AN ADAPTIVE AGENT ARCHITECTURE FOR EXOGENOUS DATA SALES FORECASTING

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE,
FACULTY OF SCIENCE
AT THE UNIVERSITY OF CAPE TOWN
IN FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Jonathan Jedeikin
November 2006

Supervised by
Dr. Anet Potgieter and co-supervised by Prof. Kurt April

UNIVERSITY OF CAPE TOWN

Acknowledgements

I would like to thank my supervisors, Anet and Kurt, for their help and input. A special thanks to Anet for all her hard work, support and care. Thanks for being so approachable, for listening to all my worries (of which there were many during the early months) and treating them with thought and consideration.

Thanks to the South African National Research Foundation for their generous financial assistance.

Thanks to Braam and Yvonne at Meatpackers for their assistance, and to the South African weather bureau for their cooperation.

Finally, a big thank you to my family, who have been so supportive during my studies, and to my friends for keeping me sane (most of the time).

Abstract

In a world of unpredictability and complexity, sales forecasting is becoming recognised as essential to operations planning in business and industry. With increased globalisation and higher competition, more products are being developed at more locations, but with shorter product life-cycles. As technology improves, more sophisticated sales forecasting systems are developed which require increasing complexity.

We turn to adaptive agent architectures to consider an alternative approach for modelling complex sales forecasting systems. This research proposes modelling a sales forecasting system using an adaptive agent architecture. It additionally investigates the suitability of Bayesian networks as a sales forecasting technique. This is achieved through BaBe, an adaptive agent architecture which employs Bayesian networks as internal models.

We develop a sales forecasting system for a meat wholesale company whose sales are largely affected by exogenous factors. The company's current sales forecasting approach is solely qualitative, and the nature of their sales is such that they would benefit from a reliable exogenous data sales forecasting system. We implement the system using BaBe, and incorporate a Bayesian network representing the causal relationships affecting sales.

We introduce a learning adjustment component to adjust the estimated sales towards closer approximations. This is required as BaBe is currently unable to use continuous data, resulting in a loss of accuracy during discretisation. The learning adjustment additionally provides a feedback aspect, often found in adaptive agent architectures. The adjustment algorithm is based on the mean error calculation, commonly used as sales forecasting performance measures, but is extended to incorporate a number of exogenous variables.

We test the system using the holdout procedure, with a 5-fold cross validation data-splitting approach, and contrast the accuracy of the estimated sales, provided by the system, with sales estimated using a regression approach. We additionally investigate the effectiveness of the learning adjustment component.

Overall, the system is found to be preferable to regression analysis, for the given data, and Bayesian networks prove to be desirable approaches for sales forecasting. The adjustment algorithm is effective in adjusting sales towards more accurate forecasts where sales remain consistent or follow an upward or downward trend. The system itself operates effectively as an adaptive agent architecture, satisfying the required criteria identified in our research.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Traditional artificial intelligence approaches deal with systems which obey a predefined set of cause and effect rules. They are isolated from their environment and incapable of adapting to changing external conditions. This is acceptable for simple systems, which exhibit consistent and predictable behaviour, but is less desirable for systems which require complex behaviour that is dependent on external factors.

A relatively new area of research, commonly called *reactive planning* (Brooks, 1991), considers systems which are situated in their environment. Their actions are directly dependent on the state of the environment, as they react to external influencing factors. Adaptive agent architectures model such systems through the use of simple components known as agents. They are effective in implementing reactive systems, capable of adapting over time.

Sales forecasting systems are necessary, in organisations, to anticipate sales and conduct planning. They attempt to predict future sales based on past sales, patterns or trends. Exogenous data forecasting techniques consider external factors influencing sales. They determine causal relationships between sales and exogenous factors, and attempt to forecast sales according to changes in these factors. These systems lend themselves to be modelled using an adaptive agent architecture. In this research we consider a new approach of exogenous data sales forecasting based on adaptive agent architectures, using Bayesian Networks and BaBe – an adaptive agent architecture.

In the following sections we present our motivation for this research and discuss the objectives which we wish to achieve, as well as the scope and limitations of the project. We conclude with a brief outline of this dissertation.

1.1 Motivation

As technology improves and global industry expands, sales forecasting continues to gain importance and increase in complexity. According to Lapide (2006: 22), "companies are forecasting more often, using the forecasts more to drive operational planning, and using more fact-based quantitative methods in generating forecasts." Lapide (2006) identifies increased complexity and globalisation as one of the reasons for the "evolution of the sales forecasting function." The trend towards greater competition in recent years has resulted in a vast increase in the number of products being sold, but also more frequent new product introductions and shorter product life-cycles. This has led to increased complexity in sales forecasting as "demand forecasting [sales forecasting] needs to be done for more products, at more locations, and for more business units, as well as with less historical data, given the greater numbers of short life-cycle fashion-oriented products (Lapide, 2006: 27)."

The increasing complexity of sales forecasting suggests the investigation into adaptive agent architectures as part of a sales forecasting approach. A forecasting system that uses external data to update an internal model[1], in order to make sales predictions, exhibits the behaviour of a reactive planning system and can be modelled as one using an adaptive agent architecture.

In addition, we consider a Bayesian network ideally suited as an internal model for exogenous data sales forecasting, as it has the ability to represent complex interrelationships among exogenous factors and determine causal probabilities of sales.

1.2 Objectives

The objectives of this research are twofold, namely:

1.2.1 Can a Sales Forecasting System Be Modelled Using An Adaptive Agent Architecture?

By modelling a sales forecasting system with an adaptive agent architecture we will introduce a new approach to sales forecasting, and present a further application of the reactive planning

[1] An internal model refers to a model, structure, or schema that a system uses for decision making. This concept will be elaborated on in the Adaptive Agent Architectures chapter.

approach. The system must incorporate the common characteristics of adaptive agent architectures identified in the literature.

1.2.2 Are Bayesian Networks Suitable Models For Sales Forecasting?

Bayesian networks have not been widely explored as techniques for sales forecasting. We investigate the current sales forecasting approaches and suggest why Bayesian networks should be ideally suited to sales forecasting. We then seek to determine the performance of a sales forecasting system that employs a Bayesian network as its internal model.

1.3 Scope and Limitations

We focus on a specific company, Meatpackers, rather than developing a generic sales forecasting system. This has the advantage of providing real-world data, however, we are limited to the data which is available to us. The reliability and consistency of the data could have a large impact on this research. Additionally, our conclusions might be relevant to Meatpackers, but not necessarily to other companies. Further investigations should be conducted on various companies to support this research.

1.4 Dissertation Outline

The layout of this dissertation is as follows:

- In chapters 2 to 4 we present the background theory for this research.
    - Chapter 2 discusses Adaptive Agent Architectures, including their characteristics. It then presents the theory behind BaBe, an adaptive agent architecture which we employ for developing the system.
    - Chapter 3 explains Bayesian network theory, including Probability theory and Bayesian inference.
    - Finally, Chapter 4 presents the common approaches for sales forecasting, and recognised sales forecasting performance measures. We then discuss the

suitability of a sales forecasting system to adaptive agent architectures and explain why Bayesian networks are ideally suited for sales forecasting.

- We present the research methodology employed for conducting this project in Chapter 5. We discuss our research strategy, as well as decisions made for testing; company selection; data collection; data pre-processing; and evaluating results.

- In Chapter 6 we discuss our considerations in designing the system. We present Meatpackers and identify the factors affecting their sales. We then display the Bayesian network to be used as the internal model, representing the causal relationships among exogenous factors. Finally, we devise the requirements for the high level workings of the system.

- In Chapter 7 we explain how the system was implemented, with emphasis on the behaviour of the competence agents – the agents responsible for the high level behaviour of the system in achieving a desired competence. We additionally introduce a learning adjustment component, based on feedback learning, which aims to increase the speed of learning and adjust the estimates towards more accurate results.

- Chapter 8 provides the results of our testing of the practical component of this research. This concerns the effectiveness of a Bayesian network for sales forecasting and the performance of the learning adjustment component. We discuss the results in terms of some of the performance measures highlighted in the Sales Forecasting chapter.

- In Chapter 9 we analyse our system in terms of its adherence to the requirements of an adaptive agent architecture. We pay particular attention to Brooks's (1991) characteristics of reactive planning and discuss additional characteristics identified in the Adaptive Agent Architecture chapter.

- We conclude with Chapter 10, providing a summary of the work done and the findings of this research, as well as looking at limitations and suggesting possible future work.

Chapter 2

Adaptive Agent Architectures

Traditional artificial intelligence methods are based on a fixed, predefined set of rules mapping cause to effect. This implies a static and predictable environment. However, in a real world environment, which is both dynamic and unpredictable, these methods are inadequate. Brooks (1991: 3) describes an approach commonly referred to as "reactive planning", in which "[artificial] intelligence can be reactive to the dynamic aspects of the environment." He describes the requirements that "a mobile robot operate on time scales similar to those of animals and humans, and that intelligence be able to generate robust behavior in the face of uncertain sensors, an unpredicted environment, and a changing world (Brooks, 1991: 2)." This can be achieved through the use of agents in *reactive* and *adaptive* agent architectures.

This chapter presents a brief summary of agent architectures, before discussing in more depth reactive and adaptive agent architectures. Finally, we describe BaBe, an adaptive agent architecture.

2.1 Agent Architectures

Stacey (1996: 10) describes *agents* (components) that "interact with each other according to sets of rules that require them to examine and respond to each other's behaviour in order to improve their behaviour and thus the behaviour of the system they comprise." These agents can be implemented in computer based systems using *agent architectures* – software engineering models of agents (Wooldridge & Jennings, 1995). Bobrow, Marks and Weld (2006) explain "the architecture of an agent is the computational structure that, along with the more dynamic knowledge represented within it, generates the agent's behavior in the environment."

Agent architectures are typically categorised into three categories: deliberative, reactive and adaptive.

The deliberative agent architecture uses an internal model to create control strategies – a planned sequence of actions required to achieve a task. Wooldridge (1995: 42) describes a deliberative agent as possessing an "explicitly represented, symbolic model of the world" in which "decisions are made via symbolic reasoning." Deliberative agent architectures are isolated from the world in that changes to the external environment would have no bearing on the agents' actions. The internal model remains constant irrespective of any changes to its environment.

The reactive agent architecture is considered to be *situated* (Brooks, 1991) in the world, as agents react to changes in their environment. Typically no internal model is used and no control strategies exist. A reactive agent would choose the next action given the current state of its environment. Nwana (1996: 26) explains that "they act/respond in a stimulus-response manner to the present state of the environment in which they are embedded."

In both deliberative and reactive architectures, no learning occurs. Adaptive agent architectures are characterised by their ability to learn from experience. As with the reactive agent architecture, agents are considered to be situated in the world. Actions are made based on the current state of the environment, as well as the agents' experience. Internal models or schemas are required in order to make decisions. A system that uses an adaptive architecture is able to learn and adapt over time.

Potgieter (2004: 65) describes the difference between deliberative, reactive and adaptive agent architectures to be that of "knowledge vs. control." She explains that in a deliberative architecture, the "states of the environment are separate from the control," making it impossible for them to react to changes in the environment. In a reactive or adaptive agent architecture, the states of the environment are part of the control, allowing the agents to react immediately. One can think of the environment as forming part of the reactive or adaptive architecture, whereas it is isolated from the deliberative architecture.

2.2 Properties of Adaptive Agent Architectures

Brooks (1991) describes characteristics common to both adaptive and reactive agent architectures: Stituatedness, Embodiment, Intelligence and Emergence.

2.2.1 Situatedness

Agents are situated in the world. Their behaviour is directly influenced by their environment, as they react to changes external to the system without dealing with "abstract descriptions" (Brooks, 1991: 3).

2.2.2 Embodiment

Brooks describes embodiment in the context of robots. He explains how robots "have bodies and experience the world directly". Agents would need to have sensors to enable them to interact directly with their environment. Potgieter (2004: 6) explains how agents' "actions are part of the dynamic interactions with the world, and this changes their experience of the world."

2.2.3 Intelligence

Agents are observed to be intelligent. The intelligence arises through agent interactions and their impact on their environment.

2.2.4 Emergence

The intelligence of the system emerges through the interactions with the environment, and between agents.

Baas and Emmeche explain *observation mechanisms* to introduce a mathematical framework for defining emergence. This is expressed as follows:

Let

$\{S_i\}$ $i \in I$ be a family of agents;

Obs^1 be an observation mechanism measuring the properties of the agents;

and Int^1 represent interactions among agents.

Then, through the interactions, a new structure, S^2 is generated, where S^2 is expressed as a relationship between the agents, the observation mechanism and the interactions:

$$S^2 = R\ (S_i^1, Obs^1, Int^1)$$

S^2 is known as an "emergent structure" (Baas et al.: 3) subject to a new observation mechanism, Obs^2.

Then, for some property, P:

$$P \text{ is an emergent property } \Leftrightarrow [\ P \in Obs^2\ (S^2) \text{ and } P \notin Obs^2\ (S_i^1)\ ]$$

Essentially, a property is emergent if it is observed in a system, but is not apparent in the constituent elements of the system.

2.2.5 Self-organisation

We additionally identify self-organisation as a characteristic of reactive and adaptive agent architectures. In order to exhibit emergence, the agents should be self-organised. The simple agents interact to produce an emergent behaviour, rather than a controlled, predefined behaviour enforced by an authority. Chiva-Gomez (2004: 708) explains that "new behaviour model patterns appear as a consequence of agent interaction." There is no single agent that determines the overall behaviour of the system.

Adaptive agent architectures additionally have the following characteristics:

2.2.6 Internal Models

Holland (1995) describes how an agent must select patterns (or regularities) from its input stream and integrate them by altering its internal model. These changes in the model must enable the system to anticipate the consequences that follow when the pattern is again encountered.

In general, Holland (1995: 33) states that a structure is an internal model if "we can infer something of the agent's environment merely by inspecting that structure." An internal model therefore reflects a relationship between a system and its environment. Holland (1995) introduces a further requirement of an internal model, stipulating that that this structure should also determine the agent's behaviour. The model is therefore effective if it enables the system to anticipate useful future consequences.

2.2.7 Feedback and Adaptation

The effects of an agent's actions are fed back to the agent and this affects the agent's behaviour in the future (Steel, 1999). Gell-Mann and Millikan (1990: 10) claim this to be the "essential feature of adaptation, evolution or learning."

Mauboussin (1997: 11) describes a feedback system as "one in which the output of one iteration becomes the input of the next iteration." Feedback can be positive (amplifying) or negative (dampening). Positive feedback adds the output to an input, whereas negative feedback subtracts the output from an input. In an adaptive agent architecture, amplifying feedback allows the system to evolve, while dampening feedback ensures a degree of stability. For this reason, dampening feedback is sometimes referred to as "balancing feedback". According to (Glasgow, Longstaff & Sibthorpe, 2004: 3) feedback loops generate both change and stability. They explain that "they fuel the interdependence of the system by keeping the parts synchronised, and simultaneously support evolution of the system by providing impetus and resources for adaptation."

2.3 BaBe – An Adaptive Agent Architecture

Potgieter (2004) describes BaBe - an adaptive agent architecture based on Bayesian Behaviour networks.

2.3.1 Bayesian Behaviour Networks

A Bayesian behaviour network models the regularities in the input stream of an adaptive agent architecture. It is similar in concept to a *Behaviour network* (Maes, 1989). Behaviour networks use *competence modules* to select appropriate actions according to the state of the environment. The competence modules react to the states of the environment, as well as to the spreading of *activation energy* along links of the Behaviour network (Maes, 1989). Similarly, Bayesian Behaviour networks use *competence sets* for high-level decision making. A competence set is defined by Potgieter (2004: 83) as a "tuple (C_i, A_i), where C_i is a set of constraints on a subset of nodes and their states in a Bayesian behaviour network, and A_i is the set of actions that must be executed if all the constraints in C_i are met." A node is therefore constrained by a number of conditions, C_i ,which when observed produce the actions defined by A_i.

Whereas Behaviour networks allow for reasoning based only on Boolean variables, Bayesian behaviour networks incorporate Bayesian networks into their structure. They are therefore able to perform powerful probabilistic reasoning in the presence of uncertainty (Potgieter, 2004). Bayesian Networks will be discussed in depth in the following chapter.

Figure 1 displays Potgieter's (2006) illustration of a Bayesian behaviour network. It models the browsing behaviour of users visiting an electronic bookstore website. The figure demonstrates a Bayesian network with competence sets defining constraints and actions for various nodes.

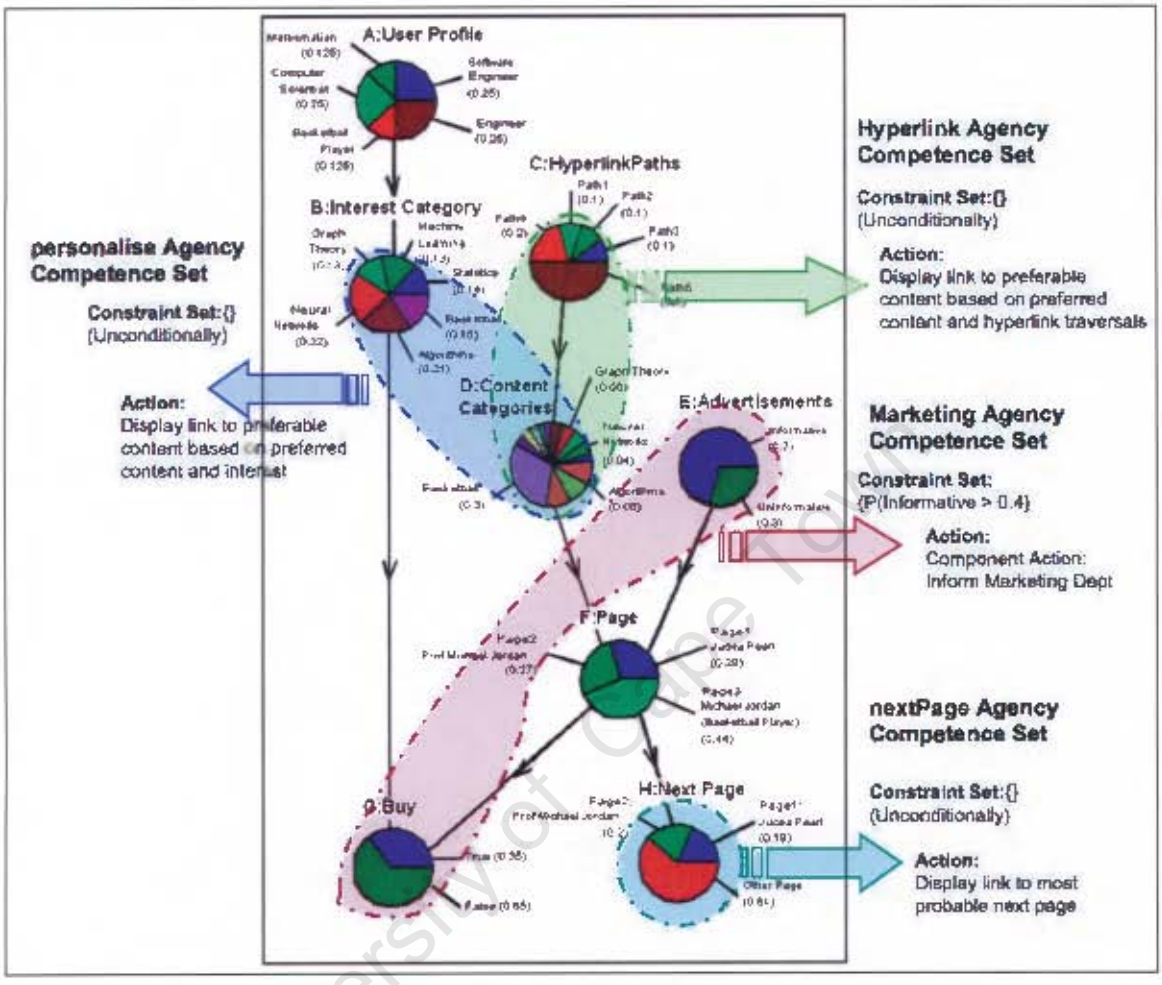

Figure 1: Bayesian behaviour network

Source: AEG Potgieter (Personal communication, October 2006)

2.3.2 The BaBe System

The BaBe system uses simple agents which interact with each other and the environment, as shown in **Figure 2**.

The internal model, situated in the environment, is a Bayesian behaviour network, reflecting probabilistic relationships among variables. The structure of the model is learnt and trained through the interactions of real-time feeder agents and learning agents. The feeder agents retrieve information from the various data stores occurring in the environment, and provide data-groups representing groups of information. The learning agents incrementally learn by mining patterns from the data groups.

