previous parameters. Hence, our technique strategically optimizes availability of the networks for intelligent systems through the improved discretizations. The mathematical approach used by the discretizer actuators is described in the next subsection.

8.2.1.1 The Algebra of the Discretizer Agent Actuators

The theoretical development is based on simple algebra – set theory. The results obtained when an entire massive dataset is loaded into a memory for discretization is equivalent to what is obtained when the attributes of the dataset are shared by symmetric processors using remote memory. A symmetric processor of an actuator is simply allocated and controlled in a shared memory for remote discretization of attribute(s). This is formally described as follows:

Lemma 1: *The discretized results obtained for an entire massive numerical dataset is equivalent to the results obtained when the values of its attributes* $V_1 \ldots V_n$ *are discretized concurrently and independently.*

Let a function ϕ and a binary operation * be defined on a massive schema D_s . For all values of attributes: $\overrightarrow{V_1}, \overrightarrow{V_{1+i}}, ..., \overrightarrow{V_n} \in D_s$ where i = 1, 2, 3…n, the commutative algebraic law in equation (8.1) holds, whose result is equivalent in equation (8.2).

$$\Phi(\overrightarrow{V_n})*,...,*\Phi(\overrightarrow{V_{1+i}}) * \Phi(\overrightarrow{V_1}) = \Phi(\overrightarrow{V_1}) * \Phi(\overrightarrow{V_{1+i}})*,...,*\Phi(\overrightarrow{V_n}) \quad (8.1)$$

$$\Phi(\overrightarrow{V_1}) * \Phi(\overrightarrow{V_{1+i}})*,...,*\Phi(\overrightarrow{V_n}) \equiv \Phi(D_s) \quad (8.2)$$

The function ϕ represents the actuator which takes instructions from the discretization algorithm, while * is a recombination operation. Equation (8.1) concurrently distributes the discretization processes with consideration of the memory management scheme. It asserts that irrespective of the order or remote memory used for discretization, the recombined results are equal. This implies that the result in 8.1 is equivalent to the discretized results obtained in 8.2 when the entire D_s is loaded onto a limited memory by the conventional algorithms, but our approach is faster and manages memory better. It is evident that this framework supports optimization of Bayesian Networks, which are available and used by intelligent systems, as the framework balances between discretization time and memory consumption. Having established some facts of the improved approach, the next subsection presents the designed configuration of the discretizer actuators.

8.2.1.2 The Configurable Discretizer Actuators

We designed and configured these actuators as shown in Figure 8.1 with dynamic packets of information to act upon the environments. The content of the packet consists of the *control information* and the *environments*. The control information provides a dynamic set of instructions that the actuators need to use to act upon the environments.

The constituents of the control information depicted in Figure 8.1 are as follows: *source-address* (e.g. agent-actuator-id), *destination,* which is any universal resource locator of the data (e.g. secondary storage or network machine address), *node-ids* (or attribute names) and *actions* (e.g. advance discretization scripts using the interval bins) taken by the actuators. The constituents retain their usual meanings as described. The environment acted upon is the schema table (or dataset) at various destinations. The configuration of the actuators can be expanded or modified as new functionalities are provided. Thus, our discretizer actuators are concurrently distributed because they are lightweight, mobile and independent, which is suitable on single-user machines and distributed architecture (e.g. agents, grid

systems, workstations, etc). Having supported Bayesian learning optimization using scalable discretization, the output thereof is used by the representative data partitioning algorithm described in the next subsection to improve the speed of learning network structures.

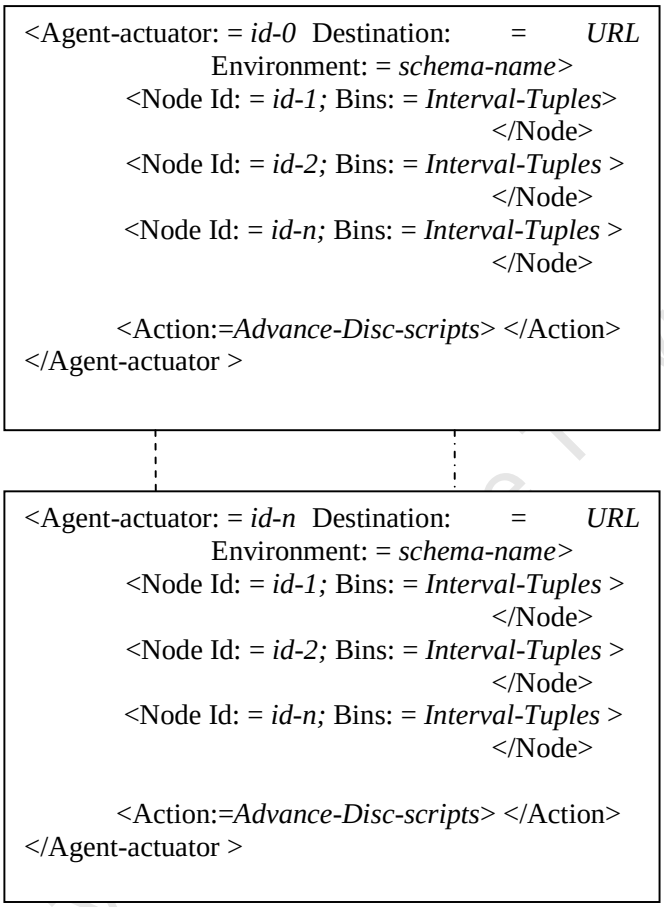


Figure 8.1: Configuration of the discretizer actuators convert numerical to discrete datasets

8.2.2 The Representative Data Partitioning (RDP) Algorithms

The RDP is instrumental to the scalable characteristics of DMMAL for handling massive datasets. We describe a partition of any dataset as a subset δ of any arbitrary set S. Supposing $\delta = \{1, 2, 3,\ldots\}$ and $S = \{\{1, 2, 3,\ldots\}, \{a, b, c,\ldots\}, \ldots\}$, then $\delta \subset S$. It is synonymous to a statistical random sample taken from a population using sampling techniques as used in [8] [87]. Many samples can be generated from a population but none has a true reflection of the entire properties or distributions of the populace. Similarly to samples, several partitions δ can be obtained from a massive dataset S, but a true reflection of the entire set of properties embedded in S can be revealed from its representative data partition, say, $\lambda = \{2, b,\ldots\}$. One can therefore make a conjecture that the similar properties embedded in both λ and the entire massive

dataset *S* are a clear indication that models learnt from both datasets are equivalent classes of Bayesian networks. Our strategy is consistent with the theorem of equivalent classes of Bayesian networks proved by Chickering [88].

Theorem [88] *Two DAGs G and G' are equivalent if for every Bayesian network B = {G, ϕ_G}, there exists a Bayesian network B' = {G', $\phi_{G'}$} such that B and B' define the same probability distribution, and vice versa.*

B and *B'* are possible equivalent emerged models from datasets *S* and λ respectively. This implies that *B'* is a representative model that captures knowledge of all network structures that can learn from all possible partitions of massive dataset *S*. With respect to the theory presented, we refined the Metropolis sampler [87] to develop the RDP algorithm shown in Figure 8.2. Our RDP algorithm eliminates random sampling in Metropolis technique to ensure that the properties of every necessary object are reflected in the representative partition. It generates a training dataset partition which accurately represents any massive datasets. The representative partition in turn produces *G'* which is equivalent to the *G* obtained from the massive dataset.

INPUT: **($\mathbf{D}_\infty$: Large Schema, α Level of Significance)**

---

1. Generate case objects, $d_x \subseteq D_\infty$
2. Compute weight, $W(\vec{x})$ of each object in d_x
3. Create initial representative partition from max weight objects, $\text{Rp}(\vec{y}_0) := W_m(\vec{x})$
4. Set *i*, the object count, to *0*
5. While $i < N$,
    - Select next object $\vec{x}_i$
    - $\rho = W(\vec{x}_i) / W_m(\vec{x})$
    - If $\rho > \alpha$ then $\text{Rp}(\vec{y}_{i+1}) = \vec{x}_i$
    - Else $\vec{y}_{i+1} \neq \vec{x}_i$
    - Increment, $i = i + 1$
6. Return $\text{Rp}(\vec{y})$

Figure 8.2: A Representative Data Partitioning (**RDP**) Algorithm

Figure 8.2 operates on massive datasets (or schema) $\mathbf{D}_\infty$. It generates case objects d_x as distinct instances, which can be a subset of $\mathbf{D}_\infty$. The distribution or weight of each object is computed as $W(\vec{x})$. An empty

representative partition Rp($\vec{y}_0$) is created, which initially keeps cases of maximum weight $W_m(\vec{x})$. For all total objects N in the large schema, the next object $\vec{x}_i$ is selected for competition as its relative contribution ρ to network relationships is computed. For winning purposes, it is compared if it has more weight than the level of significance α relatively specified by users based on machine processing capability. α is a form of pruning often used in learning research [8] and it ranges between $0.0 < \alpha < 1.0$. It dynamically determines the size of the resultant representative partition Rp($\vec{y}$).

The RDP is flexible such that it can be configured to generate many representative partitions with respect to specific key features (e.g. names) in the massive schema $\mathbf{D}_\infty$. The partition of each feature results in individual modelling. Our experiments in section 8.3 validate that this algorithm generates an accurate representative partition because it captures the significant distributions required for Bayesian reasoning which are embedded in the massive datasets. Chickering [88] also acknowledges that sophisticated algorithms (e.g. the RDP) will exercise considerable benefits on Bayesian networks.

8.2.3 Scalable Bayesian Learning

The conventional learning algorithms such as [21] [24] [38] are said to be more scalable if any of the CPT components of DMMAL presented in this section can be adapted. Our Genetic algorithm in [51] is integrated into DMMAL to learn network structures as a proof of concept. Recall that the intensity on the massive datasets implies that learning the two constituents of Bayesian networks (network structure and conditional probability tables (CPTs)) makes processor time get too consumed and therefore cause learning to come to a halt due to limited memory space. Similar theory of CPT actuators was applied in scalable discretization, described in the previous section.

8.2.3.1 The Algebra of the CPT Actuators

The theoretical development is also based on simple algebra – set theory. The CPT results obtained when an entire massive dataset is loaded into a memory for computing probabilities is equivalent to what is obtained when the attributes of the dataset are shared by symmetric processors using remote memory. A symmetric processor of an actuator is simply allocated and controlled in a shared memory for remote computations on attributes(s). This is formally described as follows:

Lemma 2: *The CPT results obtained for an entire massive dataset is equivalent to the results obtained when the values of its attributes $X_1 \ldots X_n$ are computed concurrently and independently.*

Also, let a function ϕ and a binary operation * be defined on a massive schema D_s. For all values of attributes: $X_1, X_{1+i},..., X_n \in D_s$ or equivalently nodes $X_i \in B_n$ where i = 1, 2, 3…n, the commutative algebraic law in equation (8.3) holds, whose result is equivalent in equation (8.4).

$$\Phi(X_n \mid \pi(X_n))*,...,*\Phi(X_{1+i}) * \Phi(X_1 \mid \pi(X_1)) = \Phi(X_1 \mid \pi(X_1)) * \Phi(X_{1+i})*,...,*\Phi(X_n \mid \pi(X_n)) \quad (8.3)$$

$$\Phi(X_1 \mid \pi(X_1)) * \Phi(X_{1+i})*,...,*\Phi(X_n \mid \pi(X_n)) \equiv \Phi(D_s) \quad (8.4)$$

The function ϕ represents the CPT actuator which takes instructions of parent-child relationships from the network structure, while * is a recombination operation. For instance, in equation (8.3), an actuator is instructed that a child node X_n has a number of parent nodes $\pi(X_n)$. A block-lifting Bayesian Network is presented in [37], whose model has node *M* and its associated two parents *B* and *L* nodes. Equation (8.3) concurrently distributes the CPT computation processes with consideration of the memory management scheme. It asserts that irrespective of the order or remote memory used for CPT computations, the recombined results are equal. This implies that the CPT results in 8.3 are equivalent to the CPT results obtained in 8.4 when the entire D_s is loaded onto a limited memory by the conventional learning algorithms, but DMMAL is faster and manages memory better. It is evident that this framework optimizes Bayesian Networks, which are available and used in intelligent systems as the framework balances between learning time and memory consumption. Having established the algebraic analysis, the next subsection presents the designed configuration of the actuators.

8.2.3.2 The Configurable CPT Actuators

Similarly to discretizer actuators, complete scalability is also measured by balancing between space and time, as the CPT actuators are concurrently distributed across various locations. The strategy of scalable learning in DMMAL framework also combines the memory management scheme and the agent architecture. In this strategy, an actuator is dedicated and sent to compute the CPT(s) of one or more nodes. For balancing purposes, a number of actuators, rather than all, are heuristically set by users and concurrently distributed at one time. CPT computation time is faster as the actuators act on more than one node at a time. As the actuators complete computation of some nodes in a pass, they are returned to the symmetric processors that reschedule them for subsequent nodes. With this, memory is continually and dynamically allocated which then recycles each time there is scheduling of actuators for CPT computation. We design and configure these actuators, as shown in Figure 8.3 with dynamic packets of

information to act upon the environments captured as datasets. The content of a packet consists of the *control information* and the *environment.*

The control information provides a dynamic set of instructions that the actuators need to use to act upon the environments. The constituents of the control information depicted in Figure 8.3 are as follows: *source-address* (e.g. agent-actuator-id), *destination,* which is any universal resource locator of the dataset (e.g. secondary storage or network machine address), target *node-id* (or node name), *parent-list* of target node and *actions* (e.g. advance CPT computation scripts using estimation methods) taken by the actuators. These constituents retain their usual meanings. The size of a CPT of a node grows linearly with the number of its parents and states (or parameters) therein. The actions can be triggered by estimation using subjective method or by computing from the datasets. The subjective method of probabilities is very difficult and its accuracy is not guaranteed. In practice, it is recommended to simply estimate the CPTs from datasets using algorithms such as the MLE or EM [3] [8].

The environment acted upon is simply the schema table at various probable destinations. The configuration of the actuators can be expanded as they provide new functionality for distributed agents. The CPT actuators provide a competitive advantage for developing countries where the learning process can be accidentally suspended, possibly due to electricity power failure, but modelling continues where the process stops. Thus, the configurable CPT actuators of our DMMAL framework are concurrently distributed because they are lightweight, mobile and independent. The next subsection presents the adaptive operator of DMMAL framework.

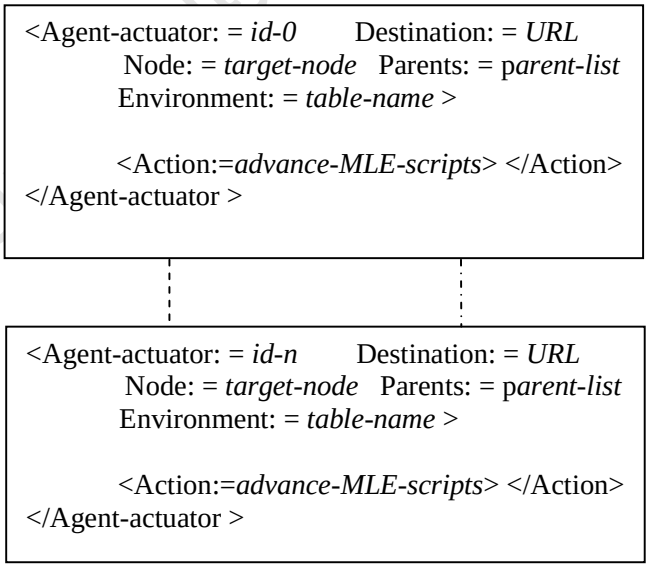


Figure 8.3: Configuration of the CPT Actuators Concurrently Computing Probabilistic Knowledge Remotely

8.2.4 The Adaptive Operator

Having presented the four principal scalability components of the DMMAL framework in the previous sections, a new adaptive operator is introduced in this section. The adaptive operator updates the knowledge of the previous model $\boldsymbol{B_o}$ when subsequent observations $\boldsymbol{d_s}$ arrive, instead of re-scanning the entire old massive dataset $\boldsymbol{D_s}$. If the update observations do not cause a significant shift in the probability distributions of D_s, $\boldsymbol{B_o}$ is returned as current model. The operator supports minimization of computational intensity. It must be noted, however, that this benefit has not been fully exploited in the literature. The outline of the adaptive operator is shown in Figure 8.4.

In order to adapt the input B_o with the arrival of input ds, the parameters $\bar{\hbar}_i$ of B_o are loaded. Suppose $\bar{\hbar}_i$ is a set of discretization parameters, then discretization technique f(.) produces d_s' using the parameters. This result is integrated into the D_s that emerged B_o by the adaptive procedure to generate D_s'. By invoking the RDP algorithm as a subroutine, the distance d between the old and new representative partitions is computed. If there is a significant difference $d > \varepsilon$, say ε = 5%, in the probability distributions, an updated B_n is emerged by triggering the genetic algorithms and the CPT actuators. Otherwise, B_o is returned. ε is heuristically set by users. Since the processes involved in the emergence of Bayesian networks are decomposable [37], only $\mathbf{d_s}$ is discretized and the updated instances in D_s' are used for the new model. Without re-scanning the entire datasets, observe that computational intensity (or NP-hard) is minimised at almost every aspect of the procedure. Although, the operator is beneficial for massive schema, retraining of models is recommended for small datasets to avoid overheads.

Hence, the adaptive operator of our DMMAL framework strategically optimizes availability of models for intelligent systems. Section 8.3 therefore brings our theory to practice.

INPUT: **($\mathbf{B_o}$:** Old Model, **$\mathbf{d_s}$:** Update Schema**)**

---

1. Load parameters, $\bar{\hbar}_i$ of B_o
2. Produce d_s' := f(d_s, $\bar{\hbar}_i$), for i = 1, 2, ..., p
3. Set Ds' = $D_s \cup d_s$'
4. d = RDP(D_s) - RDP(D_s')
5. While distribution difference, d > ε
    a. Set Rp($\vec{y}$)' := RDP(D_s')
    b. Emerge G := Genetic-Algorithms(Rp($\vec{y}$)')
    c. B_n = CPTs-Actuators**(**G**,** D_s'**)**

d. Return new Bayesian model, $B_{n.}$

6 Otherwise return old model, $B_{o.}$

Figure 8.4: An Adaptive Operator Updates Old Model With New Observations

8.3 Experimental Evaluations of DMMAL Framework

One of the objectives of our proposed DMMAL framework is to bring theory to practice with an emphasis on applications and practical work. This section describes the experiments we conducted to evaluate the performance of our framework using three learning algorithms on three massive datasets. We considered our Genetic Algorithms (GA) [51], and the GA and PC algorithms implemented in Weka and GeNIe respectively. The basic idea of how these algorithms work is to iteratively find the optimal or well-fitted models from datasets. The three dataset types include (1) El-Nino, (2) Airline traffic and (3) four banking datasets. These datasets used are publicly available massive data from the UCI machine learning [31], the department of transportation in the USA [32], and the Toronto delve repository [33], which most intelligent-systems researchers use to validate their techniques.

The El-Nino dataset from UCI contains oceanographic and surface meteorological readings taken from a series of buoys positioned throughout the equatorial Pacific. It was recorded to predict weather in different locations in the world. The real-life dataset of airline traffic from the department of transportation in the USA is recorded to control air traffic and understand the performances of airlines. We obtained a sample of over 100,000 cases for airline dataset from the department, which includes nine attributes such as origin airport, destination airport, delay time, etc. Finally, the Banking dataset from Toronto is simulated from queues in a series of banks. Customers come from several residential areas, choose their preferred bank depending on distances and their various levels of patience. Each bank has several queues, that open and close according to demand. The customers may change queue, if their patience expires. The objective here is to predict the rate at which customers reject banks due to full queues.

Once again, in practice, the major contributing factors that affect modelling performances are the number of instances, columns and number of states (distinct values) in each column of the datasets. The three datasets have varying sizes with over 178,080; 126,214; and 32,768 instances respectively. They include 12, 9 and 33 attribute columns respectively. We conducted four main experiments to compare the performance achieved by the learning algorithms on different datasets in terms of (1) algorithm efficiency in experiment 1, (2) efficiency of representative networks in experiment 2, (3) execution speed as described in experiment 3, and (4) minimizing memory consumption in experiment 4. These experiments were carried out specifically on a machine processor whose time slices are symmetric on shared memory: the controller for every CPT actuator resides in a shared memory.

8.3.1 Performance Comparison of DMMAL Framework With Other Learning Methods

We conducted experiments to find the impact of our DMMAL framework on the algorithms used with the expectation of selecting a most efficient algorithm in terms of memory usage and learning speed. The results depicted by Table 8.1 are a summary of the average performance of the three algorithms on the three datasets in terms of speed and memory used by the framework.

Table 8.1: Performance Comparison Among DMMAL Framework, Weka and GeNIe Learning Algorithms

Datasets	**Methods**	**#CPT Actuators**	**Total Speed (secs)**	**Mem-Usage (MB)**	**Status**
El-Nino (178,080)	DMMAL	1 2 3 4 5 10	712 621 576 554 452 420	36.7 36.9 37.4 37.5 37.6 37.9	Successful Modelling
	Weka GA	1	∞	59.0	Out of Memory
	GeNIe PC	1	∞	55.6	Out of Memory
Airline Traffic (126,214)	DMMAL	1 2 3 4 9	411 401 396 387 374	27.0 27.1 27.4 27.5 27.8	Successful Modelling
	Weka GA	1	-	63.6	Out of Memory
	GeNIe PC	1	-	56.8	Towards Memory Failure
Banking (32,768)	DMMAL	1 2 3 4 7 14	1839 1698 1644 1626 1582 1541	36.0 36.2 36.3 36.8 36.9 37.2	Successful Modelling
	Weka GA	1	∞	63.6	Out of Memory
	GeNIe PC	1	∞	69.0	Out of Memory

In all the experiments, the results revealed that in using GA with the DMMAL, learning models was successful while the other algorithms implemented in Weka and GeNIe suffered from memory problems by exhibiting 'out of memory' and 'towards memory failure' states. These problems are as a result of learning models from the entire massive datasets loaded on the limited memory. Observe in all experiments in Table 8.1 that no learning speed was recorded in Weka and GeNIe, because the processes were aborted. One can see that the Genetic algorithm in DMMAL performed remarkably better than the other algorithms when we consider the results provided by the highest (or best) number of actuators in each dataset. For its qualitative optimization, DMMAL speeds up learning time and minimises memory usage by emerging representative network structures and training them often with the entire massive dataset using the CPT actuators. These results suggest that using our DMMAL framework with the Genetic algorithm is an economically scalable solution which optimizes Bayesian intelligent modelling.