Competence agents perform the high level actions required to achieve a desired behaviour. They typically manipulate the internal model through reasoning agents which perform Bayesian inference or other reasoning strategies.

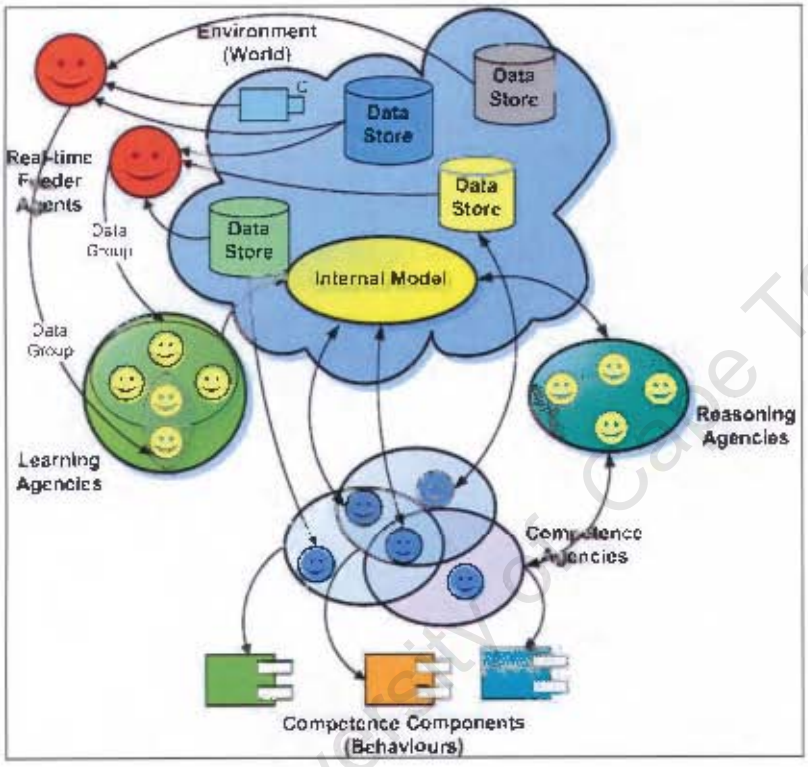

Figure 2: BaBe components

Source: AEG Potgieter (Personal communication, September 2006)

2.4 Conclusion

In a dynamic world, traditional artificial intelligence methods are inadequate, as they focus only a predefined set of rules. Deliberative architectures use internal models, maintaining a symbolic representation of the world. The model, however, remains static irrespective of changes to the system's environment. While they are able to conduct control strategies, they are not considered to be situated in the world and such strategies might not be appropriate to the current state of the environment.

Reactive architectures do not maintain internal models, but are sensitive to changes in the system's environment. They are, however, unable to learn from experience. Adaptive agent architectures are characterised by their ability to learn over time. They maintain internal models for decision making, but continually adapt their models to the state of their environment.

Reactive and Adaptive agent architectures are both characterised by Situatedness, Embodiment, Intelligence and Emergence. Situatedness implies that agents are situated in the world, and are directly affected by changes in their environment. Embodiment suggests that agents interact with the world, experiencing their environment directly. Intelligence is observed through the interactions of agents among each other and with their environment. This is regarded as Emergence, where the actions among simple agents give rise to a complex global behaviour. Self-organisation is an additional characteristic, required to achieve emergence, as the agents cannot be governed by a predefined set of actions. Adaptive agent architectures additionally maintain internal models which use feedback to adapt and learn over time.

BaBe is an adaptive agent architecture, capable of learning over time and reacting to changes in its environment. It uses Bayesian behaviour networks for powerful probabilistic reasoning. These networks are similar to Maes's (1989) Behaviour networks, but incorporate Bayesian networks into their structure.

The BaBe system uses simple agents which interact with each other and their environment. Feeder agents and learning agents interact to learn the structure of the internal model – typically a Bayesian network. Competence agents perform high level actions by interacting with reasoning agents, which manipulate the Bayesian network – setting evidence and querying nodes.

Chapter 3

Bayesian Networks

Graphical models combine probability theory and graph theory to deal with uncertainty and complexity. A graph structure provides an interface by which humans can model relationships among variables and design efficient algorithms, while probability theory ensures that the system as a whole is consistent and provides ways to relate models to their underlying data (Murphy, 1998).

A Bayesian network is a form of a *directed* graphical model whose structure represents causal relationships among variables. Their directed nature makes Bayesian networks more desirable than *undirected* models such as Markov networks. According to Pearl (1998: 116) directed graphs permit one to "distinguish genuine dependencies from spurious dependencies induced by hypothetical observations." For this reason, Bayesian networks are more popular in the fields of Artificial Intelligence and statistics (Murphy, 1998).

This chapter outlines the theory of Bayesian networks and discusses their usefulness as internal models in adaptive agent architectures. Before discussing the structure and workings of Bayesian networks, we provide a brief introduction to probability theory, which leads up to Bayes's theorem – the underlying probabilistic theorem that governs calculations in Bayesian networks.

3.1 Probability Theory, Bayes's Theorem and Conditional Independence

This section describes the fundamental probability theory required to understand Bayesian networks. It discusses the basic axioms of probability theory, before presenting Bayes's theorem for determining causal probabilities.

3.1.1 What Is Probability?

The probability of some event, A, is the likelihood of A occurring. One can say that $P(A = x)$ means A has x probability of occurring. It is important to note that each probability reflects a degree of belief rather than a frequency of occurrence (Pearl, 1988).

3.1.2 Basic Axioms

The basic axioms to probability theory are discussed in (Pearl, 1988). These are described below.

Beliefs must obey three basic axioms of probability:

$0 \leq P(A) \leq 1$, (1)

The probability of any event must lie between 0 and 1.

$P(certainty) = 1$, (2)

If an event is certain the probability is 1.
(In contrast an impossible event has a probability of 0).

$P(A \text{ or } B) = P(A) + P(B)$ if A and B are mutually exclusive. (3)

The probability of any of a set of events occurring is the sum of the beliefs assigned to its non-intersecting members.

The complement of an event, ¬A, refers to the probability of the event A not occurring.
It follows from (2) that:

$$P(A) + P(\neg A) = 1 \quad (4)$$

This says that one can be certain that either A will occur, or A will not occur.

Pearl (1988) explains that any event, A, can be written as the union of the joint events (A and B) and (A and ¬B), therefore:

$$P(A) = P(A, B) + P(A, \neg B) \quad (5)$$

Where $P(A, B)$ is the shortened form of $P(A \text{ and } B)$

Generalising this for a set B_i, $i = 1,2...n$ of exhaustive and mutually exclusive propositions, $P(A)$ can be computed as:

$$P(A) = \sum_{i=1}^{n} P(A, B_i) \tag{6}$$

3.1.3 Bayes's Theorem

Bayes's Theorem, developed by Rev. Thomas Bayes in 1763, calculates the probability of an event occurring, given additional evidence (Niedermayer, 1998). It is derived from the equation of conditional probability as shown below.

The equation of conditional probability is expressed as the following:

$$P(A \mid B) = \frac{P(A,B)}{P(B)} \tag{7}$$

Where $P(A \mid B)$ means "the probability of A, given some evidence B".

Hence:

$$P(A, B) = P(A \mid B)\, P(B) \tag{8}$$

It follows from Equ. 6 and 8, that the probability of an event A can be computed from a set of evidence Bi, $i = 1,2...n$ as follows:

$$P(A) = \sum_{i=1}^{n} P(A \mid B_i) P(B_i) \tag{9}$$

This states that the belief of an event A is a "weighted sum over the beliefs in all the distinct ways that A might be realised (Pearl, 1988: 31)."

Since $P(A, B)$ is the same as $P(B, A)$, the "inversion formula" can be written as:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)} \tag{10}$$

This equation is recognised as Bayes's theorem and is often expressed as:

$$P(H \mid e) = \frac{P(e \mid H)P(H)}{P(e)} \tag{11}$$

Where *H* refers to the *hypothesis* and *e* refers to the *evidence*. *P(H | e)* is sometimes called the *posterior* probability and *P(H)* is known as the *prior* probability (Pearl, 1988).

3.1.4 Conditional Independence

According to Nilsson (1998: 324), "a variable, *V*, is conditionally independent of a set of variables, *Vi*, given a set *Vj*, if *P(V| Vi, Vj) = P(V|Vj).*" V_i then has no effect on the probability of *V* when given V_j.

The variables $V_1 \ldots V_k$ are mutually conditionally independent, given a set *V*, if each of the variables is conditionally independent of the others, given *V* (Nilsson, 1998).
Then:

$$P(V_1, V_2, \ldots\ldots, V_k \mid V) = \prod_{i=1}^{k} P(V_i \mid V)$$

3.2 The Structure of a Bayesian Network

A Bayesian network is a directed acyclic graph, consisting of nodes and edges. The nodes represent events or random variables, while the edges represent causal dependencies between nodes. Nilsson (1998) stipulates that in a Bayesian network each node, V_i, is conditionally independent of any subset of the nodes that are not descendents of V_i, given the parents of V_i. This is expressed as $I(V_i, A(V_i) \mid P(V_i))$, where $A(V_i)$ is any set of nodes that are not descendents of V_i and $P(V_i)$ are the immediate parents of V_i.

In addition to the graph structure, the parameters of the model are specified. For each node there exists a conditional probability distribution. For discrete variables this in the form of a conditional

probability table (CPT) – a matrix that lists the probability that the child node takes on each of its values for each combination of its parents' values (Murphy, 1998).

Murphy (1998) provides a simple example of a Bayesian network in which all nodes may have only one of two values. This is shown in **Figure 3** below. The Bayesian network models the probability of the grass being wet, based on its influencing variables, i.e. whether the sprinkler is on; whether it has been raining; and whether it is cloudy. In the diagram, the rain node is conditionally independent of the sprinkler node, given the cloudy event, since the sprinkler node is not a descendent of the rain node and vice versa.

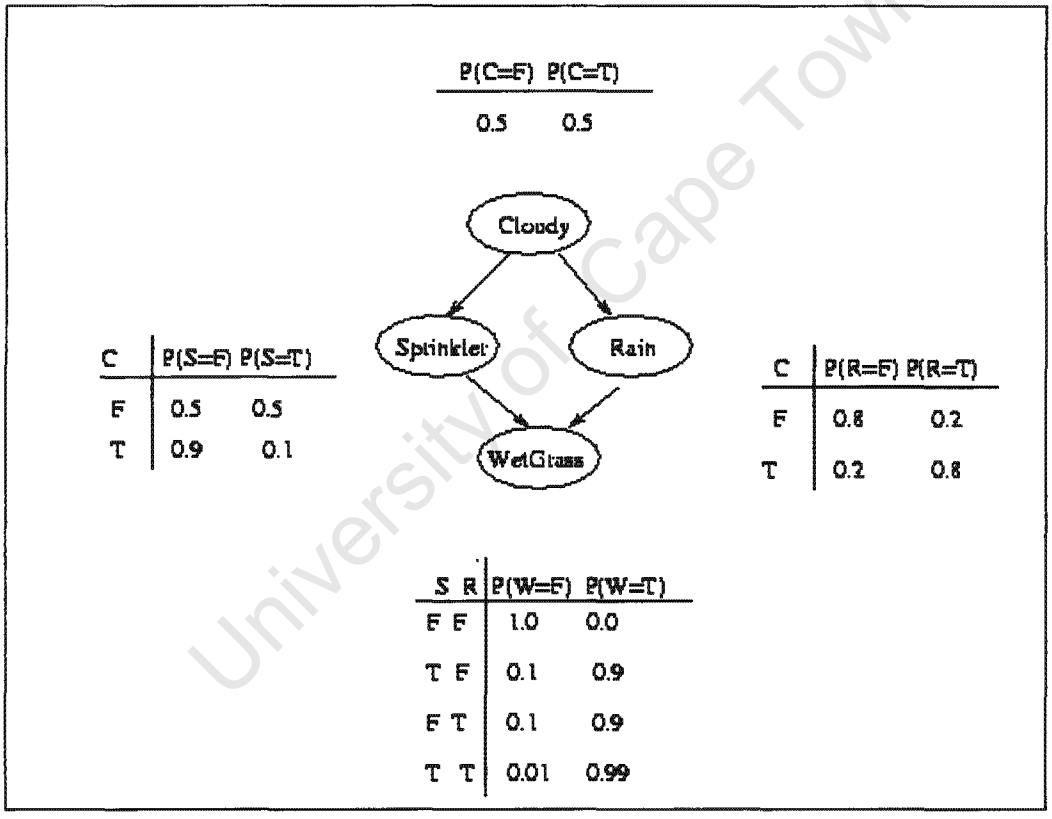

Figure 3: A simple Bayesian network

Source: Murphy (1998)

Using this Bayesian network one could infer the probability of the grass being wet, given the states of the sprinkler and rain nodes. For example $P(W = T \mid S = T, R = F)$ is 0.9. So, based on this Bayesian network, if one knows that the sprinkler is on and it is not raining, one could be

90% certain that the grass would be wet. Similarly, given the fact that it is cloudy, the probability that is raining is 0.8. This illustrates the concept of *Bayesian inference*.

3.3 Bayesian Inference

Bayesian inference refers to the process of computing the probability of each value of a node in a Bayesian network when other variables' values are known (Jensen, 1999). For each node, the posterior probability, $P(H \mid e)$, is calculated for the given evidence, e, using Bayes's theorem (Equ 11, above). This is a feature that makes Bayesian networks so powerful. Traditional inferential models do not permit the introduction of prior knowledge into the calculations (Niedermayer, 1998), but Bayesian networks allow one to apply knowledge towards backward or forward reasoning.

3.3.1 Causal or Top-Down Inference

Causal or top-down inference reasons about a node based on its evidence. The probability of an effect is calculated from given causes. Consider a Bayesian network identical to **Figure 3**, but with the probabilities of the events *Sprinkler* and *Rain* included. The probability that the sprinkler is on, $P(S)$ can be calculated as 0.3 (i.e. There is a 30% likelihood, at any given time, that the sprinkler will be on) and $P(R) = 0.5$. This is displayed in **Figure 4.**

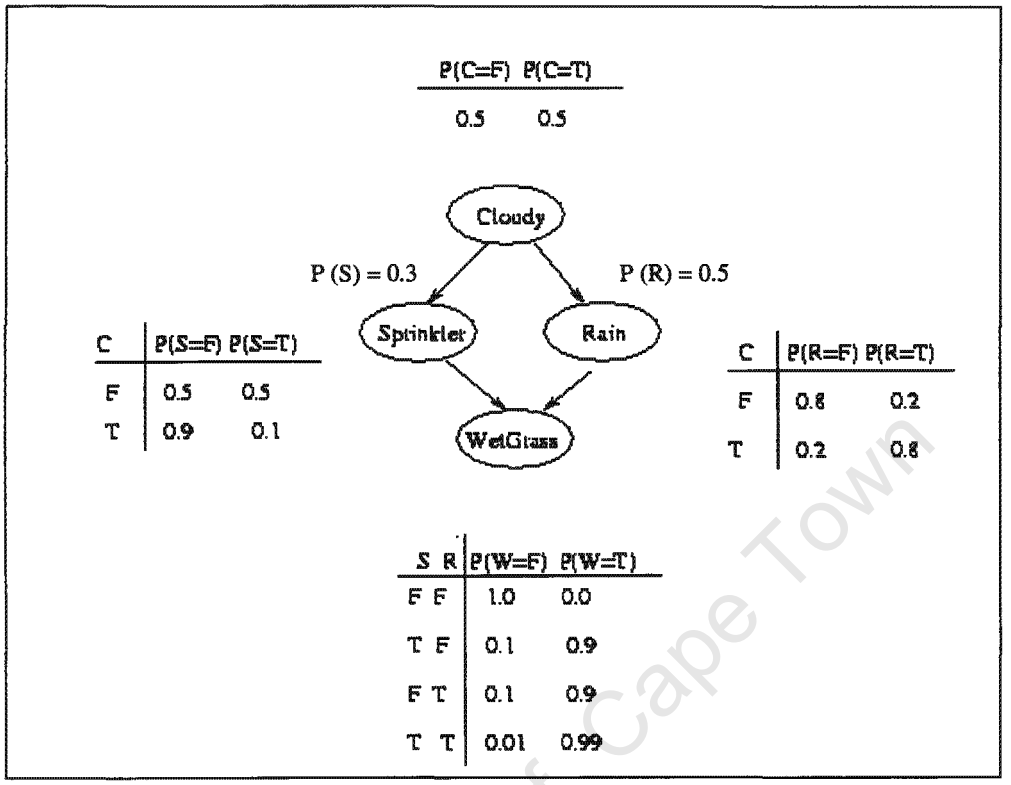

Figure 4: A simple Bayesian network with probabilities P(S) and P(R) given

As an example, one could use top-down reasoning to calculate the probability of the grass being wet, given the fact that it is raining. Nilsson (1998) explains how this is achieved.

The conditional probability of the grass being wet given that is raining, $P(W \mid R)$, can be expanded into its joint probabilities $P(W, S \mid R) + P(W, \neg S \mid R)$, using a variation of Equ. 5.

Then using a form of Equ. 10, one can express this as:
$P(W \mid S, R)\, P(S \mid R) + P(W \mid \neg S, R)\, P(\neg S \mid R)$

Since the sprinkler is assumed to be conditionally independent of rain in this Bayesian network, $P(S \mid R)$ is equal to $P(S)$, as evidence that it is raining would not alter the probability that the sprinkler is on. Similarly, $P(\neg S \mid R) = P(\neg S)$.

Then $P(W \mid S, R)\, P(S \mid R) + P(W \mid \neg S, R)\, P(\neg S \mid R)$
$= (0.99)(0.3) + (0.9)(0.7)$
$= 0.927$

3.3.2 Diagnostic or Bottom-Up Inference

Diagnostic or bottom-up inference reasons about a node based on its effects. The probability of a cause is calculated from the given effects. For example, the probability that is raining given that the grass is wet, $P(R \mid W)$, can be determined using bottom-up reasoning with Bayes's rule:

$$P(R \mid W) = \frac{P(W \mid R)P(R)}{P(W)} \quad \text{(see Equ. 11)}$$

3.3.3 Explaining Away

This is a combination of top-down and bottom-up reasoning. The probability of a node is determined from both causes and effects. Referring to **Figure 4**, this would be performed when wanting to calculate, for example, the likelihood that it is raining, given that it is cloudy and that the grass is wet ($P(R \mid C, W)$).

$$\text{Then } P(R \mid C, W) = \frac{P(C,W \mid R)P(R)}{P(C,W)} \quad \text{(Bayes's Rule)}$$

$$= \frac{P(C \mid W,R)P(W \mid R)P(R)}{P(C,W)} \quad \text{(see Equ. 8)}$$

3.3.4 Belief Propagation

Belief propagation (Pearl, 1988) uses a message-passing process to achieve probabilistic inference in Bayesian networks. Evidence is propagated through the network using π-messages and λ-messages. The π-messages propagate between each parent and its children, while the λ-messages propagate from the children to their parents.

Potgieter (2004) illustrates this algorithm with **Figure 5** and **Figure 6**.

In **Figure 5**, an arbitrary link XY divides the evidence in two. The evidence above the link is referred to as $e^+ xy$, while the evidence below the link is $e^- xy$. π-messages are propagated from X to its children Y. For each child, Y_i, the message passes a probability $P(X, e^+ xy_i)$. Similarly, λ-messages are propagated from each child node, Y_i, to its parent, X. This is the probability $P(e^- xy \mid X)$.

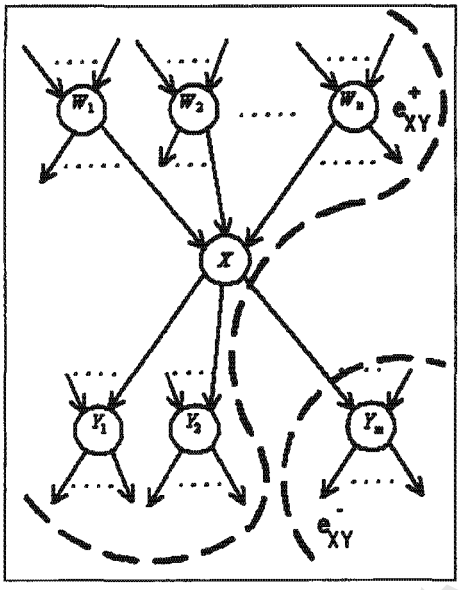

Figure 5: Separation of evidence by a link
Source: Potgieter (2004)

In **Figure 6**, an arbitrary node X divides the evidence into prior evidence ($e^+ x$) and observed evidence ($e^- x$). The prior evidence is the evidence that causes X, while the observed evidence is X's effects. $P(X \mid e^+ x)$ is calculated as the product of all the π-messages received from its parents, weighted by the conditional probability of X given its parents. $P(e^- x \mid X)$ is calculated as the product of all the λ-messages obtained from X's children. The *belief* of X is then calculated as a normalised product of its prior probabilities vector ($P(X \mid e^+ x)$) and its likelihood vector ($P(e^- x \mid X)$).

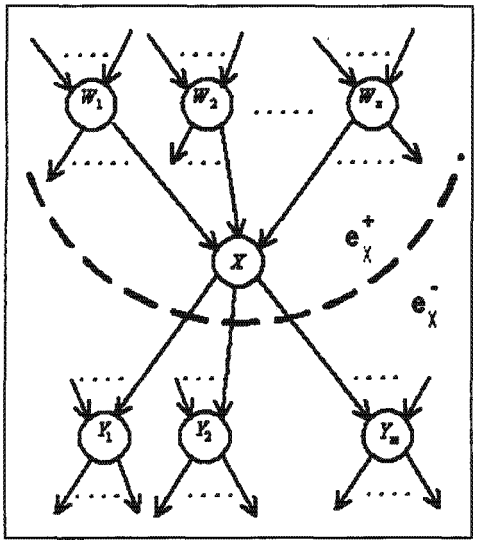

Figure 6: Separation of evidence by a node

Source: Potgieter (2004)

3.4 Bayesian Learning

It is possible to discover the structure of a Bayesian network and learn the conditional probability tables for each node, from a set of training data. Bayesian learning can be achieved even when nodes are hidden or there is missing data.

Murphy (1998) summarises Bayesian learning into the following cases:

Table 1: Cases of Bayesian learning

Structure	Observability	Method
Known	Full	Maximum Likelihood Estimation
Known	Partial	EM (or gradient ascent)
Unknown	Full	Search through model space
Unknown	Partial	EM + search through model space

These methods will not be discussed here, as a detailed study of Bayesian learning falls outside the scope of this research. For detailed descriptions of these methods, the reader is referred to (Murphy, 1998) and (Nilsson, 1998).

3.5 Dynamic Bayesian Networks

A dynamic Bayesian network is a Bayesian network reflecting temporal information in the form of individual time-slices. According to Murphy (2002: 1), "if all arcs are directed, both within and between slices, the model is called a dynamic Bayesian network (DBN)." The term dynamic Bayesian network was introduced as they model dynamic systems (Murphy, 2002). However Murphy (1998) explains how the term "temporal Bayesian network" would be more appropriate since the model structure does not change.

Hidden Markov models are the simplest case of Dynamic Bayesian networks, with one hidden node and one observed node per slice (Murphy, 1998). **Figure 7** illustrates a representation of a Hidden Markov model, adapted from (Murphy, 1998) and (MacDonald & Zucchini, 1997). The shaded nodes are the observed nodes. The square nodes are the hidden nodes, dependent on the observed nodes.

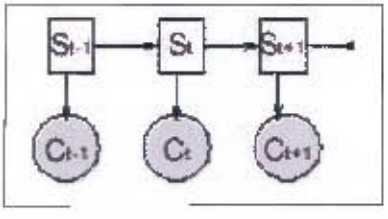

Figure 7: Graph representation of a Hidden Markov Model

3.6 Bayesian Networks as Internal Models

Potgieter (2004: 37) states that "Bayesian networks are ideally suited to be used as adaptive hyperstructures in internal models". She explains that Bayesian learning can be used to adapt to environmental changes, and belief propagation to reason about what action to take next.

Bayesian learning allows an agent to introduce new evidence, from its input stream, into its internal model. This evidence would affect the changes in its internal model described by Holland (1995). Bayesian inference would then alter the beliefs of each node and allow the system to make a more informed decision, based on probabilistic reasoning, when encountering the pattern again.

Feedback may be added to the Bayesian network in the form of evidence, enabling the system to adapt its internal model, based on the results of its own actions.

3.7 Implementation Considerations

Typically, an edge from node A to node B represents a cause and effect relationship, where the parent node, A, is the cause and the child node, B, is the effect. This is desirable to give an intuitive feel of the interrelationships between variables. However, often this is impractical to implement due to the demands of the CPTs. When each node has only two states, a CPT is of size 2^n rows, where n represents the number of immediate parents of a node. Already this indicates a swiftly increasing CPT size. Often nodes have more than two states – five to ten states per node would not be considered uncommon. The CPTs might then grow at a rate of 5^n or even 10^n. It is conceivable how a node with a high number of states and a high number of parents might grow excessively large.

Since Bayesian inference can be achieved using a top-down approach, bottom-up approach or a combination of the two, the direction of the edge could be reversed without affecting the calculations. One could sacrifice the intuitive representation of the Bayesian network with a more efficient structure. Consider the following example, shown in **Figure 8** .

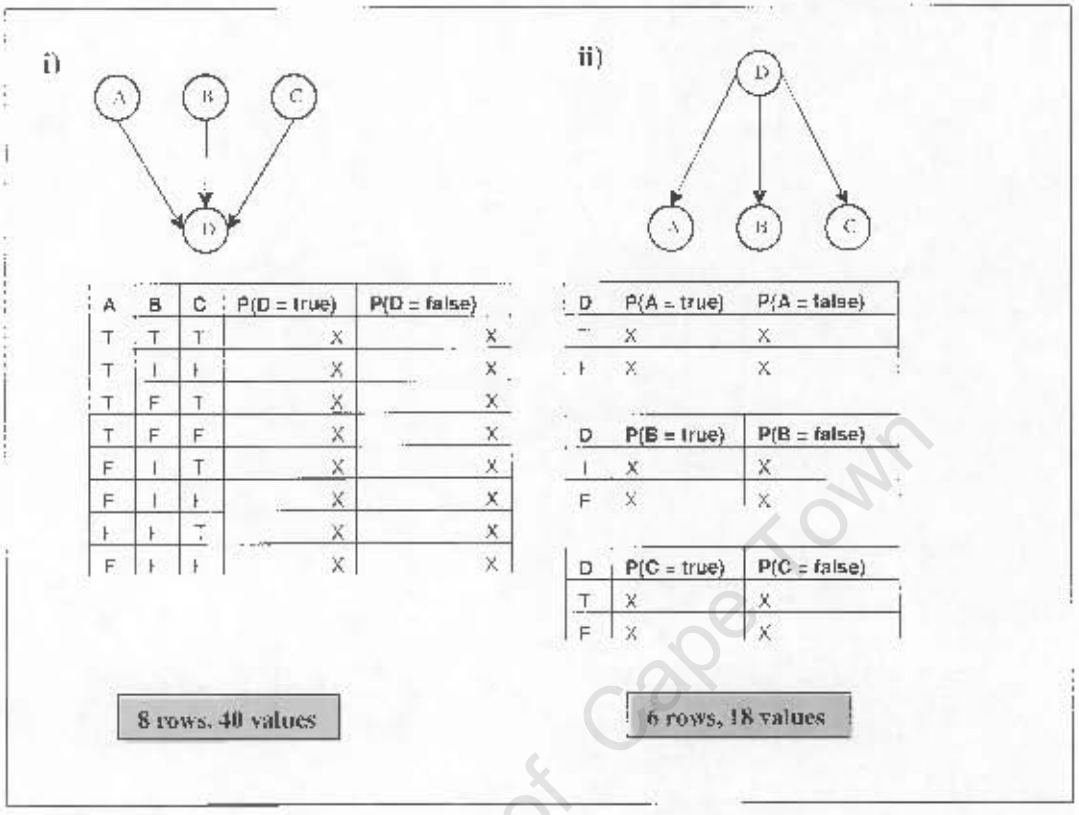

Figure 8: Changing edge direction

In the first representation, i, node D is expressed as the effect of nodes A, B and C. Its CPT requires 8 rows and a total of 40 values. In the second representation, ii, node D is expressed as the cause of nodes A, B and C. In this case, nodes A, B and C each requires a CPT. The total number of rows of their combined CPTs is 6, with 18 values.

We have shown that inference can be conducted top-down or bottom-up. We would therefore be able to calculate $P(D \mid A, B, C)$ as $\frac{P(A,B,C \mid D)P(D)}{P(A,B,C)}$

Where $P(A,B,C \mid D) = P(A \mid D)P(B \mid D)P(C \mid D)$ since A, B and C are mutually conditionally independent, given D.

We could therefore achieve the same inference with a more efficient structure. Careful consideration should be applied to determine an optimal Bayesian network structure.

3.8 Conclusion

Bayesian networks are directed graphical models. They provide an interface with which humans can model relationships among variables and perform probabilistic reasoning, based primarily on Bayes's theorem.

Bayesian inference allows for forward or backward reasoning, and enables the introduction of prior evidence for calculations. Bayesian learning allows for the discovery of both the structure and the conditional probability tables of a Bayesian network.

These features make Bayesian networks ideally suited as internal models for adaptive agent architectures. A system is able to change its structure and adapt its beliefs in order to respond to a pattern when it is again encountered. This satisfies Holland's (1995) requirements for the internal model of a system.

Feedback may be added to the Bayesian network in the form of evidence, enabling the system to adapt its internal model based on the results of its own actions.

Chapter 4

Sales Forecasting

Sales forecasting attempts to predict the expected level of sales for a company's goods or services, for some future period. Mentzer and Moon (2005: 9) define it to be "a projection into the future of expected demand, given a stated set of environmental conditions." Most companies perform some form of sales forecasting, whether it be formally or informally conducted. The simple act of a manager anticipating sales for the following day is a basic form of sales forecasting.

Whether companies maintain sophisticated forecasting techniques, or rely simply on experience and intuition, sales forecasting performs a crucial role in sales and operations planning. Shim (2000: 3) explains that a forecast is a "starting point for planning", and a method for reducing risk in decision making.

Furthermore, it is becoming recognised that an accurate and reliable forecasting system is necessary for any organisation. According to Mentzer et al. (2005: 11), "without accurate and credible estimates for demand, it is impossible for organisations to effectively manage their supply chains." They continue to describe how one might be so intent on developing a plan, that one simply assumes what sales would be, rather than "giving any concentrated thought and analysis to the market conditions that will be necessary to create this level of sales (Mentzer et al., 2005: 11)."

Hyndman, Makridakis and Wheelwright (1997: 5) state that "efficient use of resources requires the scheduling of production, transportation, cash and personnel" and identify sales forecasting as an "essential input to such scheduling."

This chapter describes the theory underlying sales forecasting, including forecasting techniques and performance measures. It then explains the suitability of an adaptive agent architecture for

implementing a sales forecasting system and the desirability of Bayesian Networks for sales forecasting.

4.1 Qualitative vs. Quantitative Forecasting

When a manager uses his/her experience, intuition or judgement to anticipate sales for the following day, s/he is, perhaps unwittingly, performing a form of *qualitative* sales forecasting. Qualitative techniques rely on human reasoning, intuition and experience to make real world decisions. Mentzer et al. (2005: 21) explain that they "take into account the full wealth of key personnel experience and require little formal data." Formal qualitative techniques include *role playing*, *intentions*, *expert opinions, Delphi* and *conjoint analysis*. All of these approaches attempt to draw from human knowledge, emotion or experience. In *role playing*, an administrator asks people to play roles and uses their decisions as forecasts (Scott, 2001). Using an *intentions* approach, marketers would ask consumers whether they would purchase a particular product, while *conjoint analysis* uses a survey based approach to obtain consumer input. *Expert opinions* are opinions provided by experts in the field. This is extended by the *Delphi* method - an approach that questions a group of experts, individually, about their perceptions of future events (Shim, 2000).

Qualitative techniques are desirable where little historical data is available, or in a rapidly changing environment in which current sales bear no relation to sales history. According to Shim (2000: 7), they can "identify systematic change more quickly and better interpret its effects on the future." However, these techniques are subject to a number of shortcomings. Apart from their requirement for considerable personnel time (Mentzer, et al., 2005), judgements often prove to be unreliable when the task is more complex or the environment is uncertain. When the acquisition of information relies on perception, pattern recognition or memory, forecasts are susceptible to human error (Scott, 2001). In addition, Shim (2000: 12) explains that "humans tend to be optimistic and underestimate the future uncertainty."

Quantitative approaches were introduced to provide more reliable forecasting than the existing qualitative methods. Quantitative forecasting draws on data and statistics to provide a numerical prediction, based on history rather than opinion. Hyndman et al. (1997) provide three conditions for when quantitative forecasting can be applied:

1. *Information about the past is available.*
2. *This information can be quantified in the form of numerical data.*
3. *It can be assumed that some aspects of the past pattern will continue into the future.*

One might wonder how we can learn from the past if everything constantly changes. Hyndman et al. (1997: 9) explain that "after some familiarity with data and forecasting techniques... it becomes clear that although nothing remains the same, history does repeat itself in a sense." Patterns or regularities appear to emerge from the data and existing relationships can be identified which influence sales.

Quantitative techniques are further divided into endogenous data (otherwise known as time-series) techniques and exogenous data (otherwise known as causal) techniques. Endogenous data techniques use only the history of sales, while exogenous data techniques consider other data, such as price, promotions, competitive actions, or economic measures to explain changes in sales (Mentzer, 2005).

The following section discusses common quantitative techniques and when these techniques should be applied. While such techniques are often preferable to qualitative approaches, one should not disregard the value of human judgement in many situations. In practice, a combination of both qualitative and quantitative forecasting is desirable (Shim, 2000).

4.2 Quantitative Forecasting Techniques

4.2.1 Endogenous Data (Time-series) Sales Forecasting

Time-series forecasts are based on the assumption that sales follow one or more of the data patterns: level, trend, seasonality and noise (or irregularity). *Level* represents some horizontal pattern, where sales remain mostly constant. *Trend* is a continuing upward or downward pattern, reflecting sales increase or decrease. *Seasonality* is a repeating pattern occurring within a year or less. Mentzer (2005: 19) explains that "the pattern of high sales in certain periods of the year and low sales in other periods repeats itself every year [where the pattern of seasonality is observed]." *Noise* is any random fluctuation which follows no predictable pattern.

The following figures illustrate the concepts of trend, seasonality and noise.

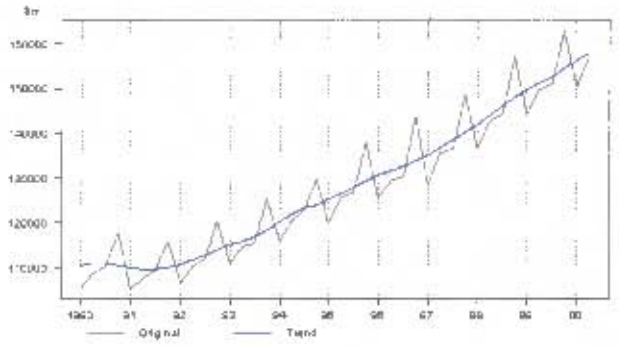

Figure 9: Trend

Source: Australian Bureau of Statistics (http://www.abs.gov.au)

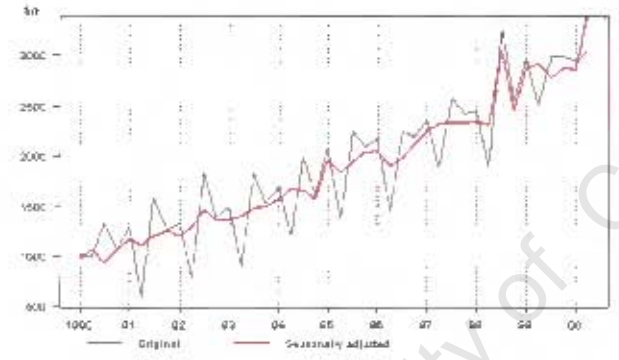

Figure 10: Seasonality

Source: Australian Bureau of Statistics (http://www.abs.gov.au)

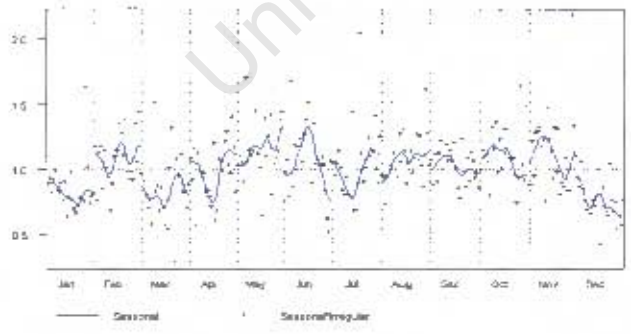

Figure 11: Noise

Source: Australian Bureau of Statistics (http://www.abs.gov.au)

With level, trend and seasonality, it is evident how patterns (or regularities) can be extracted from sales. In each of the cases, the sales are an emergent behaviour, resulting from the interactions of

a great number of contributing factors, although these factors are not considered in time-series prediction. Sales patterns exhibiting noise, as in **Figure 11**, likely contain a high degree of non-linearity as these sales often differ from their expected sales.

Fixed-Model Time-series Techniques (FMTS)

Fixed-Model techniques remain the same irrespective of any patterns in the data, unlike open-model techniques which first analyse the data to determine existing patterns. We now consider some fixed-model time-series techniques.

The simplest time-series approach, also known as the *naïve* approach, predicts sales for a period based on its previous period's actual sales.

A more sophisticated technique uses the average sales over a number of periods. Averaging dampens out any fluctuations, thereby omitting noise from the forecast. However, it additionally omits trend and seasonality (Mentzer et al., 2005). A *moving average* approach averages sales over only recent periods, rather than the entire sales history. This is able to maintain trend and seasonality. However, there remains a trade-off, as one is unsure of how many periods to include. When averaging over too few periods, irregularities still exist, but when using too many periods trend and seasonality patterns are dampened.

Exponential smoothing, previously known as *exponentially weighted moving average*, solves this problem by placing a greater weighting on more recent periods, while not omitting older periods. This is expressed in Equ. 12.

$$F_{t+1} = \alpha S_t + (1 - \alpha) F_t \qquad (12)$$

where

F_{t+1} is the forecast for the next period

F_t is the forecast for the given period

S_t is the sales for the given period

α is a constant value between 0 and 1

Each period, the forecast from the previous period has in it the forecasts (and the sales) from all previous periods. It essentially breaks down to the following:

$$F_{t+1} = \alpha S_t + (1 - \alpha)\alpha S_{t-1} + (1 - \alpha^2)\alpha S_{t-2} + \ldots + (1 - \alpha^N)\alpha S_{t-N} \quad (13)$$

where N is the number of past periods calculated.

The higher the value of α, the more weight placed on the last period's sales and the less weight placed on all the previous periods combined.

However, one is never certain of the appropriate value for α. A technique known as Adaptive Smoothing was introduced to attempt to automatically select the value of α (Mentzer et al., 2005). Other time-series techniques include *exponential smoothing with trend*, *exponential smoothing with trend and seasonality*, and *adaptive exponential smoothing with trend and seasonality*.

While the most commonly used approach is exponential smoothing, none of these techniques is regarded as preferable over any others, as a whole. The effectiveness of each technique depends on the time-series scenario for which each technique was designed (Mentzer et al., 2005).

Open-Model Time-series Techniques (OMTS)

OMTS techniques first analyse the patterns in the time-series to determine which exist (Mentzer et al., 2005). OMTS techniques include *Decomposition Analysis*, *Spectral Analysis*, *Fourier Analysis*, *Auto-regressive Moving Average (ARIMA)* and *the Box-Jenkins* method. These techniques are fairly complex and require considerable data history. They are therefore seldom used in practice and will not be discussed in depth in this research.

Time-series techniques should be regarded when there is substantial sales history data, but limited exogenous data.

4.2.2 Exogenous Data (Causal) Sales Forecasting

If one is to more accurately anticipate sales, one cannot consider sales history alone. Exogenous data sales forecasting recognises that there exist correlations, or cause-effect relationships, between sales and exogenous data (data apart from the actual sales themselves). Consider how factors such as promotions, competitive actions and advertising might affect sales.

We now discuss some exogenous data sales forecasting approaches, in the form of regression analysis, econometric forecasting and Markov models.

Regression Analysis

Regression analysis is the most common form of exogenous data sales forecasting. It uses a statistical approach to establish a relationship between sales and exogenous variables. Shim (2000: 45) explains it as "a statistical procedure for estimating mathematically the average relationship between a dependent variable and independent variables." When used in sales forecasting, the dependent variable is the quantity of sales, while the independent variables are the exogenous factors. Often correlation is used to determine the strength between variable relationships. The exogenous variables found to be closely correlated to sales should be used for sales forecasting (Mentzer et al., 2005).