8.3.2 Empirical Validation of Representative Network Structures

This subsection supports our claim that representative network structures that emerge from the DMMAL framework produce similar results to those of their corresponding models that can be emerged from the entire range of massive datasets when reasoning about situations. We first modelled the airlines and El-Nino using DMMAL framework. In the second place, unlike the aborted learning processes in Weka and GeNIe, we managed to learn a model from the entire El-Nino dataset using our GA without DMMAL. It took too long and almost crashed the memory. This learning process without DMMAL was repeated on the entire airline dataset but it was not successful (crashed) because of too many states that formulate parameters of the model.

By reasoning, a set of three random situations in equations (8.5) – (8.7) act upon the airline model, which was obtained using GA with DMMAL. The reasoning results are presented in Table 8.2. The situations of the equations are interpreted as follows:

*pr (*Dest *?* | Origin =*'LAX', Cancelled = 0, Days = 'Sundays')* (8.5)

- Which destination airport is most critical to LAX airport on Sundays, if flights are not cancelled?

*pr (*Airline *?* | *Dep_Delay = 0, Arr_Delay = 0, Quarter = 1)* (8.6)

- Which airline performs excellently at the first quarter of the business by avoiding departure or arrival delays?

*pr (*Days ? | Origin = *'SAN', Diverted = 0, Quarter = 1)* (8.7)

- What is the busiest day of the week at SAN airport, if there are no diversions of flights on the first quarter of aviation business?

Another set of three random situations is presented in equations (8.8) – (8.10) which act upon the El-Nino model that was obtained using GA without DMMAL, and its corresponding model obtained using GA with DMMAL. The situations of the equations are interpreted as follows:

*pr (*Rainfall *?* | Latitude = *'2<=4'*, Longitude = *'6<=12', A*irTemp = *'-6.7<=-2.1')* (8.8)

- What is the rainfall in a location of the world whose latitude is between 2^{o}-4^{o} and longitude is between 6^{o}-12^{o} and whose air temperature falls between -*6.7 and -2.1* ^{o}C?

*pr (*Rainfall ? | Latitude = *'8<=10'*, Longitude ='*12<=18', Humidity = '-111.09<=137.63')* (8.9)

- What is the rainfall in a location of the world whose latitude is between 8^{o} -10^{o} and longitude is between 12^{o}-18^{o} and whose humidity falls between -*111.09 and -137.63*?

*pr (*AirTemp *?* | Latitude = *'<=2'*, Longitude = *'<=6'*, SeaSurfTemp = *'2.2<=6.5'*,
ZonalWind = *'<=840710')* (8.10)

- What is the air temperature in a location of the world whose latitude is below 2^{o} and longitude is below 6^{o} with sea surface temperature between 2.2^{o}C and 6.5^{o}C and whose zonal wind falls below 840,710?

The reasoning results are also presented in Table 8.2. Observe that the predicted results are equivalent in both cases, especially for El-Nino models. The absence of DMMAL in the GA learning network crashes the airline model, which makes it difficult to compare the results of that situation. Thus, DMMAL emerges qualitative representative network structures, which thereby saves lots of wasted learning time and minimises memory usages using DMMAL components.

Table 8.2: Summarized Results Validating Representative Network Structures of DMMAL Framework

Datasets	**Equation**	GA-DMMAL	GA+DMMAL
Airline Traffic (126,214)	8.5	crashes	LAS
	8.6	crashes	19393
	8.7	crashes	3 (or Tuesday)
El-Nino (178,080)	8.8	25.65<=28.01	25.65<=28.01
	8.9	25.65<=28.01	25.65<=28.01
	8.10	-6.7<=-2.1	-6.7<=-2.1
Accuracy		Equivalent Results	

8.3.3 Evaluation of Execution Speed

We conducted another set of experiments to compare the execution speed of the three algorithms on the three datasets in Table 8.1. From the results, we specifically compared the learning speeds of Weka, GeNIe, and our framework using GA on the El-Nino and Banking datasets stored remotely on a secondary storage. In the same vein as with Weka and GeNIe, which use a processor, we learned models from the massive datasets with one symmetric processor (or actuator). This set of experiments was successfully repeated by distributing and concurrently increasing the number of configurable CPT actuators while recording the learning time. The effectiveness of our framework using concurrent CPT actuators is depicted by Figures 8.5 and 8.6. The results show that using the Genetic algorithm and increase in the number of CPT actuators in our framework makes the learning process faster.

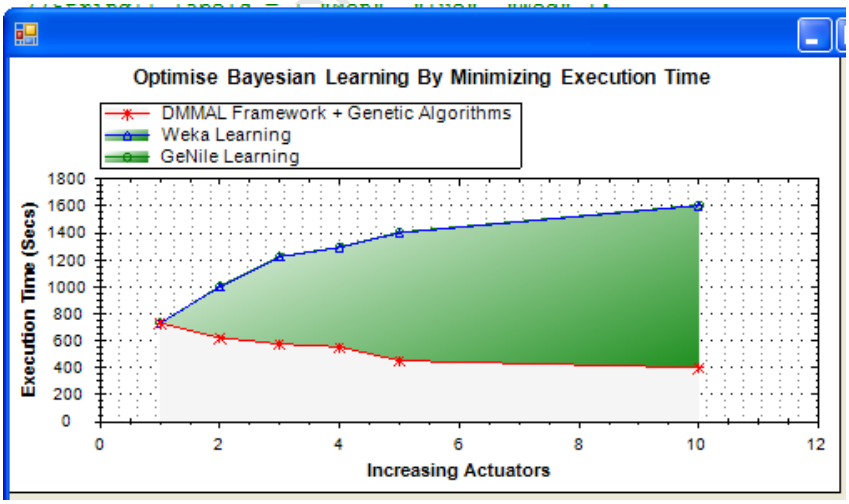


Figure 8.5: Increasing number of CPT actuators on El-Nino dataset minimizes (or speeds up) learning time better than the Weka and GeNIe, whose learning process was aborted.

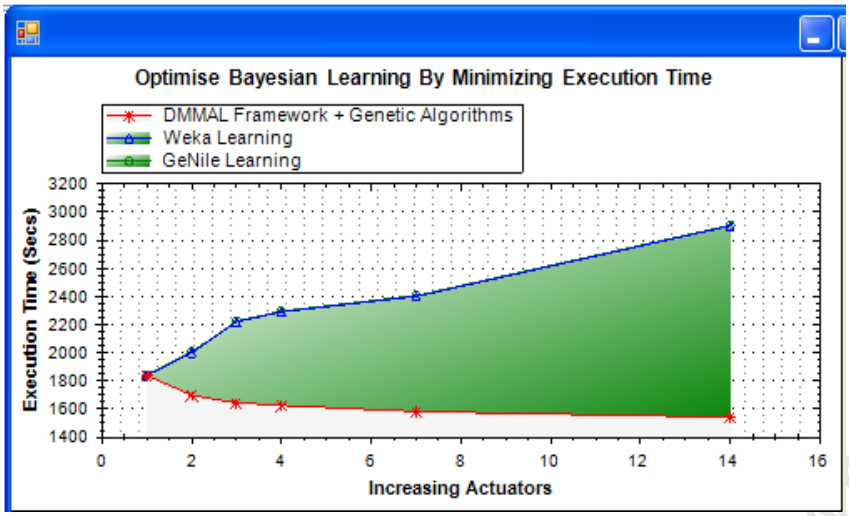


Figure 8.6: Increasing number of CPT actuators on Banking dataset minimizes (or speeds up) learning time better than the Weka and GeNIe whose learning process was aborted.

By contrast with DMMAL, when looking at Figures 8.5 and 8.6, one can observe that the time of Weka and GeNIe learning algorithms increases exponentially as they tend toward infinity. Their modelling processes were eventually aborted due to the limited memory, as shown in Table 8.1. By comparing the learning speed of the highest (best) number of CPT actuators used with the usual single processor of Weka and GeNIe, within the allocated limited memory, the GA using the framework performs remarkably better than Weka and GeNIe. This shows from Table 8.1 or Figures 8.5 and 8.6 that by using the framework, the GA modelling is successful and faster than Weka and GeNIe when learning models from the three datasets respectively.

A similar performance pattern is revealed by Figures 8.5 and 8.6 when the old model adapted new observations using the adaptive operator in Figure 8.4. By cross validation method [8], 20% of each of the datasets were selected at random as new observations, and were used to update the old knowledge (or model). Minimization of the learning time results was also recorded by increasing the CPT actuators similarly to Figures 8.5 and 8.6. However, such excellent improvement balances scalability by equally managing the limited memory, as experimented in the next subsection. Thus, a complete scalable learning optimizes Bayesian networks and makes them reliably available for intelligent systems.

8.3.4 Evaluation of Memory Consumption

The results of experiments 1 to 4 show that users who are not fortunate enough to be in a networking environment or who cannot afford a suitable networked machine can safely learn models from massive datasets on a machine with limited memory using our DMMAL framework.

We conducted experiments similar to 1 to 3 above to evaluate the memory management capability of our framework when using the El-Nino and the Banking datasets. The results are depicted in Figures 8.7 and 8.8. As discussed in our introduction, most existing conventional learning methods load the entire datasets into the memory, which can lead to memory failure when they become massive. Also, a client workstation that receives processing loads from a server is susceptible to memory failure, if most of the available memory slots are fully occupied or the processing takes too long to complete. One can observe in Figures 8.7 and 8.8 from the Weka and GeNIe learning processes that varying the number of CPT actuators does not improve on memory usage because all the records are loaded onto the memory at one time. The details of occupied megabytes of memory can be seen in Table 8.1, which eventually results in a halt state.

From the results in Figures 8.7 and 8.8, the GA using our framework successfully managed the same limited memory by concurrently exploiting secondary storage resources on remote locations (e.g. hard disk on a machine or on workstations). In comparing DMMAL with Weka and GeNIe learning, only 37.9 megabytes and 37.2 megabytes of memory were used in Figures 8.7 and 8.8 respectively by the fastest learning process of the configurable actuators. Though there are slight increases in memory usage as the number of CPT actuators increases, one can observe in Figures 8.7 and 8.8 that our framework reduces the memory usage to a minimum acceptable level. For example, this shows that DMMAL saves 35.76% and 41.51% of the limited memory from crashing, as compared with Weka learning in Figures 8.7 and 8.8 respectively.

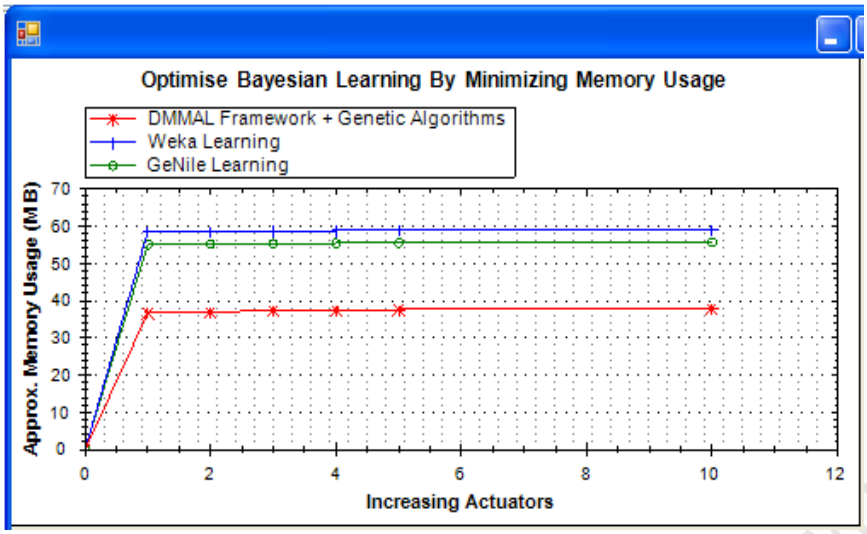


Figure 8.7: Concurrent distribution of actuators on El-Nino dataset minimizes memory usage better than Weka and GeNIe whose learning process crashes the memory.