The idea of regression analysis is to *fit* a line to the data. This is described by (Mentzer, et al., 2005: 115) as "finding the one line through the data that minimises the mean squared error, or minimises the sum total of the distance of each data point from the line."

Regression can be *simple* or *multiple*. Simple regression applies when a single independent variable and a single dependent variable exist (essentially when there is only one exogenous factor), while multiple regression considers multiple independent variables.

Figure 12 illustrates an example of how a regression line could be plotted to data points using simple regression.

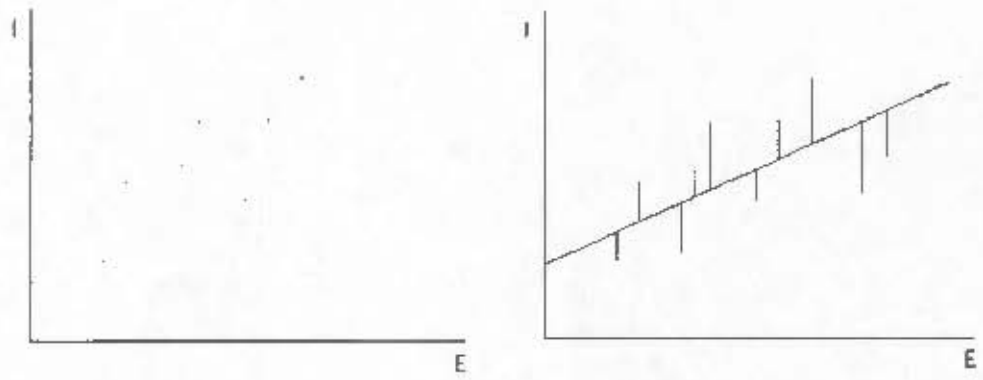

Figure 12: Plotting a linear regression line to data points

Source: Sykes.

Regression analysis works relatively well when independent variables have no correlation among each other. However, often these variables do affect each other. This is known as multi-collinearity and according to Shim (2000: 72) it is a "pitfall" because "equations with multi-collinearity may produce spurious forecasts."

Econometric Forecasting

The principle behind econometric forecasting is that everything in the real world depends on everything else (Hyndman et al., 1997). Econometric models generally consist of a number of simultaneous multiple regression equations, each with their own dependent and independent variables. Sales might, for example, be represented as a function dependent on price, advertising and promotions, while price could itself be a function of advertising, promotions and profit. Hyndman et al. (1997) provides a possible scenario, shown in **Figure 13**. This figure shows sales as a function of GNP, price and advertising. Advertising is a function of sales; and price is a function of production, cost, selling expenses, administrative overhead and profit.

sales = *f* (GNP, price, advertising)
production cost = *f* (number of units produced, inventories, labor costs, material cost)
selling expenses = *f* (advertising, other selling expenses)
advertising = *f* (sales)
price = *f* (production cost, selling expenses, administrative overhead, profit)

Figure 13: Possible econometric interrelationships

The main advantage to econometric methods is their ability to deal with interdependencies. Hyndman et al. (1997: 301) explain that they are "valuable tools for increasing the understanding of the way an economic system works and for testing and evaluating alternative policies." However, they are not always ideal for forecasting. Such methods are difficult to develop, and identifying complex interrelationships can be a challenging task.

Despite their complexity, econometric models often give very accurate results (Shim, 2000).

Markov Models

A Markov model determines a probability of moving from a given state to some other state. Probabilities are calculated using *transition matrices* - matrices describing the probability of moving from one state to another. Shim (2000: 21) explains that the Markov Model could be used to "predict market share by considering brand loyalty and switching behaviors." Markov models can be used effectively to investigate the impact of promotions or marketing strategies, by estimating their effect on market share.

MacDonald and Zucchini (1997) discuss Hidden Markov models for discrete-valued time-series forecasting, but do not consider their application to sales forecasting.

4.3 Performance Measurement

In an unpredictable world, it is not possible to anticipate the exact sales on every occasion. Mentzer et al. (2005) stress that sales forecasting should be measured in terms of its accuracy, rather than exactness.

Mentzer et al. (2005) identify three categories of forecasting measures: actual measures; measures relative to a perfect forecast; and measures relative to a perfect forecasting technique. This section will describe some common techniques, in each of these categories, used to determine the accuracy of sales forecasting methods.

4.3.1 Actual Measures of Forecasting Accuracy

Performance is measured in terms of error – the difference between actual sales and predicted sales for a given time period.

Mean Error

The mean error is a running average of how much the forecast has been off in the past. It is calculated as:

$$\text{Mean Error} = \text{ME} = \sum_{i=1}^{n} \frac{E}{n} \tag{14}$$

where E is the error and n is the given number time periods.

This might give an impression of how much the estimated sales generally differ from the actual sales. However, positive and negative values cancel each other out when summing the errors, giving a false impression of accuracy. Consider a result whose error is 10 and another, whose error is -10. This would produce a mean error of 0, indicating perfect prediction, when in fact both have been off by 10.

Mean Absolute Error

To overcome the problem with Mean Error, the absolute value of each error is used, in the equation:

$$\text{Mean Absolute Error} = \text{MAE} = \sum_{i=1}^{n} \frac{|E_i|}{n} \tag{15}$$

However, with only positive signs, one does not know whether the estimate was too high or too low. In addition, those months in which the forecasts were off only slightly count as heavily as months in which they differed substantially. If very large errors have a greater impact on company operations, it is preferable to place a higher weighting on larger errors (Mentzer et al., 2005).

Mean Squared Error

Each error is squared and an overall average is obtained in the equation:

$$\text{MSE} = \sum_{i=1}^{n} \frac{E^2}{n} \tag{16}$$

This has the effect of magnifying large errors and still keeps positives and negatives from cancelling each other out.

The problem with these approaches is that they do not give an indication of whether the forecast is good or bad.

4.3.2 Accuracy Measures Relative to a Perfect Forecast

Percentage Error

The error is expressed as a percentage, with the equation:

$$\text{PE}_t = \frac{(\text{Forecast}_t - \text{Sales}_t)}{\text{Sales}_t} \times 100 \tag{17}$$

MAPE

The average of the absolute values of each percentage error is obtained.

$$\text{MAPE} = \sum_{i=1}^{n} \frac{|PE_i|}{n} \tag{18}$$

However, this considers all percentage errors, regardless of how old they may be (Mentzer et al., 2005). To solve this problem one might use Year-to-date MAPE, a calculation that considers only the periods for the last twelve months.

These approaches provide comparable statistics but also do not indicate whether a forecast is good or bad.

4.3.3 Accuracy Measures Relative to a Perfect Forecasting Technique

Theil's U-statistic

Theil's U-statistic calculates a ratio of the accuracy of the forecasts relative to the naïve forecast's accuracy. If the ratio is greater than 1.0, it is less accurate than the naïve forecast and should be discarded (Mentzer et al., 2005). It follows that the smaller the U-statistic, the more accurate the forecast.

The U-statistic is expressed in (Hyndman et al., 1997) as:

$$U = \sqrt{\frac{\sum_{i=1}^{n-1}(FPE_{i+1} - APE_{i+1})^2/(n-1)}{\sum_{i=1}^{n-1}(APE_{i+1})^2/(n-1)}} \tag{19}$$

Where

$FPE_{i+1} = \frac{F_{i+1} - S_i}{S_i}$ is the forecasted relative change;

$APE_{i+1} = \frac{S_{i+1} - S_i}{S_i}$ is the actual relative change;

and F and S are the forecasted sales and actual sales, respectively.

This simplifies to

$$U = \sqrt{\frac{\sum_{i=1}^{n-1}\left(\frac{F_{i+1} - S_{i+1}}{S_i}\right)^2}{\sum_{i=1}^{n-1}\left(\frac{S_{i+1} - S_i}{S_i}\right)^2}} \tag{20}$$

Sales Forecasting Technique Accuracy Benchmark (SFTAB)

An alternative, simpler approach (proposed by Mentzer et al., 2005) calculates the MAPE of the forecast divided by the MAPE of the naïve forecast to achieve a similar effect to that of the U-

statistic. As with the U-statistic, a value under 1.0 indicates an acceptable result, while the lower the result the better.

4.4 Sales Forecasting as an Adaptive Agent Architecture

Consider a typical sales forecasting system. It is situated in the real world, processing data from companies and providing results of vital importance. Through masses of data it distinguishes patterns - whether based solely on sales using a time-series approach, or incorporating exogenous data with regression or econometric models. These are likely represented in a form of internal model or schema which assists in decision making and prediction. Over time it adapts its results to the current state of the market, perhaps through exponential smoothing or sophisticated regression analysis. Through the adopted forecasting technique, forecasts emerge which attempt to match the emergent real-world sales.

This description of a sales forecasting system portrays its suitability of being implemented using an adaptive agent architecture – one in which agents are situated in the real world; an internal model is used to make decisions; and the system is able to adapt over time.

4.5 Bayesian Networks as Internal Models

Implementing a sales forecasting system as an adaptive agent architecture provides an interesting approach to sales forecasting; and a practical use of an adaptive agent architecture. However, it might not affect the performance of the system. If one were to use regression analysis or econometric measures as the internal model, it should perform as such techniques generally do, provided that the data is reliable.

The use of Bayesian networks for sales forecasting seems a logical approach, yet it has not been widely explored. Our research revealed Hidden Markov Models (simple dynamic Bayesian networks) as perhaps the only application of Bayesian networks in sales forecasting. The ability of a Bayesian network to represent causal relationships is ideally suited to exogenous data forecasting. One can incorporate the causal effects of all influencing factors, and determine, probabilistically, the highest likelihood of sales when these factors are observed. In addition,

Bayesian networks can incorporate interrelationships among all variables, solving the problem of multi-collinearity experienced in regression analysis.

Bayesian networks can serve a similar purpose to econometric models, but in a clearer, more intuitive manner. The difficulties experienced in developing complex econometric models are alleviated, as the designer does not need to determine exact functions, but simply the causal relationships.

Bayesian network structures can be mined from data, or manually designed by a human being. Both approaches have their merits. By allowing a designer to determine the model, s/he incorporates his/her knowledge into the system. This introduces a qualitative aspect, found to be desirable for sales forecasting. Alternatively, mining the structure might be preferable in systems involving complex interdependencies beyond the intuition or knowledge of the designer. A combination of the two could be achieved by mining the structure but making intuitive changes to the model.

4.6 Conclusion

Sales forecasting should play a vital role in any company, and should form an integral part of sales and operations planning.

Forecasting can be qualitative: based on human opinion, or quantitative: numeric solutions derived from data. Quantitative approaches typically perform better when enough data is available, although a combination of both quantitative and qualitative techniques is preferable. When considering quantitative techniques, an approach should be based on the data available. Endogenous data (otherwise known as time-series) techniques are applicable where only sales history is available, while exogenous data (otherwise known as causal) techniques investigate causal relationships between sales and exogenous data such as promotions, prices, advertising and competitive actions.

This chapter discussed various approaches to sales forecasting. It briefly described qualitative techniques, in the form of *role playing*, *intentions*, *expert opinions, Delphi* and *conjoint analysis,* and provided a more in depth discussion of quantitative approaches. Fixed-model time-series

techniques include the *naïve method, averaging, exponential smoothing* and *adaptive smoothing*. Open-model time-series techniques include *Decomposition Analysis, Spectral Analysis, Fourier Analysis, Auto-regressive Moving Average (ARIMA)* and *the Box-Jenkins* method. Exogenous data techniques are of most relevance to this research. The most common approaches used are *regression analysis* and *econometric forecasting*.

The *situatedness* and adaptive nature of sales forecasting lends itself to be modelled using an adaptive agent architecture. Implementing a sales forecasting system in this manner would provide an alternative approach to sales forecasting and a further practical use of an adaptive agent architecture.

Bayesian networks are ideally suited to sales forecasting systems, as they are able to represent causal relationships in a clear, intuitive manner. Their ability to deal with multi-collinearity overcomes the problems associated with regression analysis, while they are simpler to design than complex econometric systems.

The suitability of both an adaptive agent architecture and a Bayesian network to sales forecasting systems, suggests that BaBe would be an ideal infrastructure with which to implement a sales forecasting system.

Chapter 5

Research Methodology

This research proposes both a practical aspect and a purely theoretical aspect. The practical aspect considers the effectiveness of a Bayesian network as a sales forecasting technique, as well as the performance of a learning adjustment component which we devised. This is investigated analytically through quantitative measures. The ability to model a sales forecasting system using and adaptive agent architecture, the theoretical aspect, is examined in terms of its adherence to the identified characteristics of adaptive agent architectures.

In this chapter we present our approach for conducting this research. We discuss a strategy for testing the effectiveness of the Bayesian network and the learning adjustment component. We describe our decisions for company selection and data collection; discuss the necessary actions required for data pre-processing; and detail a means for evaluating the test results. Finally, we explain how the system is evaluated as an adaptive agent architecture.

5.1 The Practical Aspect

5.1.1 Research Strategy

We develop a sales forecasting system, using the BaBe adaptive agent architecture, with a Bayesian network as the internal model. Rather than a generic system, we focus on an existing company to test the system with real-world data.

Testing is conducted according to the *holdout procedure* (Frank & Witten, 2005), whereby the system is trained with a portion of the data and tested with the remaining data. Morrison (2004: 1) identifies this procedure as the "most common approach to model validation." Various

approaches exist for *data splitting* – deciding which data should be used for testing, and conversely, which should be used for training. According to Morrison (2004), if the data is plentiful one can simply use a single portion of data for testing and the remainder for training. Frank et al. (2005: 149) explain that it is "common to hold out one-third of the data for testing and use the remaining two-thirds for training," but as a guideline, suggest that "each class in the full dataset should be represented in about the right proportion in the training and testing sets." Careful consideration should be paid to ensuring that the random sample contains a representation of each class in both testing and training data. This is known as stratification (Frank et al., 2005). Where the data is limited, or stratification cannot be achieved in a single holdout procedure, the *cross-validation* approach suggests performing multiple holdout procedures. In cross-validation, the data is split into a number of partitions, known as *folds*. Each fold, in turn, is used for testing, while the remainder is used for training (Frank et al., 2005). This ensures that the full dataset is covered in both training and testing. The most common approach is 10-fold cross-validation, where the data is split into ten folds, but Frank et al. (2005: 150) state that "5-fold or 20-fold cross-validation is likely to be almost as good." Due to the time required for training and testing the system, we adopt the 5-fold cross-validation approach.

Testing the Bayesian Network Forecasts

The data is divided into five folds, each with five months' data. Using the 5-fold cross-validation approach we iteratively test the system for each fold, training with the remaining data on each occasion. The sales forecasts provided by the Bayesian networks are then compared with the actual sales for their corresponding days.

To gauge the effectiveness of the Bayesian network, we compare the results with regression forecasts for the same test data, using the same training data for the regression model (again performing 5-fold cross-validation). Regression was previously identified as the most common form of exogenous data sales forecasting, and we determine whether a Bayesian network approach is more desirable for our data. The regression forecasting is conducted using Weka, an open source software package consisting of a collection of machine learning algorithms, developed at the University of Waikato, New Zealand (Frank et al., 2005).

Testing the Learning Adjustment Component

For each Bayesian network estimate, a corresponding adjustment is calculated using the learning adjustment component. To test its effectiveness, we compare the adjusted estimates with the actual sales and determine whether they are more accurate than the Bayesian network estimates. We anticipate that the learning adjustment component would be more effective where less training data is available. We therefore investigate the performance difference for increasing test data, by testing the system for three months', five months', eight months' and twelve months' data, while training with the remainder on each occasion.

5.1.2 Company Selection

Our main considerations for selecting a company on which to model the sales forecasting system are:

- Relevance for an exogenous data sales forecasting system; and
- Availability of data.

The company must be affected by exogenous factors in order to determine a causal model. In addition, companies are often unwilling to disclose data that they might deem to be confidential, such as sales records, promotions, or pricing strategies.

Meatpackers is a meat wholesale company. Their sales are largely affected by exogenous factors and they were willing to unconditionally cooperate with us for our research. In Chapter 6 we discuss Meatpackers in more detail and present the exogenous factors which influence sales.

5.1.3 Data Collection

The required data includes sales data, prices and all necessary exogenous data (identified in Chapter 6). Meatpackers has provided us with their available data. This includes two years worth of sales and price data. We are, however, limited in promotions and advertising data, as these were not formally recorded. Sporting events were captured from the Cape Argus (2006) website, which provides an archive of sporting events shown in South Africa, as well as sporting related newspaper articles. Weather data was provided by the South African weather bureau.

5.1.4 Data Pre-Processing

At the time of investigation, BaBe was unable to train a Bayesian network using continuous data. We therefore had to discretise the sales and pricing data. Discretisation is the process of converting continuous data of a potentially infinite range, into a fixed number of discrete intervals. Popular discretisation algorithms include Equal Width Discretisation (EWD), Equal Frequency Discretisation (EFD), Non-Disjoint Discretisation (NDD) and Proportional k-interval Discretisation (PKID) (Webb & Yang, 2002).

- EWD divides the number range between the minimum and maximum values into a fixed number of equal width intervals.
- EFD divides the values into *k* intervals, such that each interval contains approximately the same number of values.
- NDD overlaps intervals, always placing a value towards the centre of an interval.
- The PKID approach attempts to give equal weight to discretisation bias and variance by making the interval size equal the interval number. Webb et al. (2002: 7) explain that "discretisation bias and variance relate to interval size and interval number." If the interval size is large, the variance is low but the bias is high. Conversely, if the interval size is small, the bias is low, but the variance is high. According to Webb et al. (2002: 7), "previous experiments showed that PKID significantly reduced classification error for larger datasets."

In the light of Webb et al.'s (2002) findings, we select PKID as our discretisation method. For each product, at each branch, we calculate the number of intervals as $\sqrt{v_{max} - v_{min}}$, where v_{max} and v_{min} are the maximum and minimum values of the range. Discretisation is then essentially similar to EWD.

5.1.5 Evaluating the Results

We consider various performance measures identified previously as recognised means of evaluating the performance of sales forecasting systems. These include the MAE, MAPE, MSE and Theil's U-statistic. Based on the research presented in Chapter 4, the MAE, MAPE and MSE statistics were identified as desirable approaches for providing comparable statistics. Theil's U-statistic was presented as a common measure of accuracy for sales forecasting systems, and a

good indication of whether the forecasting technique can be considered acceptable for the given data.

To determine whether the Bayesian network estimates are significantly different from the adjusted estimates, or whether Weka's regression forecasts differ significantly from the Bayesian network estimates, we perform the two-sided t-test on the MAE, MAPE and MSE values. This a statistical test to compare two sample means without knowing the population standard deviation (Bradfield & Underhill, 2000). An f-test must first be conducted to determine whether the sample standard deviations are significantly different. The reader is referred to Bradfield et al. (2000: 218) for this calculation.

If the sample standard deviations are not significantly different, the t-test will take the form:

$$t = \frac{\overline{X}_1 - \overline{X}_2 - (\mu_1 - \mu_2)}{s\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \sim t_{n_1+n_2-2} \qquad (21)$$

Where $s = \sqrt{\frac{(n_1-1)s_1^2 + (n_2-1)s_2^2}{n_1+n_2-2}}$

If the sample standard deviations are found to be significantly different, the t-test can be approximated by

$$t^* = \frac{\overline{X}_1 - \overline{X}_2 - (\mu_1 - \mu_2)}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \sim t_{n^*} \qquad (22)$$

Where

$$n^* = \frac{\left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^2}{\frac{(s_1^2/n_1)^2}{n_1+1} + \frac{(s_2^2/n_2)^2}{n_2+1}} - 2$$

In the above calculations:

$\overline{X}_1$ and $\overline{X}_2$ are the sample means;

μ_1 and μ_2 are the population means;

s_1 and s_2 are the sample standard deviations; and

n_1 and n_2 are the sample sizes.

To investigate whether the means are significantly different, we use the null hypothesis that they are the same and determine whether this should be accepted or rejected. In this case $\mu_1 = \mu_2$, or equivalently, $\mu_1 - \mu_2 = 0$. The t-value for a given significance level depends on the sample sizes and the *degrees of freedom*. The null hypothesis is accepted at the given significance level if the t-value is within the required value of the t-distribution at the significance level. One would typically consider a significance level of at least 95%.