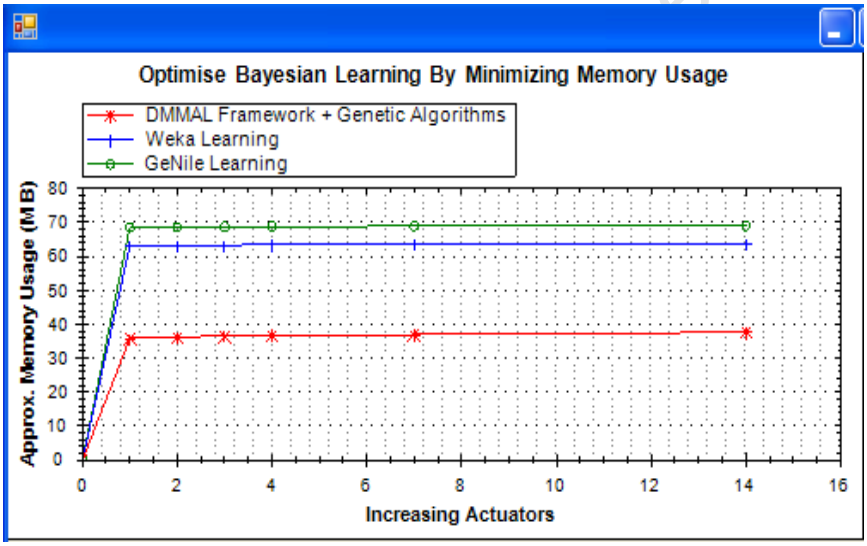


Figure 8.8: Concurrent distribution of actuators on Banking dataset minimizes memory usage better than Weka and GeNIe whose learning process crashes the memory.

As percentage differences of the best (highest) number of actuators are computed from Table 8.1, Table 8.3 shows the average percentages of memory minimized by our framework as improvement over the Weka and GeNIe learning processes in terms of memory management. The overall average memory minimized is 43.76% as shown in Table 8.3. This once again supports our claim that users cannot afford to trade-off between *time* and *space* in real-life Bayesian learning. These impressive scalability results of our economic framework on a computer machine guarantee further improved performance if deployed on

environments with additional physical processors or more suitable networks. Thus, this framework is significantly beneficial to the existing work on learning approaches in optimizing learning and making Bayesian networks reliably available for intelligent systems.

Table 8.3: Average Percentage of memory minimized by DMMAL over other memory usages of learning methods.

Other Methods	**Datasets**	**DMMAL Optimization (%)**	**Average Memory Minimization**
Weka	El-Nino	35.76	44.52%
	Airlines	56.29	
	Banking	41.51	
GeNIe	El-Nino	31.83	42.99%
	Airlines	51.06	
	Banking	46.09	
Overall Average Memory Minimization			**43.76%**

8.4 Conclusions

We have proposed in this chapter the development of a DMMAL framework for learning Bayesian network models from massive datasets, as an alternative optimization solution to the computational intensity (or NP-hard) problems arising in intelligent systems. Experimental results revealed that the use of our framework is an economically scalable solution to the problem, as it does not require purchasing expensive hardware. The results of Figures 8.5 – 8.8 support the claim that using the framework with our Genetic algorithm leads to faster emergence from massive datasets without memory failure as compared with conventional algorithms such as Weka and GeNIe modelling. This excellent result confirms the concurrent distribution of the configurable CPT actuators including other efficient components in DMMAL.

Our framework was rigorously subjected to a number of scalability evaluations to show that models can be emerged from massive datasets by balancing between space and time to mitigate memory failures, computational time problems, and therefore optimize network learning. One of its greatest competitive advantages is the capability of our framework to continue modelling where execution stops, if the learning process is accidentally suspended possibly due to electricity power failure. This is as a result of the CPT actuators acting on the massive datasets residing on the hard-drive. This is a tremendous solution for developing countries.

We have shown through experimentation that limited memory needs to be dynamically managed by concurrently distributing configurable CPT actuators remotely (secondary storage or workstations) with faster execution. We have also shown qualitative experimental results by using representative network structures which guarantee a significant optimization of Bayesian models, making these models available for reasoning by intelligent systems.

This study shows that the framework has the potential to become a more powerful scalable solution that puts an end to all computational problems raised in various research efforts. The impressive results presented in this chapter motivate the successful application of DMMAL components in [73] [89]. If Bayesian model researchers and intelligent-system engineers can integrate the framework into their developments, there will be more contributive solutions to computational intensity. We have a vested interest in applying our framework to solve computational intensity problems which are also challenges in diverse research fields and industries.

Chapter Nine

Concluding Remarks and Suggested Future Work

9.1 Research Summary and Results

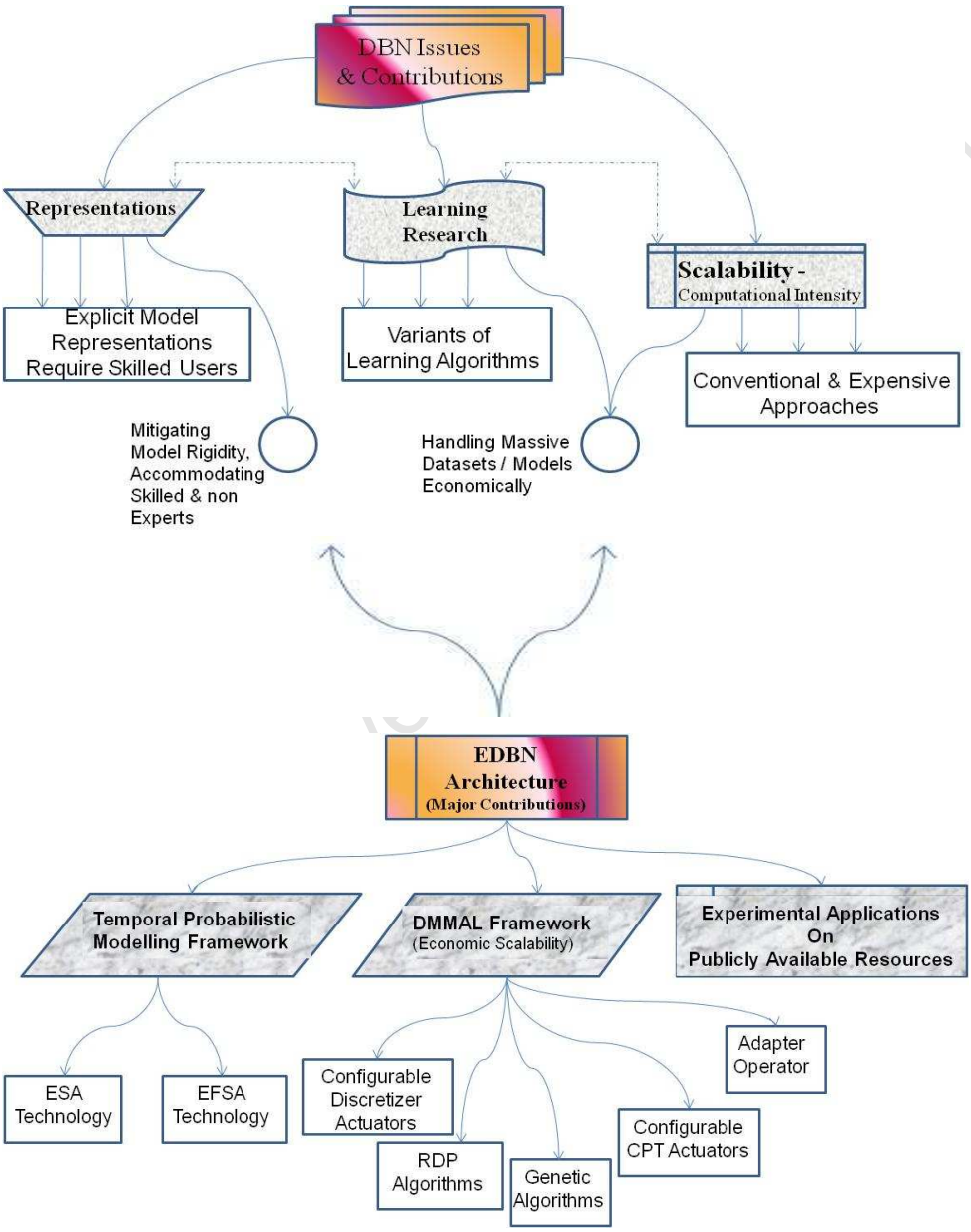


Figure 9.1: The EDBN Architecture Providing Computational Intelligent Systems Solutions to Address DBN Modelling Issues.

It is common knowledge and inevitable that at the initial stages when a new technology is introduced, it may be less welcomed by the public than a few years down the line when it is better known, and has been improved upon. One would recall that the architecture initially proposed for the aeroplane in some decades back was not expected by the public to be as good as what it is producing today. In the same way, the architecture proposed in this thesis will potentially one day provide most computational intelligence solutions to many issues in the areas of science and technology. This chapter therefore summarises this research and the contributions of the proposed newly developed EDBN (Evolving Dynamic Bayesian Network) architecture to mitigate issues with DBN (Dynamic Bayesian Network) models as shown in Figure 9.1. It is worth noting that part of the originalities of our approach includes illustrations and the semantics one can associate with them.

Chapter Two critically reviewed what researchers have done so far in DBN models, described why Bayesian learning research cannot be separated from DBNs, and presented the suggested solutions for addressing the ongoing challenges of learning. Our analysis, illustrative descriptions and comparisons revealed that most of the existing DBN models, such as the Factorial HMM, Coupled HMM, Autoregressive HMM, PDBN, etc. are based on HMM representations, which have contributed to the baseline of temporal modelling; however, they are still rigid and are limited in their expressive power. The representations of these models are often associated with skilled users or domain experts, which means that non-skilled users struggle to use them. The full capabilities of DBNs have not been exploited and learning such models from massive datasets requires an economic solution that is completely scalable. These significant reviews contributed to applications in [51] [73].

In Chapter Three, theoretical backgrounds of DBNs are methodically described, substantiating the development of the proposed EDBN architecture. It addresses and accounts for the inherent ongoing computational intensities associated with learning BNs (Bayesian Networks) from massive datasets. The focus of Chapter Three is on providing fundamental theories such as principles of probability, situation awareness (SA) and calculus, which can be used to easily evolve BNs dynamically. These theoretical backgrounds contributed to applications in [69] [89].

After our preliminary research investigations into various Bayesian learning algorithms, we first identified ‘backtracking’ as an overhead of the computational intensity issues presented in Chapter Four. In that chapter, we presented an improved GA (genetic algorithm) to emerge optimal Bayesian Networks with the emphasis on avoiding backtracking problems and integration of the ADTree technique. The GA was benchmarked with some known networks to show that it can structurally discover equivalent network models and we compared its performance with the Weka system, the Hugin system and the work of William et al. [39]. It is one of the major components of the EDBN. The chapter therefore demonstrated how immediate detection of telecommunication anomalies can be achieved, as optimized learning

algorithms like the GA can emerge individual Bayesian Networks to enhance the response rates of detection models. The enhanced GA and its applications to achieve immediate detections in telecommunications anomaly led to publications in [51] [63].

Varieties of existing DBN models are restricted (or rigid) with assumptions and are limited in their expressive power because they often require the intervention of domain experts. After extensive research investigations, we first clarified the models into three well-defined research niche areas and introduced the EDBN architecture to address the third research niche area without technicalities so as to accommodate all readers. Chapter Five integrates two new orthogonal frameworks as the development of the EDBN architecture for temporal probabilistic modelling from MTS and as an alternative economic solution to the computational intensity (or NP-hard) problems arising in intelligent systems. The temporal probabilistic modelling framework derived the ESA and EFSA technologies, where the ESA monitors complex environments over current time steps and the EFSA projects the behaviour of the environments into the future. The DMMAL (dynamic memory management) framework provides a scalable economic solution for handling massive datasets and models. From the evaluations, the chapter reveals that the EDBN has the potential to become a more powerful deliberative and reactive architecture that finally puts an end to the worries of non-expert practitioners with regard to choosing the appropriate model among DBN types; it also ends the computational problems raised in various research efforts. This contributed to the publications in [68] [71].

Decision-makers such as non-expert practitioners struggle to interpret most existing DBN models when attempting to take correct actions. This has consequently led to ongoing information gap problems (e.g. between complex models and correct interpretation by decision-makers) [16]. The information gap problem is a result of decision-makers not being well acquainted with being patterns currently occurring in their various domains. Bridging this gap is a challenge. Our E**SA** technology simply enables one to have better understanding over time about current hidden situational patterns embedded in any complex systems of interest. It is rigorously subjected to a number of evaluations to show that it can reveal variability for having a detailed pattern understanding of real-life problems. Its power motivates the successful applications of the ESA to many areas, most notably in the assessments of rainfall onsets for southern Africa using real-life data from a meteorological station [70], in business intelligence research of the southern Africa Institute for Management Scientists [68], in water quality management [85] and in project profitability analysis [69].

Prediction into the future for making anticipatory planning is often said to be the cornerstone of every successful business. The varieties of the existing DBNs obviously opened up challenging problems to non-expert practitioners such as managers and decision makers on how to choose the most suitable model for various real-life applications. Some prediction models such as statistical models and HMMs

also suffer from convergence or exponential problems over wider time steps [8]. That is, the prediction steps get stuck towards zero or tend towards infinity. The EFSA technology takes the ESA further by reasoning and projecting anticipatory situations into the future. The EFSA eliminates the worries of choosing a good DBN model type, with its automatic and complete emergence of temporal models over time from historical MTS. The EFSA's modelling spans orthogonal two-dimensional (2D) time space as a prediction strategy to avoid convergence problems. It has been rigorously subjected to a number of prediction evaluations using publicly available MTS in the UCI repository and real-life datasets obtained locally, which led to publications in [71]. The experimental studies show that the EFSA can potentially one day become a powerful temporal probabilistic modelling approach for solving most anticipatory problems in science and industry.

One can recall that researchers and practitioners have often stressed that learning Bayesian networks from massive datasets is a computationally intensive problem. Yet, very little research shows how popular single-user machines (e.g. desktops, laptops, etc.) can find solutions to learn Bayesian networks from massive datasets, despite the fact that there are methods of distributed learning on networks, which are, however, expensive to set up. With further investigations into the computational intensity problems that may arise from the DBNs, the economic scalable framework of the EDBN that we propose is reactive in nature and is called the DMMAL (Dynamic Memory Management in Adaptive Learning). Using various components such as the discretizer actuators, RDP (representative data partitioning) algorithm and the CPT actuators, DMMAL optimizes computational time efficiency and saves massive Bayesian Network models from crashing, even when used on reasonably inexpensive single-user hardware, such as a PC or laptop. One of its greatest competitive advantages, especially in developing countries, is the capability of DMMAL to continue modelling where execution stops, if the learning process is accidentally suspended, possibly due to electricity power failure. The excellent results obtained motivate its application in [73] [89].

The new class of temporal probabilistic modelling using the EDBN architecture presented in this thesis has great universality of applications to wider real-life areas, with the added benefit of being able to accommodate all users. We relaxed the limited expressive power on the previous DBN work by completely emerging the network structures and the CPTs of DBN models with integration of scalability. This has been avoided by existing DBN modelling approaches for convenient modelling. Thus, the impact of effective monitoring and good planning is often felt in environments using sophisticated computational intelligent systems, such as the EDBN.

9.2 Recommendations

Due to the universality of our modelling architecture, we strongly recommend its application for use in research and industry especially when environmental situations become too complex for human comprehension. The situational awareness analysis of our technologies will be very beneficial to water quality managers, robotics systems, business and marketing departments, manufacturing, the medical field, and other areas where detailed understanding and anticipatory planning are highly crucial.

Individuals (such as mobile researchers, small to medium scale companies, etc.) who keep massive datasets on single-user machines, who are unfortunate enough to be in a sophisticated networking environment or who cannot afford to set one up, can now apply an economically scalable DMMAL framework to overcome computational intensity (or NP-hard) problems to a minimum acceptable level.

9.3 Suggested Future Work and Open Problems

The two broad directions for further work in this thesis are proving of more concepts and extensive applications of the EDBN architecture and/or its components.

As a proof of concept, we are committed to keep improving on the performance of the EDBN architecture as technology advances. It will be interesting to see what happens to any complex systems in what we term 'emergent past situation awareness' (EPSA). This enables one to predict and reason over time, and understand what the situation of an environment in the past was. For instance, the cosmologists may want to know more about situations in the galaxy of stars many years ago. As applications to the same complex environments, will the prediction results that will be obtained from the EPSA be equivalent to the results that will be obtained from the EFSA?

As extensive applications, we want to render the technology implementations from the EDBN architecture more robust and applicable to very many diverse situations for possible future usages in research and industries – commercialization, open source, etc. We have a vested interest in applying the DMMAL framework to solve computational intensity problems, which are also challenges in diverse research fields and industries.

In computational biology, there is an ongoing demand for computational intelligence models, which can learn, remember and predict patterns efficiently to solve real life problems. Imagine that we are asked to develop causal models and predict relationships among different elements of protein sequences, ribonucleic (RNA) acid, deoxyribonucleic (DNA) acid, genetic materials, etc. from multivariate time series. There are some relatively new emerging prediction models, which can contribute to address this

problem besides the technologies in this thesis. It is an open problem to find a simple and efficient technique, and do comparative analysis in terms of protein prediction accuracy, computational intensity, etc. Newly proposed techniques, which elicit their algorithms from biological systems, learn, memorize, and predict patterns over time are artificial immune systems (AIS), conditional random fields (CRF), dynamic Bayesian Networks (DBNs), etc. If the results of these techniques agree or disagree on the protein sequences, it will be interesting to know which one will eventually be applied.

Having applied our EDBN technologies to over eight diverse real-life areas such as business intelligence, water quality management, project profitability analysis, securities in telecommunications, sensor networks, etc., other interesting application areas for our modelling approach are the banking industry, oil exploration, robotic systems, etc.

References

[1] Choudhury, T., Rehg, J., M., Pavlovic, V., Pentland, A. (2002) Boosting and Structure Learning in Dynamic Bayesian Networks for Audio-visual Speaker Detection, *Proceedings of IEEE 16th International Conference on Pattern Recognition*, Volume 3, Pages 789 – 794.

[2] Shenoy, P. and Rao, R., P., N. (2005) Dynamic Bayesian Networks for Brain-Computer Interfaces, *Advances in NIPS*, MIT Press, Volume 17, Pages 1265-1272.

[3] Murphy, K. (2002) Dynamic Bayesian networks representation, inference and learning, PhD thesis, UC Berkeley, Computer Science Division.

[4a] Bengio, Y. and Frasconi, P. (1996) Input/output HMMs for sequence processing. *IEEE Trans. on Neural Networks*, Volume 7, Pages 1231–1249.

[5a] Brand, M. (1996) Coupled hidden Markov models for modeling interacting processes. Technical Report 405, MIT Lab for Perceptual Computing.

[6a] Ghahramani, Z. and Jordan, M. (1997) Factorial hidden Markov models. Journal of *Machine Learning*, Volume 29, Pages 245–273.

[7] Deviren, M. and Daoudi, K. (2001) Structural Learning of Dynamic Bayesian Networks in Speech Recognition, *Proceedings of the 7th European Conference on Speech Communications and Technology (Eurospeech)*, Aalborg, Denemark.

[8] Russell, S. and Norvig, P. (2003) *Artificial Intelligence, A Modern Approach*, 2nd Edition, Prentice Hall Series Inc. New Jersey 07458.

[9] Frigault, M., Wang, L., Singhal, A., and Jajodia, S. (2008) Measuring network security using dynamic bayesian network, Proceedings of the 4th ACM workshop on Quality of protection, Alexandria, Virginia, USA, ISBN:978-1-60558-321-1, Pages 23-30.

[10] Chapelle, O. and Zhang, Y. (2009) A dynamic bayesian network click model for web search ranking, Proceedings of the 18th international conference on World wide web, ISBN:978-1-60558-487-4, Pages 1-10.

[11] Inomata, T., Naya, F., Kuwahara, N., Hattori, F. and Kogure, K. (2009) Activity recognition from interactions with objects using dynamic Bayesian network, Proceedings of the 3rd ACM International Workshop on Context-Awareness for Self-Managing Systems, ISBN:978-1-60558-439-3, Pages 39-42.

[12] Poropudas, J. and Virtanen, K. (2007) Analyzing air combat simulation results with dynamic Bayesian networks, Proceedings of the 39th conference on Winter simulation, ISBN:1-4244-1306-0, Pages 1370-1377.

[13] Shermin, A. and Orgun, M. A. (2009) Using dynamic bayesian networks to infer gene regulatory networks from expression profiles, Proceedings of the 2009 ACM symposium on Applied Computing, ISBN:978-1-60558-166-8, Pages 799-803.

[14] Brandherm, B., Prendinger, H. and Ishizuka, M. (2008) Dynamic Bayesian network based interest estimation for visual attentive presentation agents, Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems, Volume 1, ISBN:978-0-9817381-0-9, Pages 191-198.

[15] An, X., Jutla, D. and Cercone, N. (2006) Privacy intrusion detection using dynamic Bayesian networks, Proceedings of the 8th international conference on Electronic commerce: The new e-commerce: innovations for conquering current barriers, obstacles and limitations to conducting successful business on the internet, ISBN:1-59593-392-1, Pages: 208 – 215.

[16] Endsley, M., R. (2000) Theoretical underpinnings of situation awareness: a critical review, *In M. R. Endsley & D. J. Garland (Eds.) Situation Awareness Analysis and measurement*, Mahwah, NJ: Lawrence Erlbaum Associates, Pages 3- 32.