For this calculation the data is assumed to follow a normal distribution. This is acceptable since the central limit theorem states that "the sum of a large number of random variables always has a normal distribution (Bradfield et al., 2000: 183)." Bradfield et al. (2000: 183) state that "a sample size of 30 or more is large enough so that the distribution of the sample mean can be assumed to have a normal distribution." As all our tests have a sample size far greater than 30, we may use the t-test for a normal distribution.

We conduct these tests using Microsoft Excel's Data analysis tools.

5.2 The Theoretical Aspect

5.2.1 Research Strategy

In order to investigate the ability to model a sales forecasting system using an adaptive agent architecture, we examine our implemented system and describe how it fulfils the requirements of an adaptive agent architecture. We consider Brooks's (1991) characteristics of reactive planning systems, as well as additional characteristics identified in the Adaptive Agent Architectures chapter. We adopt the premise that if a system adheres to these requirements, then it can be considered to be an implementation of an adaptive agent architecture.

5.3 Conclusion

This research contains both a practical aspect a theoretical aspect. The research strategy to investigate the practical aspect, proposes implementing a system using the BaBe adaptive agent architecture, with a Bayesian network as the internal model. This system is tested to determine the effectiveness of a Bayesian network as a sales forecasting model, and the performance of the learning adjustment component.

The tests are conducted using the holdout procedure (Frank et al., 2005), with a 5-fold cross-validation data-splitting approach. In investigating the Bayesian network, the results are compared with regression forecasts conducted using Weka, a collection of machine learning algorithms developed by the University of Waikato, New Zealand. The same training and testing data is used for the regression forecasting as with the Bayesian network tests, and the 5-fold cross-validation holdout procedure is again employed. This determines whether the Bayesian network is preferable to a regression model, the common means of conducting exogenous data sales forecasts, for this data. In addition, the adjusted forecasts are compared with the Bayesian network forecasts to determine the impact of the learning adjustment component. Tests are conducted for increasing sample sizes, in order to investigate the effect of the quantity of training data and test data on the adjusted forecasts.

The system focuses on Meatpackers, a meat wholesale company whose sales are affected by exogenous factors. This company agreed to supply us with necessary data, including sales, prices, promotions and advertising data. Sports data was retrieved from the Cape Argus (2006) website, while weather data was provided by the South African weather bureau.

The results are evaluated in terms of recognised sales forecasting performance measures – the MAE, MAPE, MSE and Theil U-statistic. Where comparisons are made between regression forecasts and Bayesian network forecasts, or between Bayesian network forecasts and the adjusted forecasted values, calculated by the learning adjustment component, the t-statistic (Bradfield et al., 2003) is used. This is a recognised statistical measure to test whether there exists a significant difference between two sample means, without knowledge of the population standard deviation.

Finally, the system is examined in terms of Brooks's (1991) characteristics of reactive planning and other criteria identified in the Adaptive Agent Architectures chapter, to determine whether it does in fact implement an adaptive agent architecture.

Chapter 6

Designing the Meatpackers System

Meatpackers is a meat wholesale company that supplies meat to a retail store called *Meat City*, a division of *Fruit & Veg City* – the largest sellers of fruit and vegetables in South Africa (Fruit & Veg City, 2006). Their sales forecasting is conducted using an informal, qualitative approach. The manager uses his knowledge and experience to estimate sales for the following day or week. Meat sales are largely affected by environmental conditions, and the manager has a number of factors to consider. With enough exogenous data, the company would benefit from a reliable exogenous data sales forecasting system.

In this chapter we discuss the system scope in terms of the products and branches on which the system will focus. We then discuss the exogenous factors considered for the system, and present the Bayesian network to be used as the internal model. Finally we consider the requirements for the system as a whole, in terms of its behaviour and necessary components.

6.1 System Scope

The types of meat sold by Meatpackers are beef, lamb, chicken and processed meats (also known as manufactured meats). These are further divided into sub-categories. For example beef products include Texan steak, rump steak, sirloin, stew, Scotch fillet etc. Lamb includes lamb chops, lamb knuckles etc. Each sub-category, which we will now refer to as a *product*, might be affected by different environmental conditions. Consider steaks which might be eaten on a sunny Sunday afternoon, or beef stew which is popular on a cold, rainy night. In addition, Meat City has a number of branches situated in Cape Town. Each product's sales are dependent on the branch

location. For example, wealthier areas might buy more lamb chops, while poorer areas might favour the knuckles.

In this research we want to investigate a large portion of the company, while remaining realistic in terms of time and data constraints. Endeavouring to consider all branches and all products would be impractical and unnecessary. The research would be as valuable if conducted for a sub-section. However, considering only one branch or very few products would not give a good reflection of the company as a whole, and would negate some causal dependencies that we are attempting to investigate.

Therefore, in consultation with the Meatpackers manager, we identified three relevant branches and fifteen products on which to focus our system. These are displayed in **Table 2** and **Table 3**. They are the most popular branches and the highest selling products.

Table 2: Branches being investigated

Branch
Access Park
Table View
Tokai

Table 3: Products being investigated

Product
Beef: Espetada
Beef: Prime Rump
Beef: Scotch Fillet
Beef: Stew
Beef: Tenderised Steak
Beef: Texan Steak
Lamb: ¼ Mutton Pack
Lamb: ½ Mutton Pack
Lamb: Knuckles
Lamb: Loin Chops
Lamb: Neck
Manufactured: Beef Burgers
Manufactured: Beef Rashers
Manufactured: Grabouw Boerewors
Manufactured: Prima Boerewors

6.2 Exogenous Factors

Together with the Meatpackers manager, we identified appropriate exogenous factors affecting sales. We now consider these factors and how they might influence sales.

6.2.1 Price

Customer behaviour is influenced by price. When prices are considered low, sales are generally high. Conversely, when prices increase, sales generally decrease if all other factors remain constant.

6.2.2 Promotions

Promotions are a way of manipulating sales for a limited period. They typically reduce the price of a product, but might also exhibit the product in a store – perhaps allowing customers to taste the meat. Promotions generally increase sales of the product being promoted, and might additionally increase sales of other products by attracting more customers to the store.

We identified six of the most common promotions for the purposes of this project. These are displayed in **Table 4** below.

Table 4: Promotions being investigated

Product
Beef: Scotch Fillet
Beef: Texan Steak
Lamb: Mutton
Manufactured: Beef Burger
Manufactured: Beef Rasher
Manufactured: Boerewors

6.2.3 Advertising

Advertising encourages customers to a store by making them aware of the store or of possible promotions. Advertising is present in different forms of media, including television, radio, magazines and newspapers. While different forms of advertising might have different effects on sales, due to limited data we simply focus on whether an advertisement occurs.

6.2.4 Competitive Actions

We focus on competitors' promotions and advertisements. Practically, it does matter which competitor has advertisements or promotions. Larger companies are likely to attract more customers than smaller, less known ones. But due to limited data, we focus only on whether some competitor has a promotion or advertisement.

6.2.5 Date

Although we focus on exogenous data, we still need to consider aspects such as seasonality found in time-series approaches. By including the day of the week, the week of the month, and the month of the year, the Bayesian network can identify temporal patterns. Generally sales are high on a Friday, when customers prepare for the weekend. Often there are more sales in the first week of the month, when customers have more money. Or, it is likely that in the holiday period, around December, sales are greater than in any other month.

6.2.6 Weather

Meat sales are affected by the weather. Consider the example of a sunny Sunday afternoon, when many South African families would enjoy a braai (barbeque). This would affect the sales of products such as steak or chops. However, on a cold winter's evening, customers are more likely to remain indoors and eat stews or scotch fillet.

We therefore consider weather conditions in the form of wind, rain and heat (categories of temperature).

6.2.7 Sporting Events

A number of events might affect meat sales. Some might have a negative impact on sales, while others are positive. It is impossible to consider all relevant events, as even obscure or low profile events might have an impact. Often events occur without warning, such as a strike or power cut. These often cannot be anticipated and therefore cannot be used for forecasting.

We have chosen to focus on sporting events. Sport plays a large role in South Africa's culture, and a high profile game is often associated with braais, biltong and festivities. Consultation with the Meatpackers manager confirmed that meat sales are often affected by sports matches.

We focus on the main sports in South Africa – cricket, rugby and soccer. In addition, each sport may be national or international. We consider both forms.

6.3 The Bayesian Network

Using BaBe, we mined a Bayesian Network structure from the available data. We added a qualitative aspect by making minor changes to the mined structure, altering edges which seemed to incorrectly represent the real world situation. The resulting Bayesian network is shown in **Figure 14**. In the figure, promo is an abbreviation of promotion, MC is an abbreviation of Meat City, Opp is an abbreviation of opposition, and *wors* is a shortened form of boerewors.

BaBe's learning algorithm attempts to achieve an efficient structure for the Bayesian network. We therefore abandon the notion of a parent node being a cause and its children being the effects, as was discussed in section 3.7.

This Bayesian network encompasses the following causal relationships:

- Sales are directly affected by product, branch and price, as well as the temporal aspects – day of week, month and week of month.
- Promotions, advertisements, weather conditions and national sporting events are all dependent on the month. They additionally influence sales by propagating probabilities through the month node.
- *All* sporting events are dependent on the day of week and may additionally influence sales by propagating probabilities through the Day_of_week node.
- Price is affected by product and month.
- The Meatpackers promotions influence each other.

It should be noted that although BaBe employs a Bayesian *behaviour* network, we do not impose any constraints on the Bayesian network nodes. It may therefore be regarded either as a Bayesian network or as a simple Bayesian behaviour network.

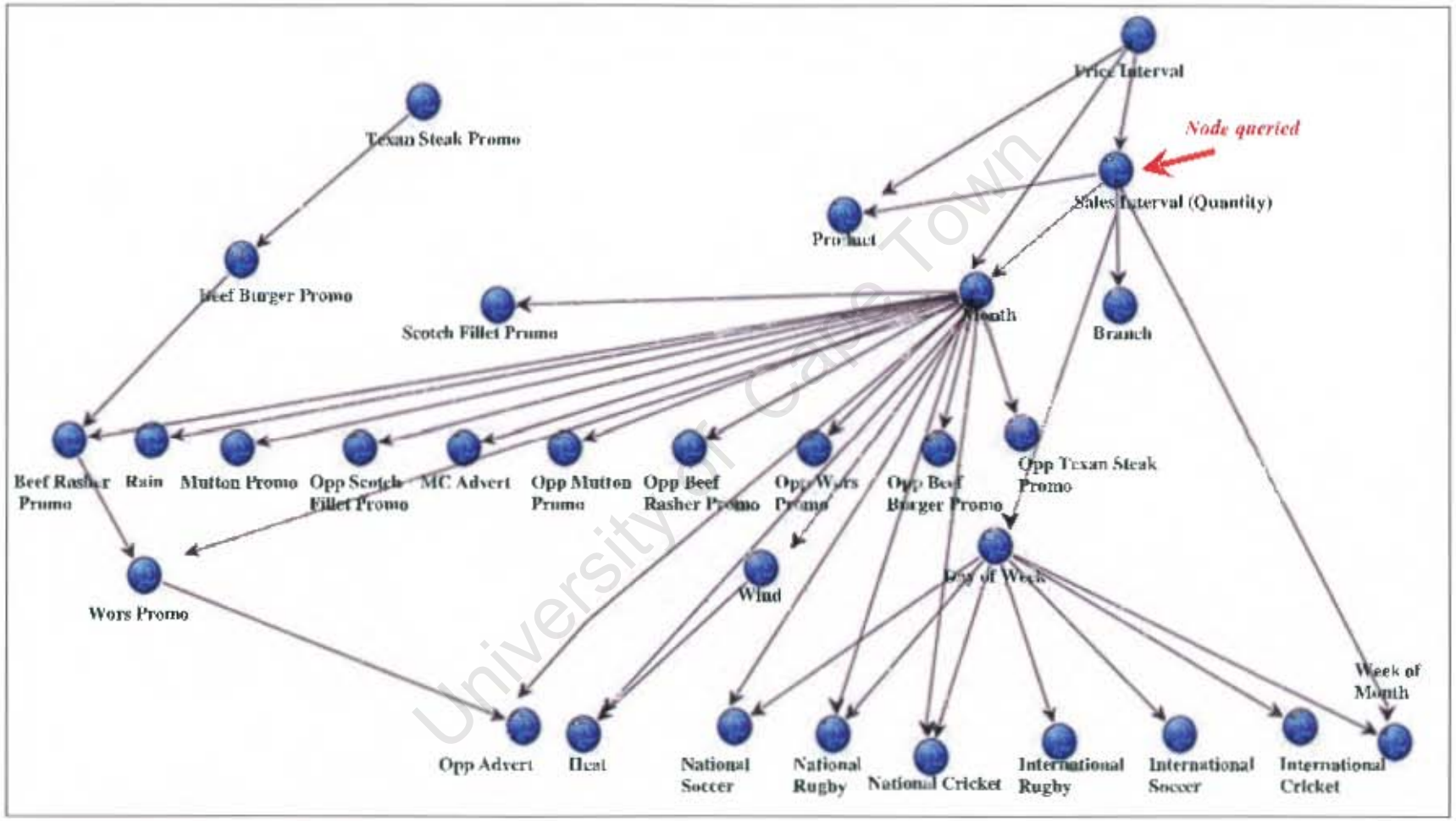

Figure 14: The Bayesian Network Structure

6.4 High Level System Design

The system as a whole requires a number of components, as depicted in **Figure 15**. We now focus on the high level design of the system and consider each of the required components.

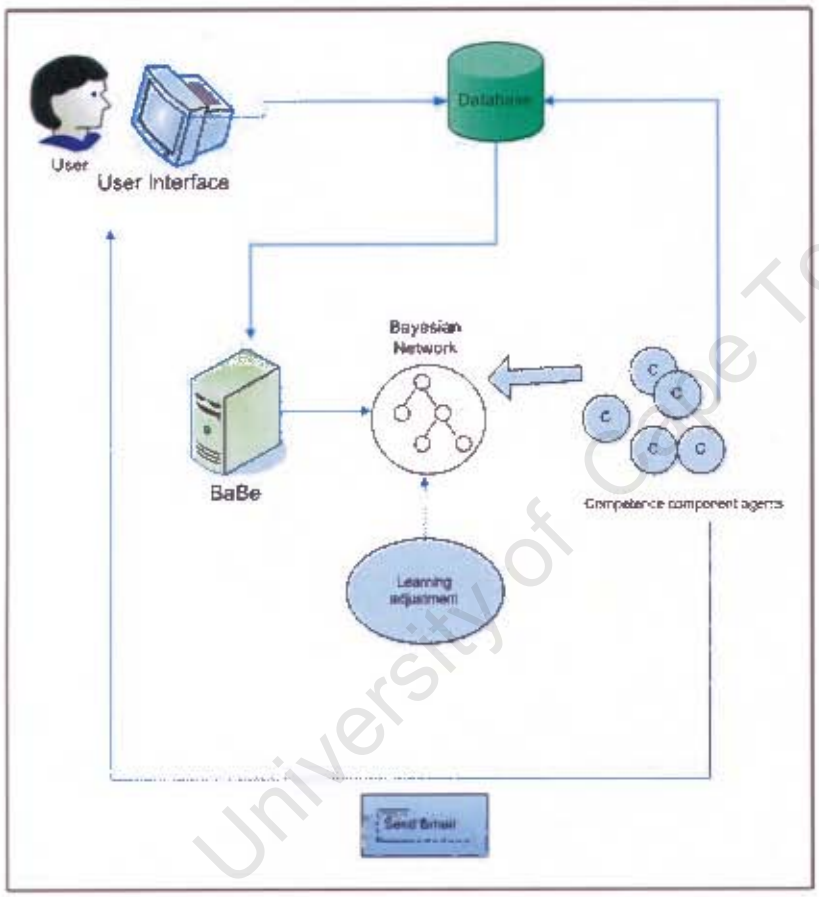

Figure 15: The high level design of the forecasting system

The user interface allows a user to interact with the system. This serves two important functions:

1. Capturing data into the database.
2. Setting the observed evidence for a given day.

The data entered by the user is recorded into a database for long-term storage. History data is used to train the Bayesian network, while Bayesian learning continues as long as data is captured.

BaBe trains the Bayesian network and conducts algorithms for Bayesian learning and inference, while competence agents manipulate the Bayesian network – setting evidence and querying the quantity. These agents are the high level workings of the BaBe system. They communicate with BaBe's reasoning agents and perform the steps necessary to obtain the estimated sales, and corresponding turnover, for the observed conditions.

A learning adjustment component performs feedback learning. It serves three purposes:

1. Bayesian learning requires a large amount of data in order to provide accurate results. The learning adjustment component is aimed to speed up the learning process, allowing for less data to be present before reliable results are seen.
2. BaBe's Bayesian algorithms currently do not handle continuous data. Quantities must be divided into discrete intervals. The Bayesian network would therefore supply an interval, rather than a specific number. While this is satisfactory for our purposes, we would like to estimate an exact number. Our approach is to consider the midpoint of the interval, however a learning adjustment could help direct the estimate towards a more accurate answer.
3. It serves as an investigation into an alternative approach of learning (explained in the following chapter.)

The Bayesian estimate is stored in the database for future learning (to be used by the learning adjustment component). The adjusted sales estimate is provided to the user through an email, rather than an interface. This adds to the autonomous nature of the system. Provided that the data is always up-to-date, and that the observed values for a given day have been supplied, the system could act autonomously - responding to a change in date and emailing the result.

6.5 Conclusion

Meatpackers would benefit from an exogenous data sales forecasting system, rather than relying only on human judgement. Meat sales are largely dependent on events and exogenous factors. Such factors include prices, promotions, advertisements, weather and competitive actions. Events such as sports matches also influence sales. In addition, the day of week, week of month and month of year are valuable in identifying seasonality and temporal patterns.

Together with the Meatpackers manager, we identified branches and products on which to focus this research. These are the most popular branches and the highest selling products.

We mined a Bayesian network based on the identified factors, and the branches and products being examined. This serves as the internal model for our system. The system as a whole requires an interface, linked to a database for data capturing. BaBe would train the Bayesian network, while competence agents manipulate it. A learning adjustment component allows for faster learning and adjusts the discretised forecasts produced by the Bayesian network towards a closer approximation of actual sales.

Chapter 7

Implementation

In the previous chapter we discussed the requirements for the system as a whole. A user interface is required to allow an end user to capture environment data, to be recorded into a database for processing by the BaBe system. BaBe trains a Bayesian network that can be queried by competence agents. These agents interact with each other and with BaBe's reasoning agents to forecast the sales of the meat products for a given date and branch. A learning adjustment component performs a feedback learning algorithm to increase learning speed and to adjust the discretised estimates towards closer approximations. Finally, a mailing agent completes the behaviour by autonomously sending the results to the MeatPackers company.

We now focus on the implementation of each aspect of the system. We briefly discuss the user interface and database, but pay more attention to the competence agents and learning adjustment, which form a large portion of this research. BaBe is used as a third party tool whose workings were explained in Chapter 4. Its learning algorithms and agent implementations are beyond the scope of this research.

7.1 The User Interface and Database

The user interface allows an end user to capture environment data. These are the factors that influence the system. In this investigation, it refers to those factors which affect the meat sales.

The interface, as shown in **Figure 16**, is a compilation of Java applets that can be accessed on the Internet. An applet exists for each of the factors that influence meat sales. The hyperlinks on the left allow navigation to each applet.

The applets are linked to a MySQL database for storing the captured data.