[17] Silva, E., Plazaola, L. and Ekstedt, M. (2006) Strategic Business and IT Alignment: A Prioritized Theory Diagram, *In Proceedings of PICMET*, Turkey.

[18] Camponovo, G., Osterwalder, A. and Pigneur, Y. (2003) Assessing a complex, uncertain and disruptive technology environment for better IT alignment, *International workshop of Utility, Usability and Complexity of Emergent IS.*

[19] Brooks, T. and Davis, M. (1994) Are Your Phone Bills Fraud Free? *Security Management*, Volume 38, Pages 67-68.

[20] Chickering, D., Heckerman, D. and Meek, C. (2004) Large-Sample Learning of Bayesian Networks is NP-Hard, *The Journal of Machine Learning Research*, Volume 5, MIT Press, Pages 1287 – 1330.

[21] Koivisto, M. and Sood, K. (2004) Exact Bayesian Structure Discovery in Bayesian Networks, *The Journal of Machine Learning Research*, Volume 5, MIT press, Pages 549–573.

[22] Newman, D., Smyth, P., and Steyvers, M., (2006) Scalable Parallel Topic Models, *Journal of Intelligence Community Research and Development.*

[23] Larranaga, P., Kuijpers, C., Murga, R. and Yurramendi, Y. (1996) Learning Bayesian Network Structures by Searching for the Best Ordering with Genetic Algorithms, *IEEE Transactions on Systems, Man, and Cybernetics*, Volume 26, Pages 487-493.

[24] Friedman, N. and Koller, D. (2003) Being Bayesian About Network Structure - A Bayesian Approach to Structure Discovery in Bayesian Networks, *Journal of Machine Learning*, Springer, Volume 50, Pages 95 – 125.

[25] Myers, J., Kathryn, L. and DeJong, K. (1999) Learning Bayesian Networks from Incomplete Data using Evolutionary Algorithms. *Proceedings of the Genetic and Evolutionary Computation Conference*, Morgan Kaufmann Publishers, Volume 1, Pages 458-465.

[26] Witten, I., H. and Eibe, F. (2009) *Data Mining Practical Machine Learning Techniques and Tools*, University of Waikato - WEKA, Morgan Kauffman, San Francisco. http://www.cs.waikato.ac.nz/$\sim$ml/weka/

[27] GeNle 2.0 (2009) *Decision Systems Laboratory*, University of Pittsburgh, URL = http://genie.sis.pitt.edu

[28] Richardson, K., A. and Cilliers, P. (2007) Explorations in Complexity Thinking: *Pre-Proceedings of the 3rd International Workshop on Complexity and Philosophy*, ISBN 0-9791688-1-3, ISCE Publishers.

[29] Zhang, Y., Luke, E., A. (2005) Dynamic Memory Management in the Loci Framework. In: Sunderam, V. S., van Albada, G.D., Sloot, P.M.A., Dongarra, J. (eds.) ICCS 2005. LNCS, Springer, Heidelberg, Volume 3515, Pages 790–797.

[30] Moore, A. and Lee, M. S. (1998) Cached Sufficient Statistics for Efficient Machine Learning with Large Datasets, *Journal of Artificial Intelligence Research*, Volume 8, Pages 67 – 91.

[31] Newman, D., Hettich, S., Blake, C. and Merz, C. (2009) UCI *Repository of Machine Learning Databases*, University of California, Department of Information and Computer Science, Irvine, CA. http://www.ics.uci.edu/$\sim$mlearn/MLRepository.html

[32] RITA, (2009) Research and Innovative Technology Administration (RITA), U.S. Department of Transportation (US DOT) 1200 New Jersey Avenue, SE. Washington, DC. http://www.transtats.bts.gov/

[33] Delve Data Repository (2009) The University of Toronto, Toronto, Ontario, Canada. URL = http://www.cs.toronto.edu/~delve/

[34] Pearl, J. (1988) *Probabilistic reasoning in intelligent systems, networks of plausible inference*, Morgan Kaufmann Publishers.

[35] Murphy, K. (1998) A Brief Introduction to Graphical Models and Bayesian Networks, http://www.ai.mit.edu/~murphyk/bayes/bnintro.html.

[36] Chang, K., C. (2007) Almost Instant Time Inference for Hybrid Partially Dynamic Bayesian Networks, *IEEE Transactions on Aerospace and Electronics Systems*, Volume 43, Pages 13-22.

[37] Nilsson, N. (1998) *Artificial Intelligence, a new synthesis*, first edition, San Francisco USA, Morgan Kaufmann Publishers.

[38] Wai, L., Man, L., Kwong, S. and PoShun, N. (1999) Using Evolutionary Programming and MDL Principle for data mining of Bayesian networks, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE Computer Society, Volume 21, Pages 174-178.

[39] William, H., Haipeng, G., Benjamin, P. and Julie, S. (2002) A Permutation Genetic Algorithm For Variable Ordering In Learning Bayesian Networks From Data. *Proceedings of Genetic and Evolutionary Computation Conference*, Morgan Kaufmann Publishers Inc., ISBN: 1-55860-878-8, Pages 383-390.

[40] Dougherty, J., Kohavi, R. and Sahami, M. (1995) Supervised and unsupervised discretization of continuous features, *In 12th International Conference on Machine Learning*, Morgan Kaufmann Publishers.

[41] Waldron, M. and Penaloza, M. (2005) Genetic Algorithms as a Data Discretization Method, *Midwest Instruction and Computing Symposium conference*. http://www.micsymposium.org/apache2-default/mics_2005/

[42] Li, J., Liu, H., and Wong, L. (2003) Mean-entropy Discretized Features are Effective for Classifying High-dimensional Biomedical data, *Proceedings of the 3rd ACM SIGKDD Workshop on Data Mining in Bioinformatics*, Washington, DC, ACM press, Pages 17-24.

[43] Lee, C. (2007) A Hellinger-based discretization method for numeric attributes in classification learning, *Journal of Knowledge-Based Systems*, Volume 20, Pages 419-425.

[44] Ismail, M., K. and Ciesielski, (2003) An Empirical Investigation of the Impact of Discretization on Common Data Distributions, *In Proceedings of The International Conference on Hybrid Intelligent Systems (HIS'03): Design and Applications of Hybrid Intelligent Systems*, IOS Press, Pages 692-701.

[45] Olesen, K., G., Lauritzen S., L. and Jensen F., V. (1992) aHugin: A system creating adaptive causal probabilistic networks. *Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, Pages 223-229. http://hugin.sourceforge.net/download/.

[46] Alcobe, J. (2005) Incremental Methods for Bayesian Network Structure Learning, *AI Communications*, Volume 18, IOS Press, Pages 61-62.

[47] Krishnaswamy, S., Zaslavsky, A. and Loke, S., W. (2000) An architecture to Support Distributed Data Mining Services in E-Commerce Environments, *2nd Intl. Workshop on Advanced Issues in E-Commerce and Web-Based Information Systems*, ISBN: 0-7695-0610-0, Pages 239-246.

[48] Shen, J. and Lesser, V. (2006) Communication management using abstraction in distributed Bayesian networks, *Proceedings of the fifth international joint conference on autonomous agents and multiagent systems*, ISBN:1-59593-303-4, ACM, Pages 622 – 629.

[49] Jordan, M., I. (1998) *Learning in Graphical Models*, Kluwer Academic Publishers, ISBN 0-262-60032-3.

[50] Gell-Mann, M. (2001) *The Quark and the Jaguar, Adventures in the Simple and the Complex*, Abacus inc. Lancaster, London WC2E7EN.

[51] Osunmakinde, I., O. and Potgieter, A., (2007) Emergence of Optimal Bayesian Networks from Datasets without Backtracking using an Evolutionary Algorithm. *Proceedings of the Third IASTED International Conference on Computational Intelligence*, Banff, Alberta, Canada, ACTA Press, ISBN: 978-0-88986-672-0, Pages. 46 - 51.

[52] Mackay, D. (2003) *Information Theory, Inference and Learning Algorithms*, Cambridge Press.

[53] Jie, C., David, B. and Weiru, L. (2001) Learning Belief Networks from Data, An Information Theory Based Approach, *Proceedings of the Sixth International Conference on Information and Knowledge Management*, Las Vegas, Nevada, ACM, Pages 325-331.

[54] Ilori, S. and Akinyele, O. (1991) *Elementary Abstract & Linear Algebra*, Ibadan University Press, Nigeria.

[55] Shannon, T. and Zwick, M. (2002) Directed Extended Dependency Analysis for Data Mining, *In Proceedings of 12th International World Organization of Systems and Cybernetics and 4th International Institute for General Systems Studies Workshop*, Pittsburgh.

[56] Hollmen, J., Tresp, V., Taniguchi M. and Haft, M. (1998) Fraud Detection In Communications Networks Using Neural And Probabilistic Methods, *Proceedings of the 1998 IEEE Int. Conf. in Acoustics, Speech and Signal Processing (ICASSP'98)*, Volume 2, Pages 1241-1244.

[57] Potgieter, A., April, K., A., Cooke R., J., E. and Osunmakinde, I., O. (2009) Temporality in Link Prediction: Understanding Social Complexity. *Journal of Emergence: Complexity and Organization, E:CO* Issue Volume 11, No. 1, ISCE Publishers, ISSN: 1521-3250, Pages 83-96.

[58] Osunmakinde, I. O. (2006) Intelligent Detection of Anomalies in Telecommunications Customer Behaviour, *M.Sc. Thesis*, University of Cape Town, Computer Science Department.

[59] Lauritzen, S. and Spiegelhalter, D. (1988) Local Computations with Probabilities on Graphical Structures and Their Application on Expert Systems, *Journal of Royal Statistical Society*, Volume 50, Pages 157 – 224.

[60] Frayne, M. (2009) Interconnect billing–Making Telecommunications Work for Africa, *Annual Telecommunications Report*, Intec Telecom Systems.
http://www.connect-world.com/Articles/old_articles/MikeFrayne.htm

[61] Gillian, D. (1999) The changing face of fraud: phone fraud, *3rd National Outlook Symposium on Crime in Australia:* Mapping the Boundaries of Australia's Criminal Justice System, Canberra.

[62] Bay, D. and Schwabacher, M. (2003) Mining distance-based outliers in near linear time with randomization and a simple pruning rule. *In Proceedings of 9th annual ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Pages 29-38.

[63] Osunmakinde, I., O. and Potgieter, A. (2007) Immediate Detection of Anomalies in Call Data - An Adaptive Intelligence Approach, *Proceedings of Southern African Telecommunications Networks and Applications Conference (SATNAC)* Mauritius. ISBN: 978-0-620-39351-5.

[64] Kou, Y., Lu, C. Sirwongwattana, S. and Yo-Ping, H. (2004) Survey of Fraud Detection Techniques. *Networking, sensing and Control, IEEE International Conference*, Volume 2, Pages 749-754.

[65] Michael, H., Lambert, D., Pinheiro, C., and Sun, D. (2002) Detecting Fraud in the Real World, *Handbook of massive data sets*, Kluwer Academic Publishers, Pages 911-929.

[66] Bounsaythip, C. and Rinta-Runsala, E. (2001) Overview of Data Mining for Customer Behaviour Modelling. *Technical Report*, TTE1-2001-18, VTT Information Technology, Finland.

[67] Spiegel, M. R. (1972) *Schaum's Outline Series, Theories and Problems of Statistics*, McGRAW-HILL Publishing, NY.

[68] Osunmakinde, I., O. and Potgieter, A. (2008) Astute Decisions In Business Intelligence Using Temporal Probabilistic Reasoning, Competitive Research, *Proceedings of the 20th Annual Conference and Festival of the Southern Africa Institute for Management Scientists* (SAIMS), Gauteng, South Africa. ISBN: 978-1-86854-729-6.

[69] Balikuddembe, J., K., Osunmakinde, I., O., and Potgieter, A., E. (2008) Software Project Profitability Analysis Using Temporal Probabilistic Reasoning. In *Proceedings of the 2008 Advanced Software Engineering and Its Applications*, ASEA. IEEE Computer Society, Washington, DC, ISBN: 978-0-7695-3432-9., Volume 00, Pages 99-102.

[70] Cheruiyot, D. (2008) An assessment of the onset of summer rainy season in Southern Africa – case study of Botswana, M.Sc. Thesis, Computer Science Department, University of Cape Town, South Africa.

[71] Osunmakinde, I., O. and Bagula, A. (2009) Emergent Future Situation Awareness: A Temporal Probabilistic Reasoning in the Absence of Domain Experts, *In M. Kolehmainen et al. (Eds.): Adaptive and Natural Computing Algorithms (ICANNGA), Lecture Notes in Computer Science (LNCS)*, Springer-Verlag Berlin Heidelberg, Volume 5495, Pages 340–349.

[72] Graham, R., M. (1975) *Principles of Systems Programming*, John Wiley & sons Inc. New York.

[73] Osunmakinde, I., O. and Bagula, A. (2009) Supporting Scalable Bayesian Networks Using Configurable Discretizer Actuators, *In M. Kolehmainen et al. (Eds.): Adaptive and Natural Computing Algorithms (ICANNGA), Lecture Notes in Computer Science (LNCS)*, Springer-Verlag Berlin Heidelberg, Volume 5495, Pages 323–332.

[74] Borsuk, M., E., Stow, C., A., Reckhow, K., H. (2002) Integrative environmental prediction using Bayesian networks: A synthesis of models describing estuarine eutrophication. *Proceedings of the 1st biennial meeting of the International Environmental Modelling and Software Society*, Volume 2, Pages 102 – 107.

[75] McLaren, T., S., Head, M., M. and Yuan, Y. (2004) Supply chain management information systems capabilities - An exploratory study of electronics manufacturers, *Journal Information Systems and E-Business Management*, ISSN 1617-9846, Volume 2, Pages 207-222.

[76] Pike, W., A. (2004) Modelling Drinking Water Quality Violations With Bayesians Networks. *Journal of the American Water Resources Association* (JAWRA), Volume 40, Pages 1563 – 1578.

[77] Cozman, F. (2009) *JavaBayes*, Bayesian Networks in Java, University of Sao Paulo. http://www.cs.cmu.edu/~javabayes/Home/

[78] Hiirsalmi, M. (2000) Method feasibility study: Bayesian networks, *VTT Information Technology*, Espoo 35 p; MODUS-Project Waste Water Case Study, Research Report, Finland.

[79] Sanguesa, R. and Burrell, P. (2000) Application of Bayesian Network Learning Methods to Waste Water Treatment Plants, *Journal of Applied Intelligence*, Springer Netherlands Publishers, Volume 13, Pages 19 – 40.

[80] Bhattacharyya, S. (2000) Evolutionary algorithms in data mining: Multi-objective performance modelling for direct marketing, *Proceedings of the sixth International Conference on Knowledge Discovery Data Mining*, Pages. 465 – 473.

[81] Wong, M., L. and Leung, K., S., (2004) An efficient data mining method for learning Bayesian networks using an evolutionary algorithm-based hybrid approach, *IEEE Transactions on Evolutionary Computation*. Volume 8, Pages 378-404.

[82] Osterwalder, A and Pigneur, Y. (2003) Towards business and information systems fit through a business model ontology, *Strategic Management Society Conference*, Baltimore.

[83] Kardaras, D. and Mentzas, G (1997) Using fuzzy cognitive maps to model and analyse business performance assessment, *In Proceedings of the 2nd Annual International Conference on Industrial Engineering Applications and Practice II*, Pages 63-68.

[84] Petzold, J. (2005) Andreas Pietzowski, Faruk Bagci, Wolfgang Trumler and Theo Ungerer, Prediction of Indoor Movements Using Bayesian Networks, *Lecture Notes in Computer Science*, Springer Publishers, Volume 3479, Pages 211-222.

[85] Osunmakinde, I., O. and Potgieter, A. (2008) Emergent Situation Awareness Using Temporal Probabilistic Reasoning, *Journal of Pattern Recognition*, Elsevier, submitted for publication.

[86] R Development Core Team, (2008) R: *A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-07-0, http://www.R-project.org.

[87] Guo, H., Hou, W., Yan, F., and Zhu, Q. (2004) A Monte Carlo sampling method for drawing representative samples from large databases, *Proceedings of the 16th International Conference on Scientific and Statistical Database Management*, IEEE Computer Society, ISBN: 0-7695-2146-0, Pages: 419 – 420.

[88] Chickering, D., M. (2002) Learning Equivalence Classes of Bayesian-Network Structures, *Journal of Machine Learning* Research Volume 2, Pages 445-498.

[89] Osunmakinde, I., O. and Bagula A. (2009) Scalable Economic Framework for The Emergence of Bayesian Networks from Massive Datasets, submitted for publications.

[90] Osunmakinde, I., O. and Bagula, A. (2009) EMERGENT SITUATION AWARENESS OF DRIVABLE ROUTES FOR AUTONOMOUS ROBOTS USING TEMPORAL PROBABILISTIC REASONING, *Proceedings of the 14th IASTED International Conference on Robotics and Applications*, Cambridge, Massachusetts, USA. ACTA press, ISBN: 978-0-88986-813-7.

Appendix A

Implementation Visualizations – EDBN Technologies

A.1: Control Center for the ESA, EFSA and DMMAL Technologies

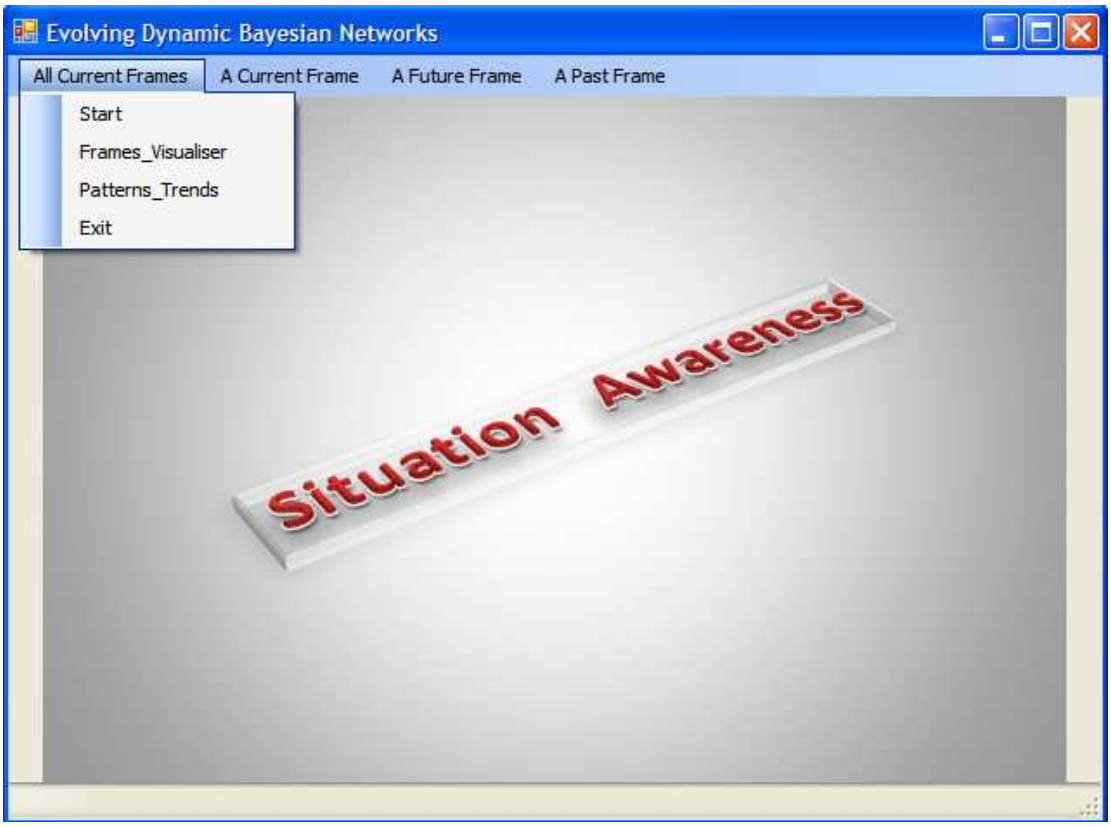


Figure A.1**:** By observing the concept of reusability of software components, the functionalities of these technologies use this control centre for their various problem solving approaches

A.2: Interface for Temporal Probabilistic Reasoning over time

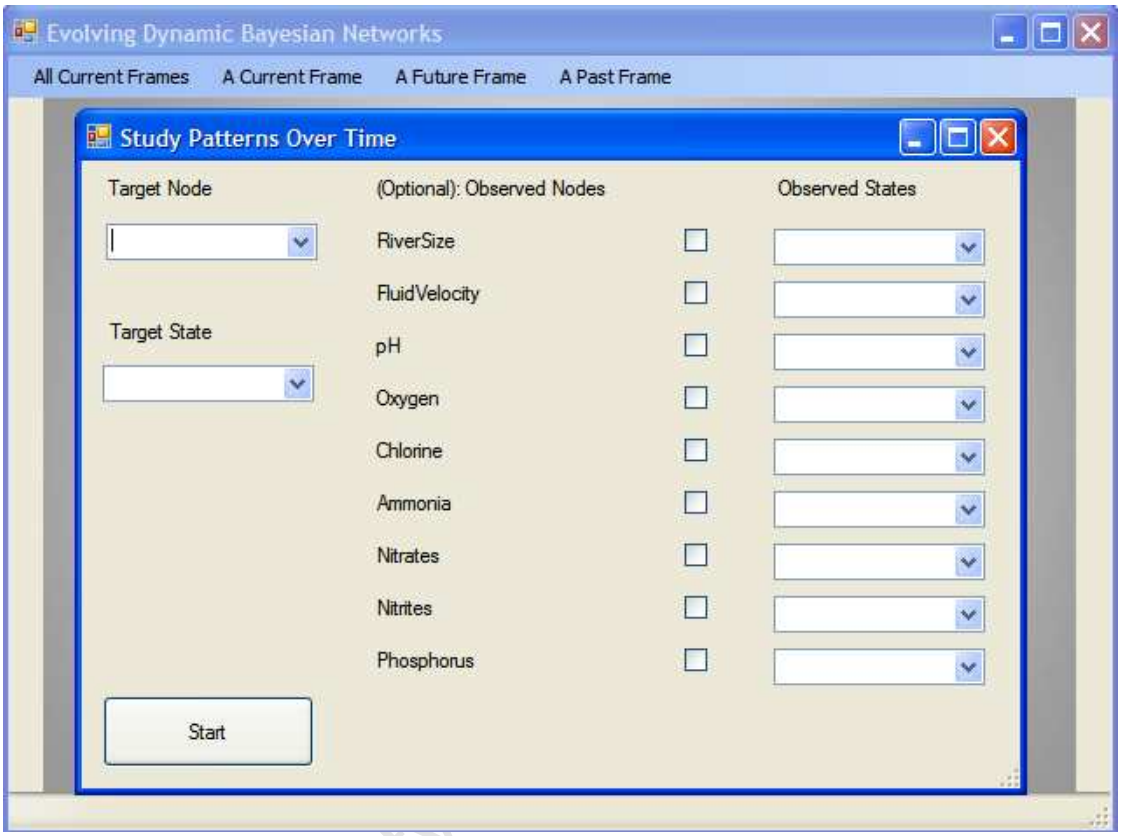


Figure A.2**:** This software environment reasons about uncertainties of hidden situations embedded in any environment of interest, such as the water quality management, business intelligence, project profitability analysis, etc.