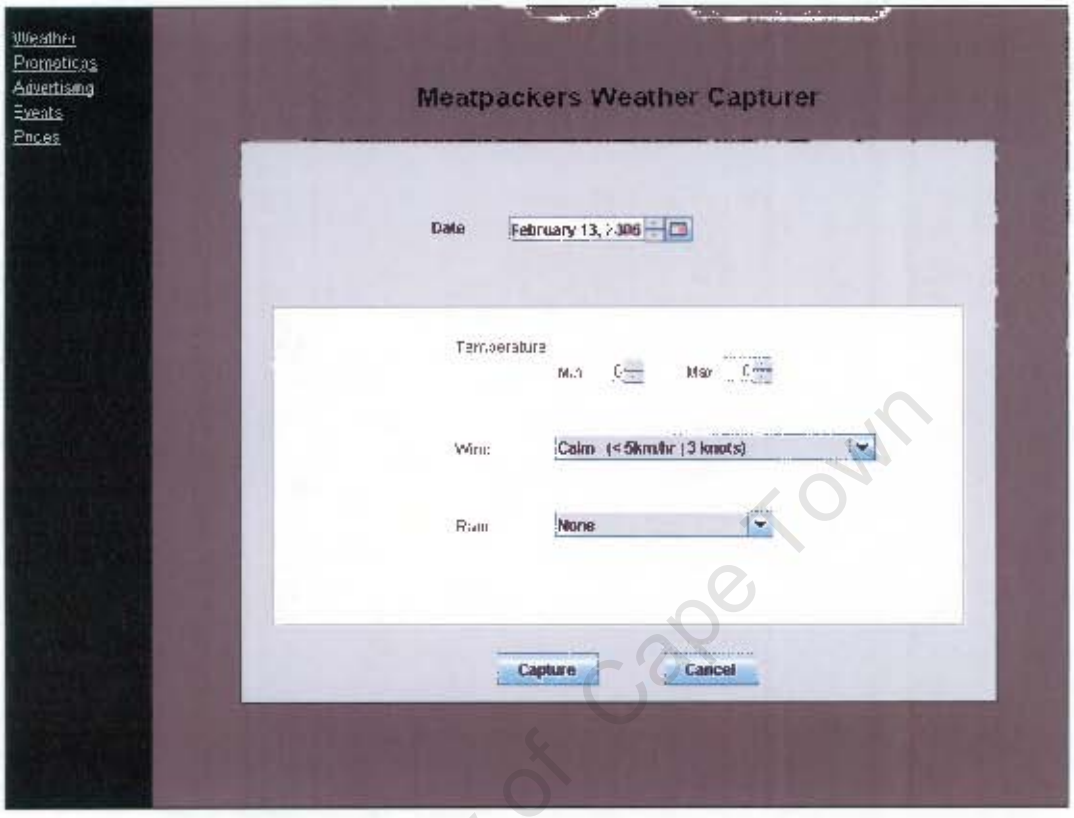

Figure 16: The Data Capturer interface

The database stores environment data, captured by the user interface, as well as the estimated sales determined by the agents. A presentation of the schema is provided in Appendix A.

7.2 The Competence Agents

The competence agents were implemented in Java, using the BaBe agent framework. Agents are either simple or complex. Simple agents each perform a single function. Complex agents may perform multiple functions, but they, themselves, can be broken down into simple agents. Complex agents could be thought of as agencies, as described in 0.

Through interactions among these agents, a complex competence emerges. There is no central control to coordinate behaviour. Each agent *listens* for some event, responds accordingly and

emits a result. When an agent listens for an event, its execution is triggered by the event. This event might be external or the emitted result of another agent.

7.2.1 How the Agents Work

Figure 17 illustrates the agents involved and the flows between them. The italicised text indicates the result emitted by the agent. This is passed to its listening agents.

The *Date Trigger* agent listens for a date change event. When the date changes, it responds and emits the new date. The *BranchProduct Trigger* agent listens for the result of the *Date Trigger* agent. Therefore, after the *Date Trigger* agent emits its result, the *BranchProduct Trigger* agent responds. This complex agent consists of simple agents that retrieve all branches and products from the database. An *Array Multiplexer* agent joins the results of these database queries to produce an array of {Branch, Product, Date} combinations. This enables sales to be estimated for all products, at each branch, on the given date.

The *Iterator* agent, belonging to the *BranchProduct Trigger* agent, iterates through the array of {Branch, Product, Date} combinations and emits a single combination at a time. Each such combination can therefore be manipulated independently by the various agents. This implementation alleviates the need for a loop, which would require some globally coordinating control.

The *Retrieve Values* agent responds to the *BranchProduct Trigger* agent as it receives the {Branch, Product, Date} combination. For the given branch, product and date, it retrieves all observed values from the database, using a SQL query. These include the weather, promotions, advertising and sporting events that affect this {Branch, Product, Date} combination; as well as the price of the product (in terms of a discretised interval) for the given branch and date. It then emits the {Branch, Product, Date} combination together with its observed values.

Two agents listen for the result of the *Retrieve Values* agent. The *Set Evidence* agent requires the observed values to set the evidence of the Bayesian network, while the *Calculate Turnover* agent requires the observed price (as well as the estimated sales) to calculate the turnover.

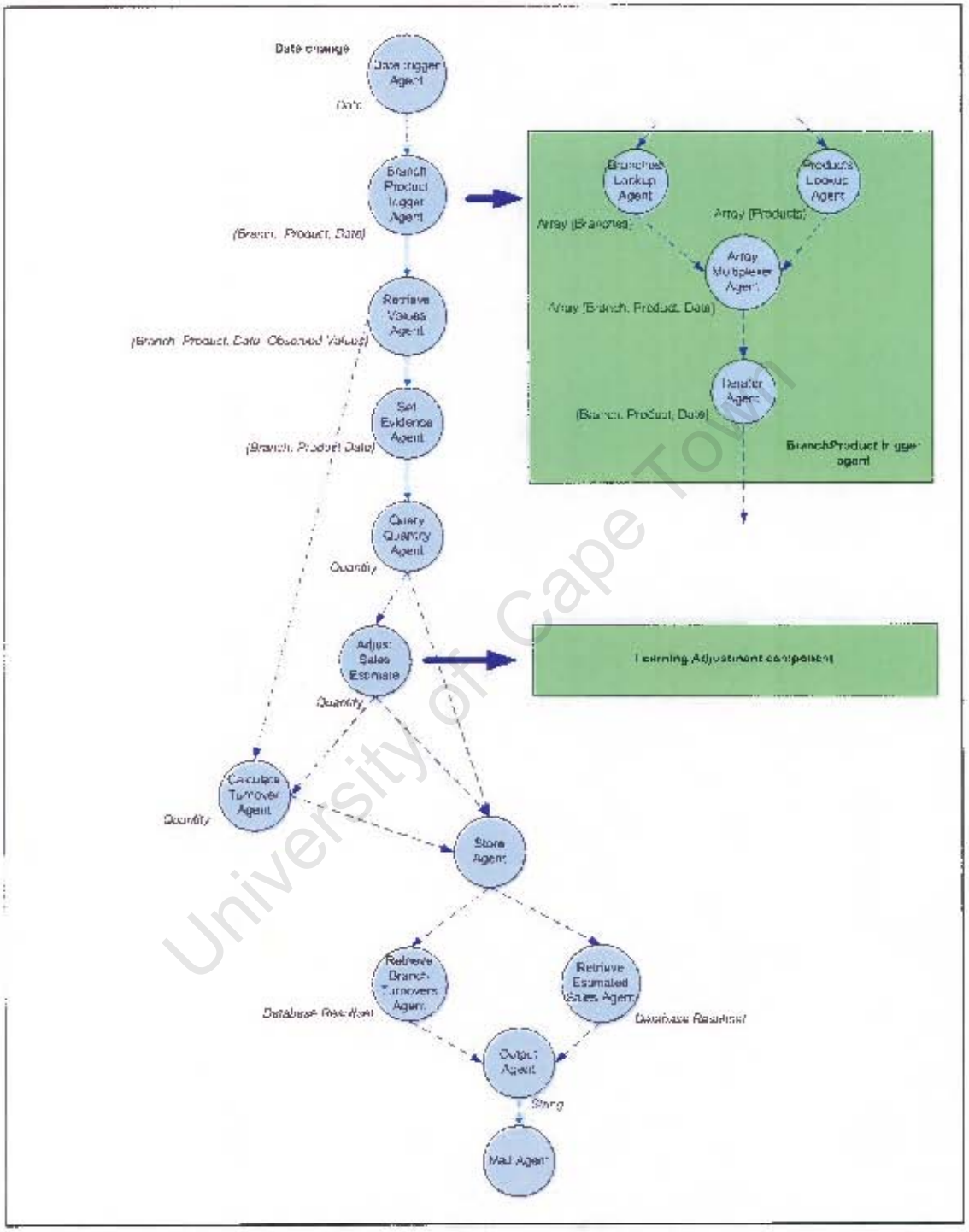

Figure 17: The flows between competence agents

The *Set Evidence* agent emits the {Branch, Product, Date} combination for the *Query Quantity* agent. This agent queries the Sales_IntervalID node of the Bayesian network for the given branch, product and date. This involves communicating with BaBe's reasoning agents, which perform Bayesian inference using the "Explaining Away" approach – a combination of top-down and bottom-up inference. The *Query Quantity* agent retrieves the sales interval with highest probability and determines a value in the middle of the interval range. The result is the estimated sales determined by the Bayesian network, which we refer to as the Bayesian estimated sales. The agent then emits a Java object which contains the {Branch, Product, Date} combination together with the Bayesian estimated sales.

The *Adjust Sales Estimate* agent performs the learning adjustment algorithm, and emits an adjusted estimated sales. This complex agent has been omitted from the diagram to avoid clutter. It will be discussed in more detail later in the chapter.

The *Calculate Turnover* agent, listening for the estimated sales, now has all its required information. It can therefore calculate the estimated turnover of the given {Branch, Product, Date} combination by multiplying the price of the product by its estimated sales.

The *Store* agent listens for the outputs of the *Query Quantity* agent, *Adjust Sales Estimate* agent and *Calculate Turnover* agent. Depending on which agent has completed, the *Store* agent will store the result in an appropriate table of the database.

After the sales and turnover have been estimated for all {Branch, Product, Date} combinations, the system sends an email to MeatPackers, informing them of its findings. This process requires retrieving the stored data from the database and preparing it in a readable format to be emailed.

The *BranchProduct Trigger* agent emits a *Completion Emitter* (discussed further in section 7.2.3) when all {Branch, Product, Date} combinations have been emitted. The *Retrieve Branch Turnovers* agent and *Retrieve Estimated Sales* agent listen for this event and proceed once the *Store* agent has completed (and emitted its own Completion Emitter). These agents are responsible for retrieving the estimated sales and turnovers from the database, for the given date. Each performs a SQL query and emits a JDBC ResultSet.

The *Output* agent listens for the completion of both the *Retrieve Estimated Sales* agent and *Retrieve Branch Turnovers* agent. It stores this output in a neatly formatted Java String, before the *Mail* agent emails this String.

7.2.2 Defining Agent Interactions

Each agent must know when to execute. As there is no central control, this is achieved through sending signals from one agent to another. By defining agent interactions, agents know when it is appropriate to respond to a signal.

Figure 18 shows how agent interactions are defined in the program. Adding the listener of one agent, A, to an emitter of another agent B, instructs A to listen for the output of B. For example, the *BranchProduct Trigger* agent (bpTrigger) has its listener added to the *Date Trigger* agent's emitter, with the line:
dateTrigger.getTriggerEmitter().addListener (bpTrigger.getETrigger Listener());.
Therefore, after the *Date Trigger* agent's "TriggerEmitter" has executed, the *BranchProductDate Trigger's* "ETriggerListener" will respond.

```
dateTrigger.getTriggerEmitter().addListener(bpTrigger.getETriggerListener());
bpTrigger.getEDataEmitter().addListener(valuesAgent.getListener());
bpTrigger.getCompletionEmitter().addListener(adjustmentAgent.getListener());
bpTrigger.getCompletionEmitter().addListener(salesAgent.getListener());
valuesAgent.getEmitter().addListener(evidenceAgent.getObservedValuesListener());
valuesAgent.getEmitter().addListener(turnoverAgent.getPriceListener());
evidenceAgent.getEmitter().addListener(queryAgent.getListener());
queryAgent.getEmitter().addListener(adjustmentAgent.getListener());
adjustmentAgent.getQuantityEmitter().addListener(turnoverAgent.getQuantityListener());
adjustmentAgent.getQuantityEmitter().addListener(stornAgent.getAdjustedQuantityListener());
turnoverAgent.getEmitter().addListener(storeAgent.getTurnoverListener());
storeAgent.getEmitter().addListener(branchTurnoversAgent.getCompletionListener());
storeAgent.getEmitter().addListener(salesAgent.getCompletionListener());
branchTurnoversAgent.getEmitter().addListener(outputAgent.getTurnoversListener());
salesAgent.getEmitter().addListener(outputAgent.getSalesListener());
outputAgent.getEmitter().addListener(mailAgent.getListener());
```

Figure 18: Progam code for defining agent interactions

7.2.3 Synchronisation

All agents run concurrently in separate threads. The BaBe framework's agent interaction, by design, is asynchronous. An agent will emit a signal after completing its execution, irrespective of the status of other agents. Therefore an agent may emit a signal even when a listening agent is processing previous data. This has the benefit of producing a higher throughput, but could result in inconsistent data.

Consider the situation where the *Query Quantity* agent responds to the *Set Evidence* agent's signal. Assume the *Query Quantity* agent is querying the quantity node (the Sales_IntervalID) for BranchID = 0, ProductID = 1, and the *Set Evidence* agent has successfully set the evidence for BranchID = 0, ProductID = 2. The *Set Evidence* agent emits its signal, to which the *Query Quantity* listener responds. It is possible that the *Query Quantity* agent will query the Bayesian network for BranchID = 0, ProductID = 1, when the evidence has been set for BranchID = 0, ProductID = 2. This will reflect an incorrect result.

Another area of concern involves the *Calculate Turnover* agent. This agent multiplies the price of a product, for a given branch, by the estimated sales of that product. With asynchronous behaviour, it is possible that the price emitted by the *Retrieve Values* agent does not refer to the same {Branch, Product, Date} combination as the quantity provided by the *Adjust Sales Estimate* agent.

In order to avoid these inconsistencies, it was necessary to ensure synchronous communication. This is achieved through the use of completion emitters and completion listeners. Each agent has a completion emitter, which is emitted upon completing its execution. In addition, each agent has a completion listener. This listener listens for the completion of its communicating agents. Once an agent has emitted its completion emitter, it indicates to listening agents that it is ready for more processing. A listening agent may then proceed with execution and emit its result.

Therefore the *Set Evidence* agent has a completion listener which listens for the completion of the *Query Quantity* agent. When the *Query Quantity* agent is ready, it emits its completion emitter. The *Set Evidence* completion listener then allows the agent to emit its result for the *Query Quantity* agent.

Figure 19 demonstrates, graphically, the completion dependencies of the agents. A solid arrow from one agent, *A*, to another agent, *B*, indicates that *B* must wait for the completion of *A*. In such as situation, *A* will emit a completion emitter for which *B* must listen. One can see from the diagram that the *Set Evidence* agent waits for the completion of the *Query Quantity* agent before setting the new evidence and emitting the result.

The *Retrieve Branch Turnovers* and *Retrieve Estimated Sales* agents wait for the completion of the *Date Trigger* agent. They execute only once the sales and turnover have been estimated for all {Branch, Product, Date} combinations.

The other completion dependencies ensure that the price supplied by the *Retrieve Values* agent corresponds with the same {Branch, Product, Date} combination as the sales estimated by the *Query Quantity* agent, and that these are stored correctly in the database.

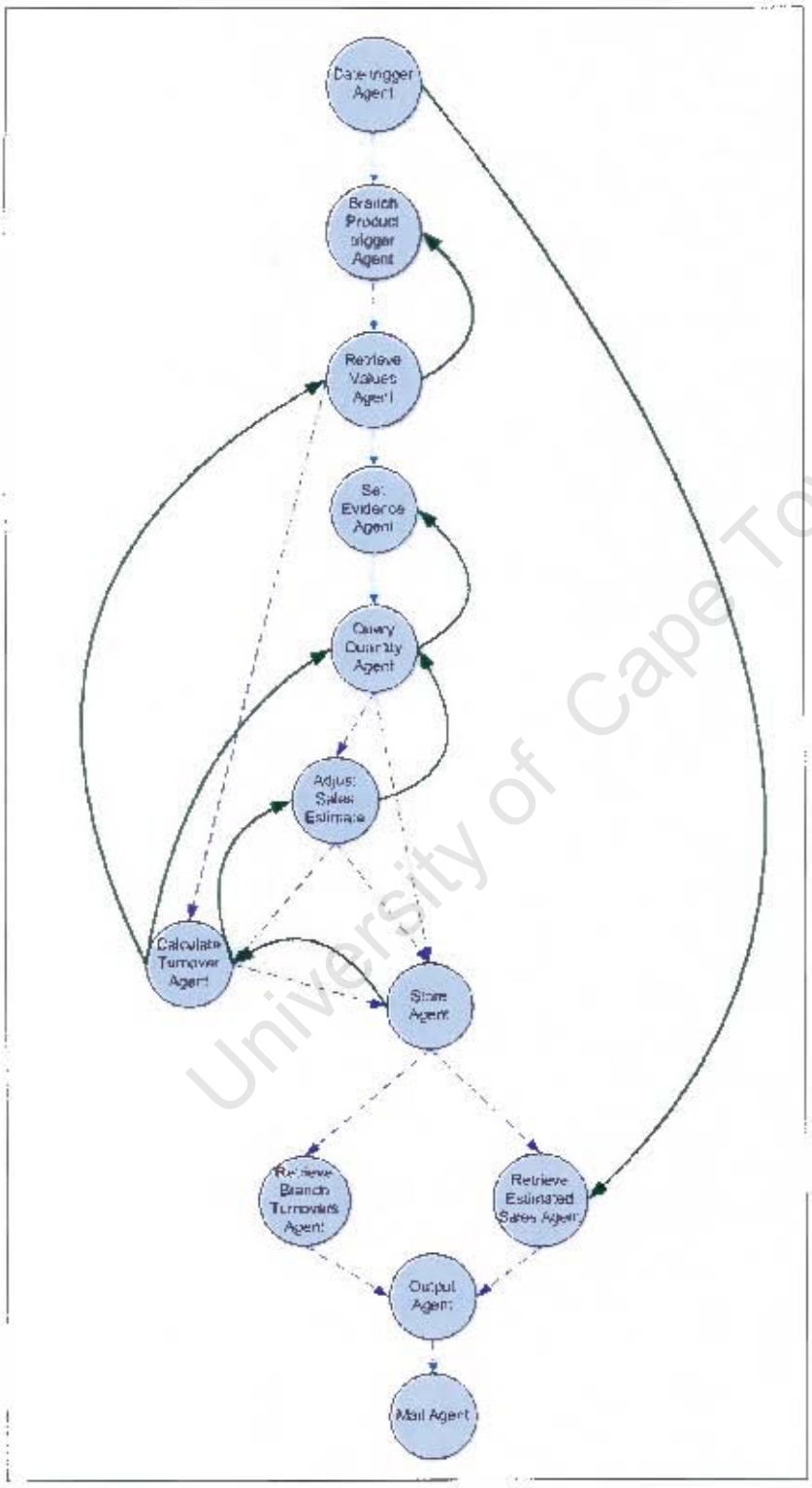

Figure 19: Agents using completion emitters and completion listeners for synchronisation

Figure 20 shows the program code for defining completion dependencies. The line
`valuesAgent.getCompletionEmitter().addListener(bpTrigger.getCompletionListener());`
instructs the *BranchProduct Trigger* agent to listen for the completion of the *RetrieveValues* agent, with its completion listener responding to the *RetrieveValues* completion emitter.

```
valuesAgent.getCompletionEmitter().addListener(bpTrigger.getCompletionListener());
queryAgent.getCompletionEmitter().addListener(evidenceAgent.getCompletionListener());
turnoverAgent.getCompletionEmitter().addListener(valuesAgent.getCompletionListener());
turnoverAgent.getCompletionEmitter().addListener(queryAgent.getCompletionListener());
turnoverAgent.getCompletionEmitter().addListener(adjustmentAgent.getCompletionListener());
storeAgent.getCompletionEmitter().addListener(turnoverAgent.getCompletionListener());
adjustmentAgent.getCompletionEmitter().addListener(queryAgent.getCompletionListener());
```

Figure 20: Progam code for defining completion dependencies

7.3 The Learning Adjustment Component

The learning adjustment component calculates an adjustment by which to alter the Bayesian estimate retrieved by the competence agents. This adjustment is calculated using previous Bayesian estimates supplied by the system, thereby introducing feedback learning. Its aim is to produce more accurate results than would be achieved through Bayesian learning alone, by increasing the learning speed and improving on the discretised results.

7.3.1 How the Learning Algorithm Works

The algorithm extends the mean error calculation, used in time-series forecasting, to determine a mean error over exogenous factors. It calculates an average deviation of the Bayesian estimated sales from the actual sales, taking into account each of the influencing factors. We discussed, in Chapter 4, how the mean error calculation is not always desirable as the positive and negative signs cancel each other out. However, in this case we are trying to determine an adjustment, rather than measure performance. Therefore the cancelling out of positives and negatives is desirable to give an average amount by which to alter the Bayesian estimate. For each {Branch, Product, Date} combination, an adjustment is calculated.

Let the error, the difference between the Bayesian estimated sales and actual sales, be E, and let the observed values pertaining to the given {Branch, Product, Date} combination be ε.