A.3: Sample Result of Temporal Probabilistic Reasoning over time

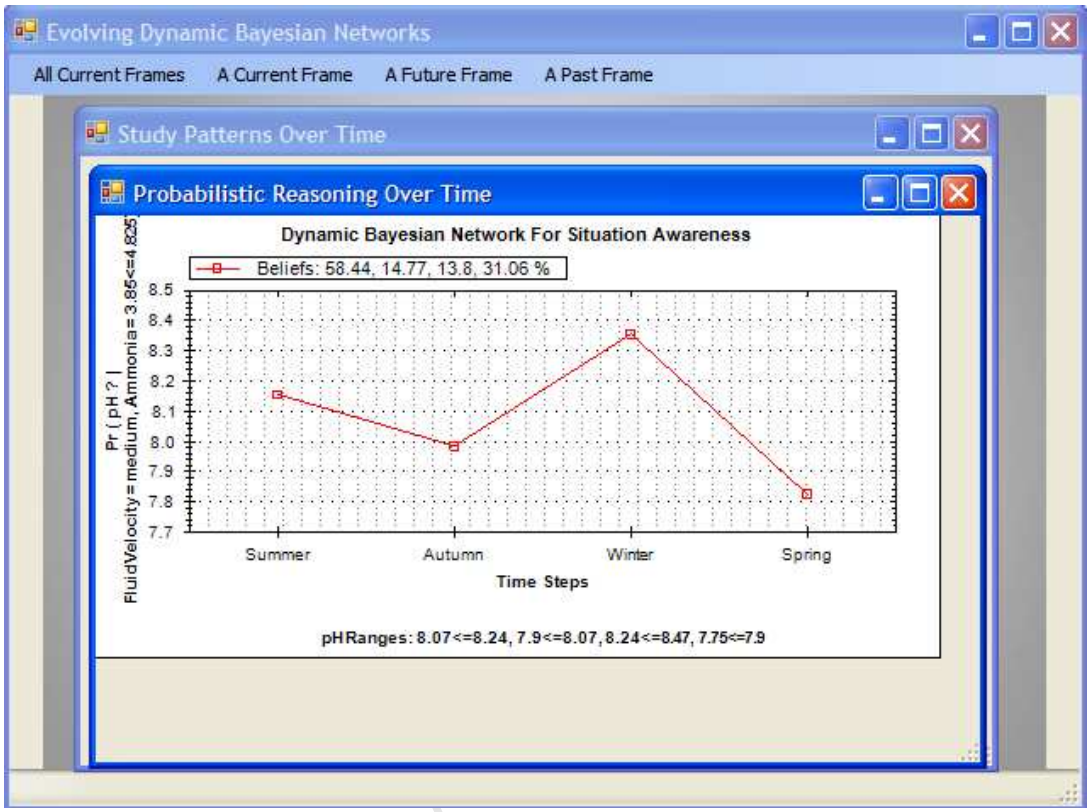


Figure A.3: The software reveals sample hidden situation over seasons from a complex water quality environment on the EDBN platform

Appendix B

Sample Real-Life Environments Captured as MTS and Prediction Results

B.1 Sample Multivariate Time Series Captured from European Rivers

Season	RiverSize	FluidVelocity	pH	Oxygen	Chlorine
Summer	large	low	8.8	8.8	43
Summer	large	medium	7.91	6.2	151.83299
Summer	large	high	8.4	10.29	5.326
Summer	large	medium	8.2	9.5	2.2
Summer	large	medium	8.3	10	3.86
Summer	large	high	8.1	10.2	7.613
Summer	large	low	8.4	8.2	23.25
Summer	large	medium	8.5	7.9	12.444
Summer	large	medium	8.5	6.7	82.852
Autumn	small	medium	8.1	11.4	40.02
Autumn	small	medium	8.06	9	55.35
Autumn	small	high	8.05	10.6	59.067
Autumn	small	high	7.55	11.5	4.7
Autumn	small	high	7.75	10.3	32.92
Autumn	small	high	6.8	11.1	9
Autumn	small	high	7.4	12.5	13
Autumn	small	medium	8.4	9.9	34.5
Autumn	small	high	8	12.1	66.3
Autumn	small	medium	7.29	11.21	17.75
Autumn	small	medium	5.7	10.8	-9
Autumn	small	medium	6.6	11.3	-9

Figure B.1**:** Perceived chemical substances and other variables describing water quality of European rivers. This is a complex environment because the variables of description are dependent on one another and there are multitudes of hidden situations embedded therein, which require to be revealed by sophisticated technologies such as the EDBN.

B.2 Sample Multivariate Time Series about Rainfall Onsets

MONTH	MTLY_RR	Atl_SST_Anom	Ind_SST_Anom	SOI	T_3_SOI
1	141.5	-0.081	-0.329	0.3	0.9
2	28.5	-0.15	-0.63	1.9	1.7
3	30.5	-0.286	-0.498	2.1	2.1
4	18	-0.099	-0.358	1.7	0.3
5	4.5	0.096	-0.133	0.7	1.9
6	0	-0.143	-0.181	0.1	2.1
7	0	-0.163	-0.224	0.1	1.7
8	0	-0.261	-0.364	1.3	0.7
9	8	-0.35	-0.314	1.6	0.1
10	58	-0.32	-0.296	1.7	0.1
11	99.5	-0.577	-0.371	0.5	1.3
12	24	-0.715	-0.407	0	1.6
1	157	-0.438	-0.31	0.4	1.7
2	103.7	-0.219	-0.269	0.8	0.5
3	53.5	-0.576	-0.263	0.1	0
4	6	-0.379	-0.443	-0.4	0.4
5	9.5	-0.207	-0.18	-2.1	0.8
6	0	-0.169	0.02	-1.1	0.1
7	0	-0.15	-0.104	-1.9	-0.4
8	0	-0.045	-0.114	-1	-2.1
9	1	-0.05	0.164	-1.6	-1.1

Figure B.2: Perceived weather characteristics of rainfall onsets for predicting farmers' planting dates. This is another complex environment because the variables of description are dependent on one another and there are multitudes of hidden situations embedded therein, which require to be revealed by sophisticated technologies such as the EDBN.

B.3: Prediction Directions on Diabetes Datasets

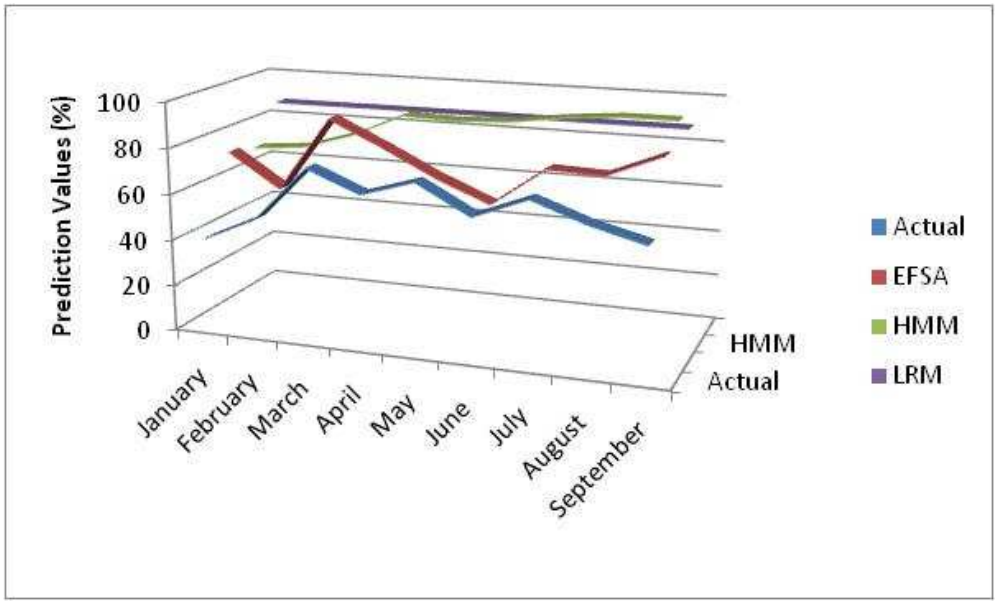


Figure B.3: Predictive Directions of the EFSA have a High Correlation with the Patterns of the Actual Results on Diabetes better than other Models. The LRM and the HMM are almost converging towards the end of the year.

B.4: Prediction Directions on Rainfall Datasets

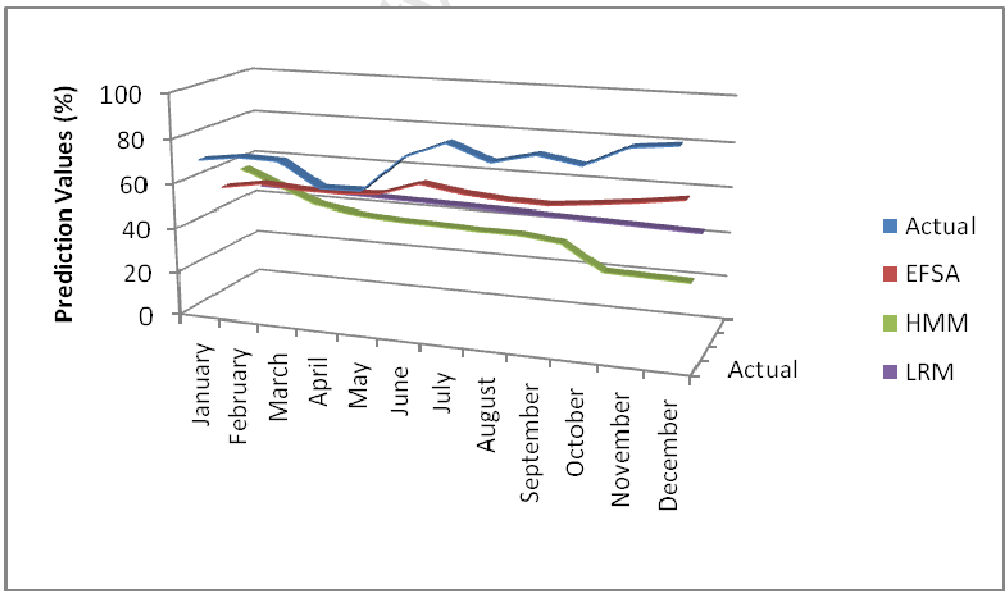


Figure B.4: Predictive Directions of the EFSA have a Good Correlation with the Patterns of the Actual Results on Rainfall better than other Models. The LRM is almost converging towards the end of the year.