Where ε contains

{windID, rainID, heat,
scotch fillet promo, Texan steak promo, beef burger promo, beef rasher promo,
wors promo, mutton pack promo,
opp scotch fillet promo, opp Texan steak promo, opp beef burger promo, opp
beef rasher promo, opp wors promo, opp mutton pack promo,
day_of_week, week_of_month, month_of_year,
meatCity advertisement, opp advertisement,
national cricket match, international cricket mach, national rug by match,
international rugby match, national soccer match, international soccer match}

For each variable x in ε, the total error, **SUM(E)**, is retrieved from the database for all values matching x. This is calculated as SUM(Actual sales) – SUM(Bayesian estimated sales).

This produces a result $\sum_x$ **SUM**(E_x)
where E_x refers to the error for the record matching variable x.

In addition, for each variable x in ε, the number of occurrences, **COUNT**(x), is retrieved for each value in the database matching x.

This produces a result $\sum_x$ **COUNT**(x).

The final adjustment is calculated by dividing the overall error by the number of occurrences of all relevant exogenous factors:

$$\sum_x \textbf{SUM}(E_x) / \sum_x \textbf{COUNT}(x). \quad (23)$$

This is added to the estimate provided by the competence agents, producing an adjusted sales estimate.

An illustration of the algorithm is provided in **Figure 21**.

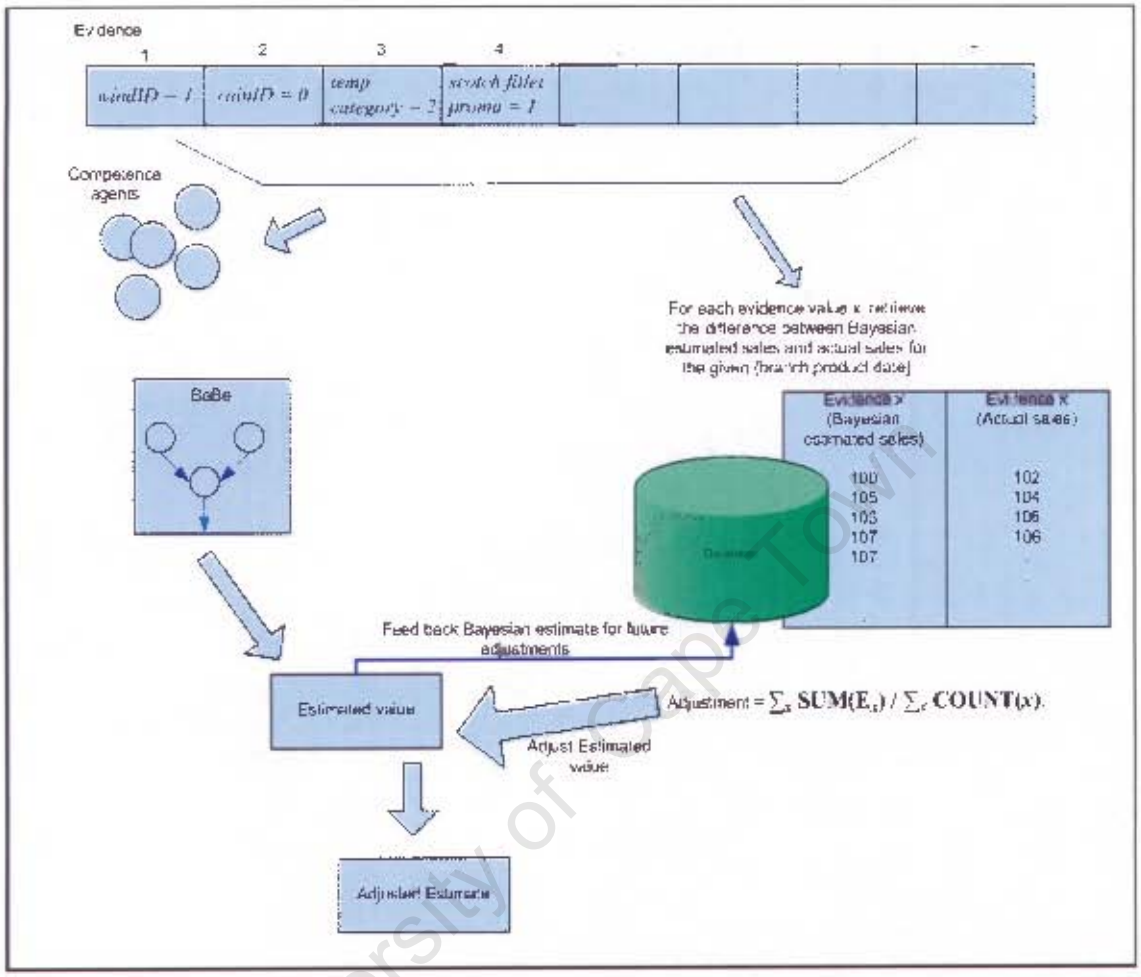

Figure 21: The learning adjustment algorithm

7.3.2 An Example

For clarity, consider the following fictitious example.

Given:

BranchID = 1, ProductID = 3, date = 2 April 2006

The following factors have been observed:

ε = *{ windID = 1, rainID = 0, temperature category = 3,*
scotch fillet promo = false, Texan steak promo = false,
beef burger promo = true, beef rasher promo = true, wors promo = false,
mutton pack promo = false,

opp scotch fillet promo = false, opp Texan steak promo = false, opp beef burger promo = false, opp beef rasher promo = true, opp wors promo = fasle, opp mutton pack promo = false, day_of_week = 1, week_of_month = 1, month = 4, meatCity advertisement = true, opp advertisement = false, national cricket match = false, international cricket match = true, national rugby match = false, international rugby match = false, national soccer match= false, international soccer match =false }

For branchID = 1, productID = 3

SUM ($E_{windID = 1}$) = 30, COUNT(*windID = 1*) = 17
SUM ($E_{rainID = 0}$) = 10, COUNT(*rainID = 0*) = 23
SUM ($E_{temperature\ category\ = 3}$) = 12, COUNT(*temperature category = 3*) = 48
SUM ($E_{scotch\ fillet\ promo\ =\ false}$) = 10, COUNT(*scotch fillet promo = false*) = 5
...
...
...
SUM ($E_{international\ soccer\ match\ =\ false}$) = 15, COUNT(*international soccer match = false*) = 25

$\sum_x$ SUM(E_x) = 1473, $\sum_x$ COUNT(x) = 295 records.
Adjustment = 4.99

Therefore add 4.99 to the result produced by the competence agent for the combination {BranchID = 1, ProductID = 3, date = 2 April 2006}.

7.3.3 Implementing the Learning Adjustment Component

The learning adjustment component is conducted by the *Adjust Sales Estimate* agent. It receives the estimated sales determined by the Bayesian network, and emits an adjusted estimated sales value. This complex agent uses simple agents, as depicted in **Figure 22**.

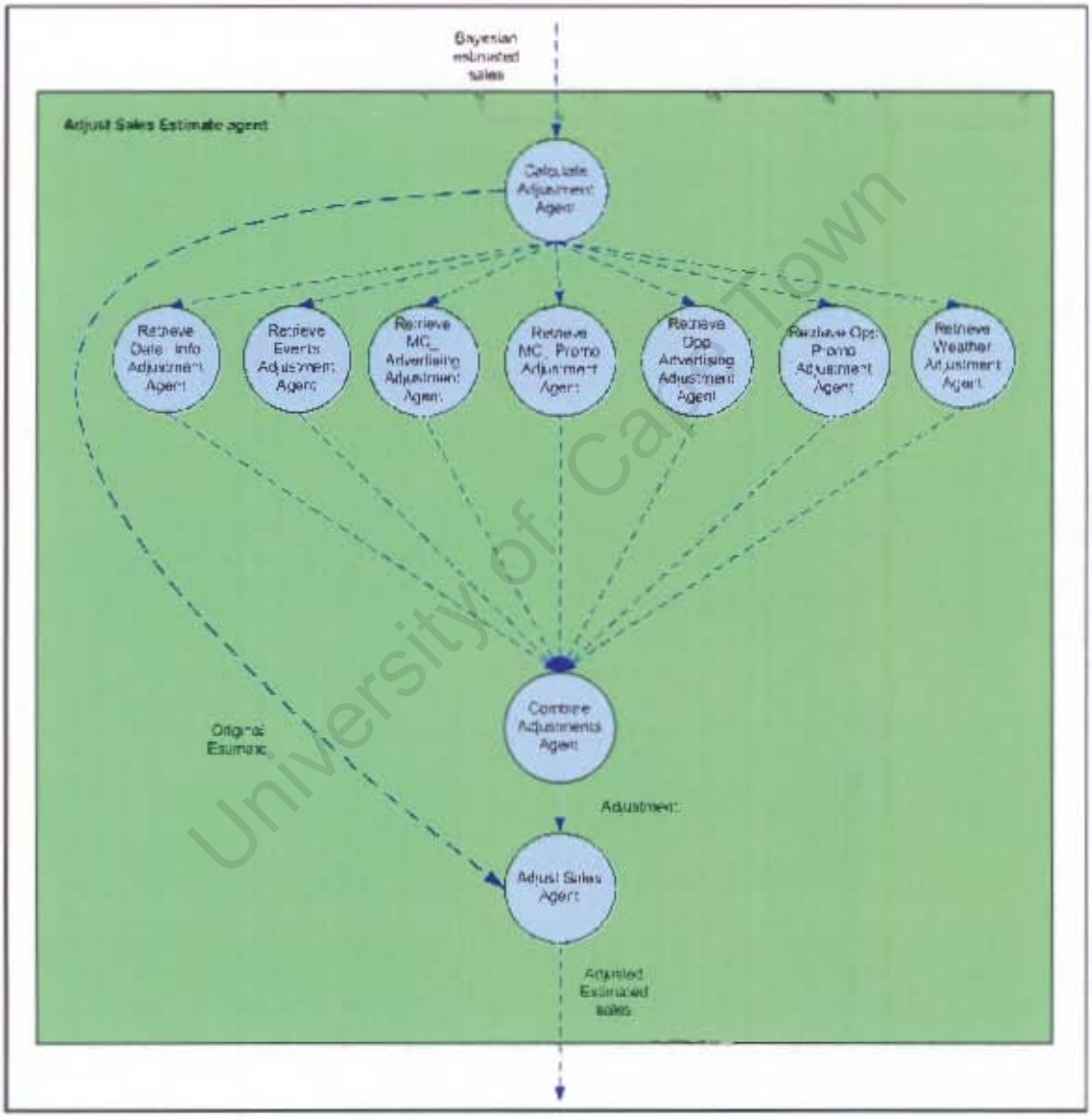

Figure 22: Implementing the learning adjustment using agents

The process of calculating an overall adjustment is divided into a number of smaller processes, each concerned with retrieving an adjustment (in the form of a $\mathbf{SUM}(E_x)$ value and a $\mathbf{COUNT}(x)$

value) based on a different environmental factor. Each process is implemented using a single, simple agent.

1 The *Retrieve Date_Info Adjustment Agent* is concerned with retrieving an adjustment based on the day of the week, week of the month, and month of the year.

2 The *Retrieve Events Adjustment* agent retrieves an adjustment for the sporting events that occur on the given date. This combines the adjustments for all sporting events.

3 The *Retrieve MC_Advertising Adjustment* agent retrieves an adjustment based on whether there is a MeatCity advertisement for the given {Branch, Product, Date} combination. Similarly the *Retrieve Opp_Advertising Adjustment* agent retrieves an adjustment based on whether there is an opposition advertisement on the given date.

4 The *Retrieve MC_Promo Adjustment* agent and *Retrieve Opp_Promo Adjustment* agent each determines an adjustment for the observed promotions.

5 The *Retrieve Weather Adjustment* agent retrieves an adjustment according to the heat, wind and rain, for the given {Branch, Product, Date} combination.

Each of these agents emits a result which includes the {Branch, Product, Date} combination being investigated, as well as values for $\mathbf{SUM}(E_x)$ and $\mathbf{COUNT}(x)$. The listening agent – the *Combine Adjustments* agent – combines each adjustment into a single overall adjustment for the given {Branch, Product, Date} combination, by calculating $\sum_x \mathbf{SUM}(E_x) / \sum_x \mathbf{COUNT}(x)$. It then emits this result together with the {Branch, Product, Date} combination.

The *Adjust Sales* agent listens for the result of the *Combine Adjustments* agent and adds this adjustment value to the Bayesian estimated sales. This value is stored in the database as the adjusted estimated sales.

7.4 Conclusion

The system required the implementation of a number of components. A compilation of Java applets provide an interface with which to capture appropriate environment data over the Internet. These are linked to a MySQL database for storing the data. BaBe trains a Bayesian network from the collection of data, while competence agents are responsible for the behaviour of the system. These are simple agents or complex agents (composed of simple agents). They perform the steps necessary to determine the estimated sales and turnover – retrieving the observed data, setting the evidence of the Bayesian network, and querying the quantity (the Sales_IntervalID node) of the Bayesian network.

The agents communicate by listening for, and emitting, events. When an agent listens for an event, its execution is triggered by the event. This event might be external or the emitted result of another agent. Their flows of communication and information are based on the interactions defined in the program.

It was necessary to ensure synchronous communication among agents, in order to avoid inconsistent data. This is achieved through completion emitters and completion listeners. Completion emitters indicate when an agent has completed its processing. Communicating agents would listen for this event with completion listeners and proceed with their execution once retrieving the event.

Additional agents are used to implement the learning adjustment component. Together, these form a complex agent, known as the *Adjust SalesEstimate* agent. The learning algorithm is based on the mean error time-series performance measurement, but is extended to incorporate exogenous data. This approach introduces feedback learning, commonly found in adaptive agent architectures. It produces an adjustment reflecting the average deviation of the Bayesian estimates for the given conditions. Agents adjust the sales, estimated by the Bayesian network, using the calculated adjustment.

Chapter 8

Results

In this chapter we focus on the testing of the quantitative aspect of this research. We tested the accuracy of the system to determine the effectiveness of the Bayesian network as an internal model for a sales forecasting system. We present the results of these tests and compare them with a linear regression forecast, conducted on the same data, using Weka. We additionally investigate the effectiveness of our learning adjustment component, and pay particular attention to its performance with varying quantities of training data.

Testing was conducted using the holdout procedure, with a 5-fold cross-validation data splitting approach, as described in Chapter 5. This involved dividing the data into five folds and iteratively testing the system for each fold, while training the system with the remaining data on each iteration.

The results are discussed in terms of recognised statistical measures, used as sales forecasting performance measures. The most relevant measures, extracted from our research in Chapter 4, are the Mean Absolute Error (MAE), Mean Squared Error (MSE), and Mean Absolute Percentage Error (MAPE). Theil's U-statistic additionally provides a good indication of whether a forecast is acceptable for the given data.

T-tests, for comparing sample means, were conducted when comparing the various mean errors and mean Theil U-statistics. Their results are discussed where appropriate.

We now present the results, in terms of:

- The Bayesian network's accuracy in forecasting the correct discretised sales interval;
- A comparison of the accuracy of the Bayesian network estimates, adjusted estimates (the

estimates calculated by adjusting the Bayesian estimate using the learning adjustment component) and regression estimates; and

- A more in depth analysis of the performance of the learning adjustment component.

8.1 Forecasting the Correct Interval

As we trained the system with discrete data, we investigated the consistency with which the Bayesian network was able to predict the correct sales interval. The results are presented, for each test fold, in **Table 5**.

Table 5: Forecasting the correct interval

Test Fold	**Correct interval prediction percentage (%)**
1 (May 2004 – Sep 2004)	64.68
2 (Oct 2004 – Feb 2005)	65.39
3 (Mar 2005 – Jul 2005)	69.16
4 (Aug 2005 – Dec 2005)	67.09
5 (Jan 2006 – May 2006)	77.22
Overall	**68.65**

For each fold, the percentage correct interval prediction is over 60%, indicating a majority of the cases being forecasted correctly. The *overall* correct interval prediction percentage is a percentage taken over all test data. This has a value of 68.65%. T-tests revealed similar percentage correct interval prediction for folds 1 and 2, with all others being significantly different from each other at the 99% significance level.

Fold 5 predicted the correct interval on 77.22% of values. This is likely due to more consistent sales behaviour in the given months. An examination into standard deviations and standard errors of the data revealed the following results, shown in **Table 6** and **Table 7**.

Table 6: Standard deviation of data folds

Test Fold	Standard Deviation
1 (May 2004 – Sep 2004)	27.20788205
2 (Oct 2004 – Feb 2005)	24.79620973
3 (Mar 2005 – Jul 2005)	26.76361887
4 (Aug 2005 – Dec 2005)	18.95157
5 (Jan 2006 – May 2006)	12.1435935
Overall	**21.77903825**

Table 7: Standard error of data folds

Test Fold	Standard Error
1 (May 2004 – Sep 2004)	0.535966354
2 (Oct 2004 – Feb 2005)	0.38858087
3 (Mar 2005 – Jul 2005)	0.467598264
4 (Aug 2005 – Dec 2005)	0.213941
5 (Jan 2006 – May 2006)	0.1932919
Overall	**0.147781063**

Standard deviation is a measure of the spread of the data, while standard error is the standard deviation of the *sampling distribution* (Lane, 2006). It essentially reflects the sampling fluctuation in the data. These statistics together generally provide a good indication of the data's variability. Due to the presence of different branches and products, each with their own sales behaviour, it is difficult to compare variability among folds. It is however evident that fold 5 has the lowest standard deviation and standard error. This suggests the sales behaviour of fold 5 to be the most consistent among the folds. We additionally suspect more reliable data capturing during the latter months.

8.2 Bayesian Network Estimate vs. Adjusted Estimate vs. Regression Estimate

In order to investigate the effectiveness of the Bayesian network and learning adjustment component, we compared them with each other and with the results of regression tests conducted on the same data. We consider commonly used sales forecasting performance measures,

described in Chapter 4. We now present these results, in terms of the MAPEs, MAEs, MSEs and Theil U-statistics.

8.2.1 Investigating the Mean Absolute Percentage Errors (MAPEs)

A commonly used statistic for sales forecast comparisons, identified in Chapter 4, is the Mean Absolute Percentage Error (MAPE). **Table 8** presents the MAPE of each approach, for the investigated folds and the data as a whole, while **Figure 23** displays these graphically.

Table 8: MAPE comparison of forecasts

Test Fold	$MAPE_{BN}$	$MAPE_{Adjusted\_estimate}$	$MAPE_{Regression}$
1 (May 2004 – Sep 2004)	108.25	118.18	262.20
2 (Oct 2004 – Feb 2005)	131.73	190.78	182.99
3 (Mar 2005 – Jul 2005)	143.03	130.02	188.66
4 (Aug 2005 – Dec 2005)	133.62	174.99	228.52
5 (Jan 2006 – May 2006)	194.89	120.96	295.56
Overall	**142.81**	**154.61**	**230.15**

$MAPE_{BN}$ is the MAPE of the Bayesian network estimates.

$MAPE_{Adjusted\_estimate}$ is the MAPE of the estimates after being adjusted by the learning adjustment component.

$MAPE_{Regression}$ is the MAPE of the estimates determined using regression.

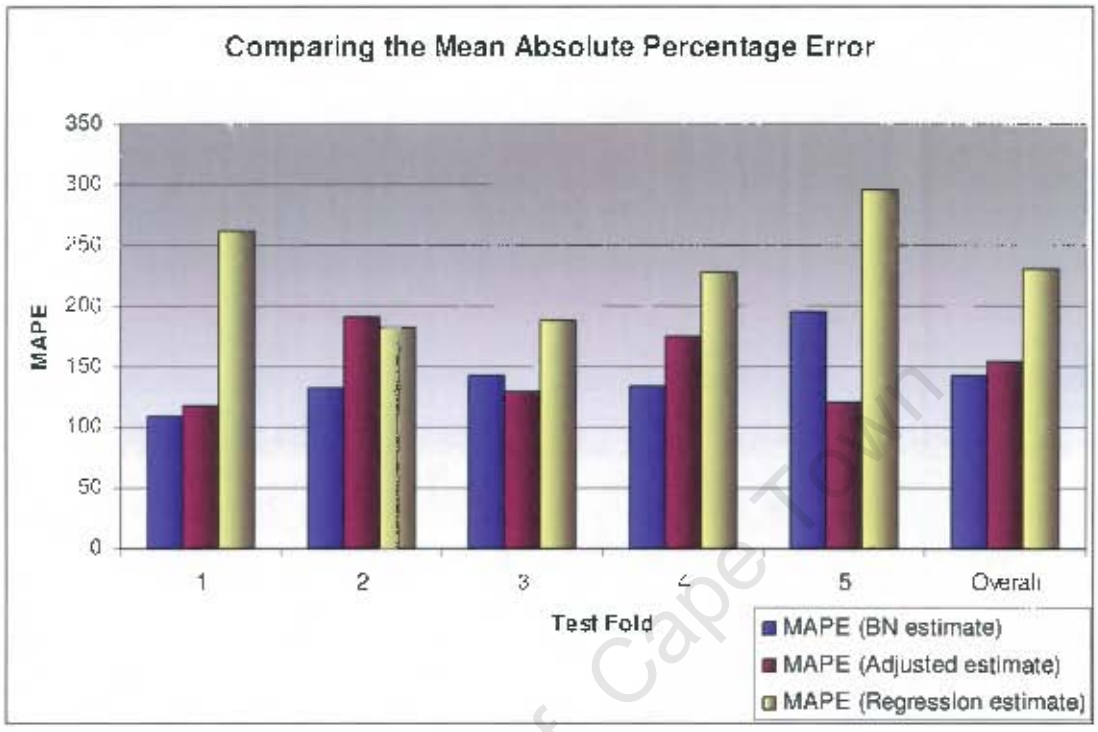

Figure 23: Graph of the MAPE comparison of forecasts

We conducted t-tests using Microsoft Excel's data analysis tool. The hypothesis $MAPE_{BN} = MAPE_{Adjusted\_estimate}$ was rejected at the 99% significance level. Therefore the MAPE for the Bayesian network estimates is significantly different from the MAPE for the adjusted estimates. Similarly, the hypothesis $MAPE_{BN} = MAPE_{Regression}$ was rejected at the 99% significance level. Therefore the MAPE for the Bayesian network estimates is also significantly different from the MAPE for the regression estimates.

These results indicate that, as a whole, the MAPE for the Bayesian network estimates is slightly lower than for the adjusted estimates and substantially lower than for the regression estimates.

However, there appears to be a contradiction. The folds with a higher percentage of correct interval predictions have a higher MAPE relative to those with a lower percentage of correct interval predictions. Consider fold 5, which has the highest $MAPE_{BN}$ of 194.89, but the best correct interval percentage of 77.22%. One would expect a lower MAPE where the results are more accurate. This apparent contradiction can be explained by examining the sales behaviour of

Prima boerewors at the Table View branch, shown in **Figure 24**.

In early months (those belonging to fold 1) there is a high demand for Prima boerewors at the Table View branch. This demand decreases over time, and by 2006 (the months belonging to fold 5) it exhibits low sales. As there is a wide spread of data, the discretisation intervals are quite large. Therefore, although the Bayesian network estimates the correct sales intervals for the latter months, the estimated value is somewhat larger than the actual value. As the actual sales are very low, this results in a large Absolute Percentage Error (APE) when dividing by a small value. Similar patterns are evident for loin chops and Grabouw boerewors. These combine to cause a higher MAPE for the latter months.

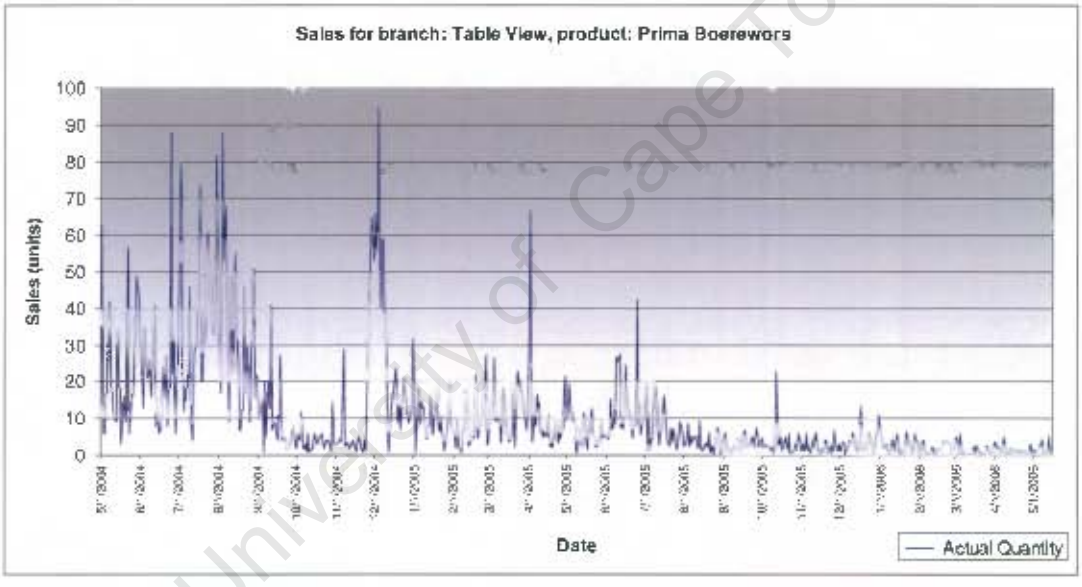

Figure 24: Graph of sales for Prima boerewors at the Table View branch

Perhaps more appropriate measures of accuracy for this data are the Mean Absolute Error (MAE) or Mean Squared Error (MSE). These are discussed in the following section.

8.2.2 Investigating the Mean Squared Errors (MSEs) and Mean Absolute Errors (MAEs)

The problem with considering only the MAPE is that for low sales a high Absolute Percentage Error (APE) will occur. Where the data is spread as in **Figure 24** a high MAPE will be calculated

which does not necessarily reflect the accuracy of the forecast. We now consider alternative performance measurements, based on the research of Chapter 4, in the form of the Mean Absolute Error (MAE) and Mean Squared Error (MSE). These are displayed in **Table 9** and **Table 10**, and graphically in **Figure 25** and **Figure 26**.

Table 9: MAE comparison of forecasts

Test Fold	MAE_{BN}	$MAE_{Adjusted\_estimate}$	$MAE_{Regression}$
1 (May 2004 – Sep 2004)	12.33	8.57	11.84
2 (Oct 2004 – Feb 2005)	11.36	9.65	10.02
3 (Mar 2005 – Jul 2005)	10.44	8.26	9.59
4 (Aug 2005 – Dec 2005)	9.20	8.99	10.74
5 (Jan 2006 – May 2006)	6.644	5.59	9.91
Overall	**9.70**	**8.34**	**10.41**

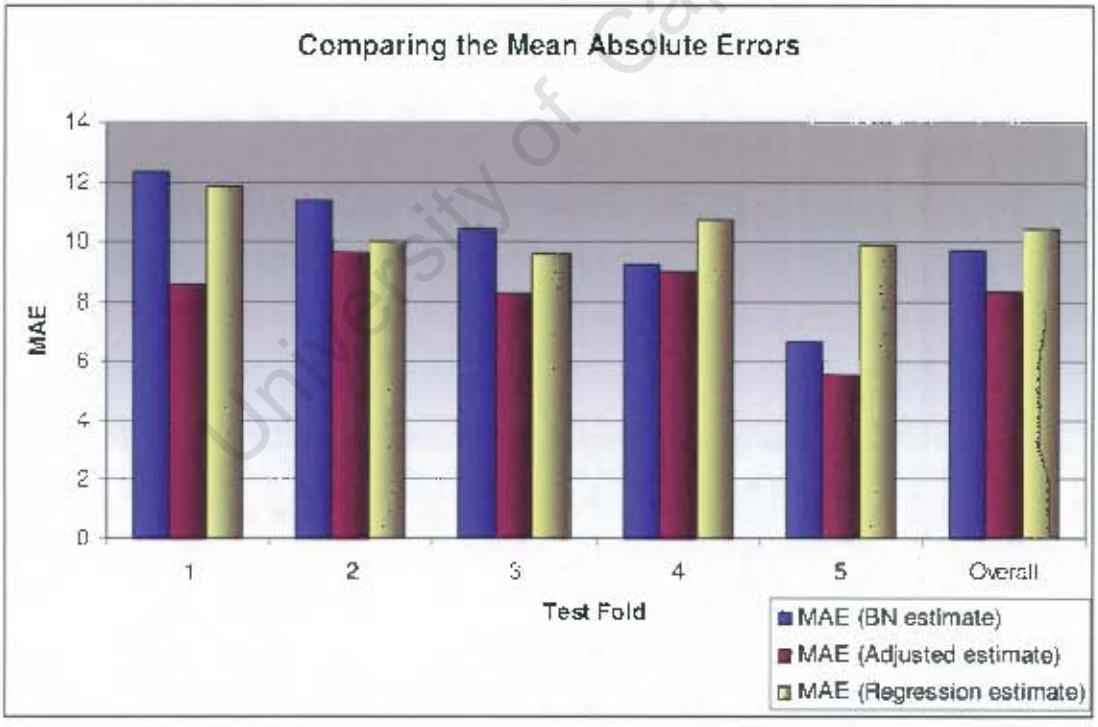

Figure 25: Graph of the MAE comparison of forecasts

Table 10: MSE comparison of forecasts

Test Fold	MSE_{BN}	$MSE_{Adjusted\_estimate}$	$MSE_{Regression}$
1 (May 2004 – Sep 2004)	655.58	270.53	428.75
2 (Oct 2004 – Feb 2005)	623.52	346.00	500.89
3 (Mar 2005 – Jul 2005)	715.26	397.49	423.83
4 (Aug 2005 – Dec 2005)	376.02	291.51	481.78
5 (Jan 2006 – May 2006)	222.92	170.57	362.37
Overall	**478.92**	**293.23**	**448.59**

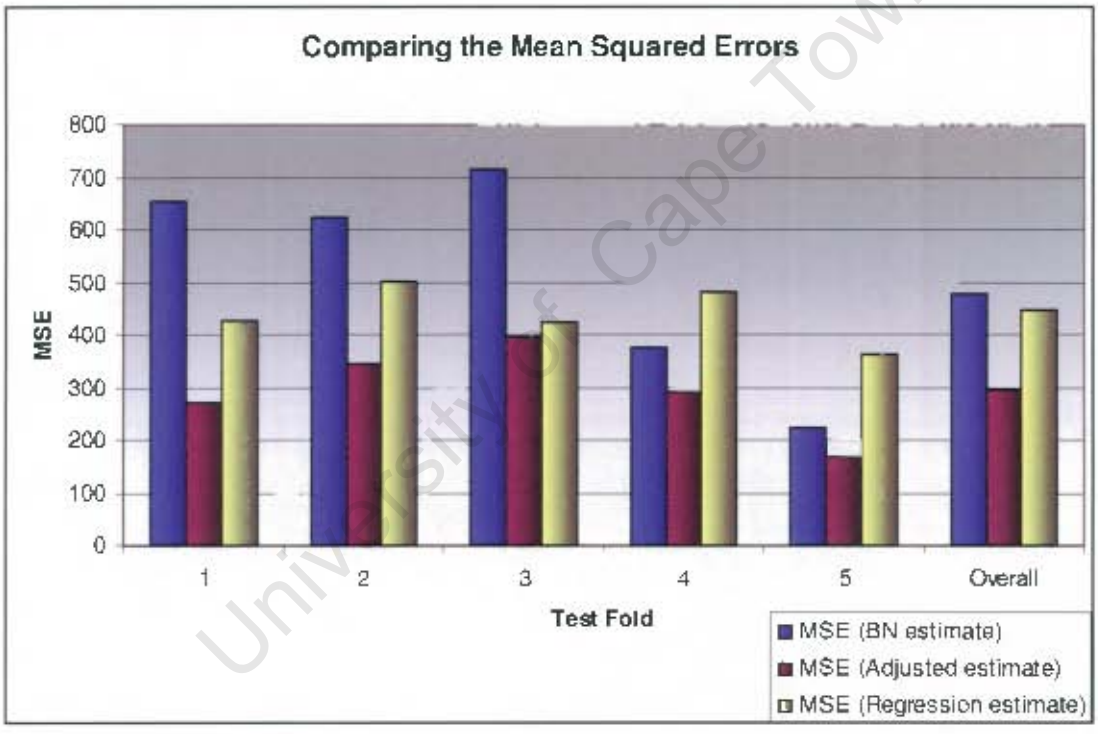

Figure 26: Graph of the MSE comparison of forecasts

Examination of these results shows the regression forecasts to be more accurate than the Bayesian network forecasts for the initial folds, with the MAEs and MSEs lower for folds 1 to 3. However, during the latter months – those which we believe to contain more reliable data – superior performance is displayed by the Bayesian network. Overall, the t-tests reveal a significant difference between MAE_{BN} and $MAE_{Regression}$ (where $MAE_{BN} < MAE_{Regression}$) at the 99% level, but no significant difference between MSE_{BN} and $MSE_{Regression}$ (where $MSE_{BN} > MSE_{Regression}$) even at

the 95% level. This suggests that on the whole, the Bayesian network slightly outperformed the Regression analysis.

For each fold, the Bayesian network with learning adjustment was superior to both the Bayesian network (on its own) and the regression analysis, exhibiting lower MAEs and MSEs on each occasion. Overall, the MAE and MSE of the adjusted estimates were found to be significantly different from those of the other approaches at the 99% level.

8.2.3 Investigating the Theil U-Statistics

Theil's U-statistic calculates a ratio of the accuracy of a forecast relative to the naïve forecast's accuracy. A U-statistic below 1.0 is considered superior to a naïve approach and therefore acceptable as a forecast. The U-statistics are presented for all {Branch, Product} combinations in Appendix B, and displayed graphically in **Figure 27**. The outlier *AP : Lamb 1/2 Lamb Pack*, with a U value of 14.37 for the regression analysis, has been omitted from the graph for visual effect.

Of the 45 combinations, the Bayesian network approach had 27 U-values (60%) below 1.0, the Bayesian network with learning adjustment had 25 U-values (55.56%) below 1.0, and the regression analysis had only seven U-values (15.56%) below 1.0. For all values which the regression analysis U-statistic was below 1.0, both the Bayesian network and Bayesian network with learning adjustment U-values were also below 1.0. Therefore for a majority of the {Branch, Product} combinations the Bayesian network and Bayesian network with learning adjustment were found to be acceptable forecasts. Performance of the regression analysis was considerably worse.

The mean values for all Theil U-statistics were calculated as 1.06, 1.18 and 1.95, for the Bayesian network, Bayesian network with learning adjustment, and regression analysis, respectively. These were compared with the naïve model, which would produce a U-value of 1.0 for each {Branch, Product} combination, to determine the significance of the means. Using the t-test, the Bayesian network mean was not found to be significantly different from the naïve mean. The Bayesian network with learning adjustment's mean was significantly different at the 95% significance level, but not at a 96% level. The regression analysis mean was significantly different at the 97% significance level.

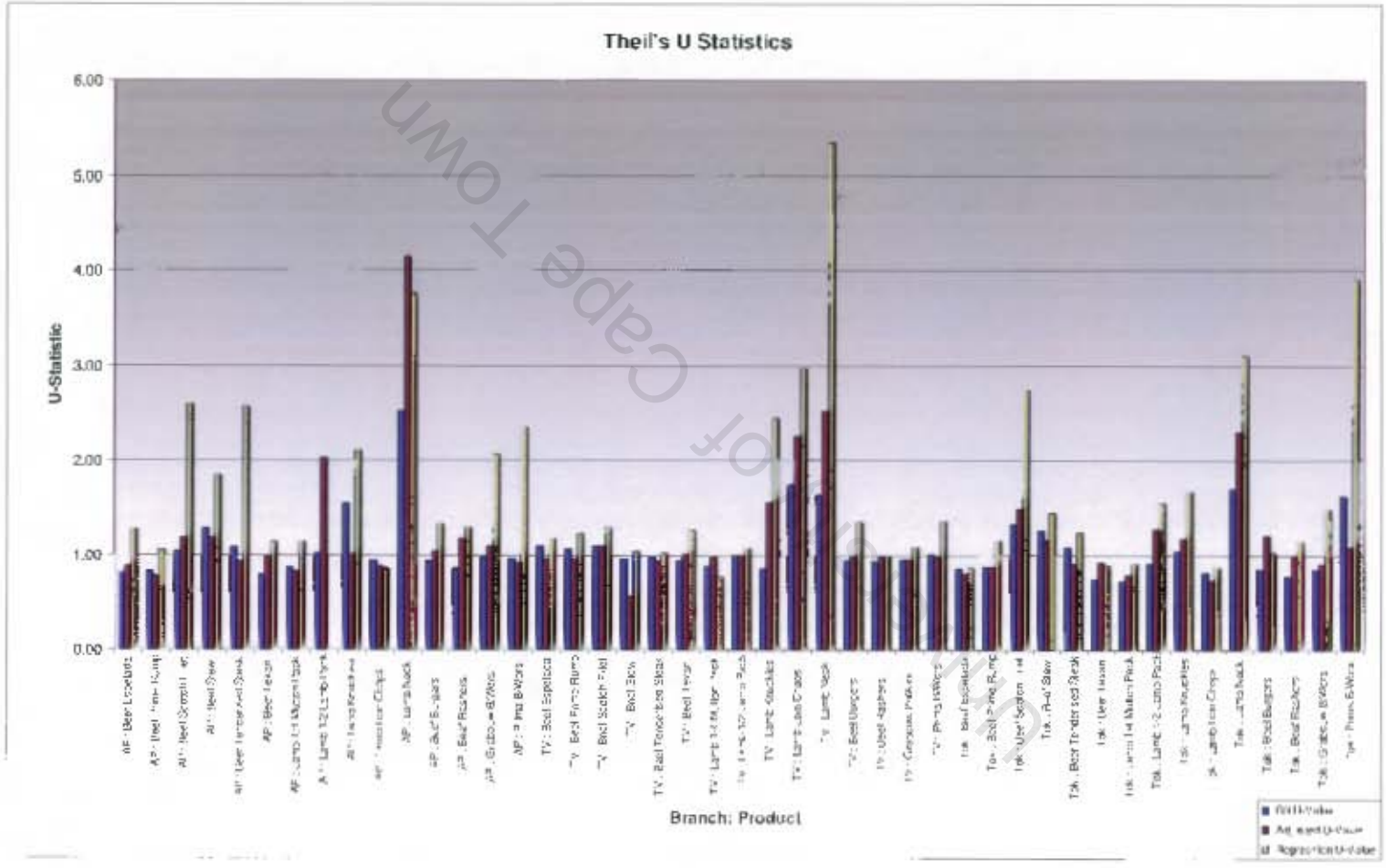

Figure 27: Graph of the U-statistics for all {Branch, Product} combinations

Based on these U-statistics, the Bayesian network approach can be considered an acceptable sales forecasting approach for the given data. The Bayesian network with learning adjustment could also be argued to be acceptable, with a U-statistic below 1.0 for a majority of the combinations, and a mean which is not significantly different from the naïve model at the 96% significance level. The regression analysis showed less favourable results and should be discarded for a majority of the {Branch, Product} combinations.

8.3 Further Analysis of the Learning Adjustment

The addition of our learning adjustment component to the Bayesian network forecasts produced favourable results when analysing the MSE and MAE, and acceptable results when considering the MAPE and Theil U-statistics.

The learning adjustment component served two purposes:

1. To adjust the discretised forecasts produced by the Bayesian network towards a closer approximation of actual sales.
2. To allow for faster learning than could be achieved through Bayesian learning alone.

To investigate the latter, we tested the system with increasing quantities of test data (and decreasing quantities of training data). We anticipated that the learning adjustment would be more effective with limited training data and increased test data.

In this section we present the results of these tests and consider the merits and possible limitations of the learning adjustment component.

8.3.1 Investigating the Effects of Increasing Test Data Quantities on the Adjusted Estimates

In **Table 11** we present a comparison of the statistical measures of accuracy in the form of the MAPE, MSE and MAE, for increasing quantities of test data (and decreasing quantities of training data). The measurement values tend to increase as the quantity of test data increases and the quantity of training data decreases. This is expected as the Bayesian network results are more accurate when trained with more training data. The adjusted estimates are based on the Bayesian estimates and should therefore also increase. However, given our initial assumption that the

adjustment is more effective where training data is limited, one would expect that the accuracy measures would increase at a slower rate for the adjusted estimates compared with the Bayesian estimates, as the test data increases.

Table 11: MAPE, MSE and MAE comparisons for increasing quantities of test data

Test Data Quantity	$MAPE_{BN}$	$MAPE_{Adj}$	MSE_{BN}	MSE_{Adj}	MAE_{BN}	MAE_{Adj}
3 Months (Jan, Mar, May 2006)	196.74	110.99	113.16	70.82	5.87	4.58
5 Months (Jan 2006 – May 2006)	194.90	120.97	222.93	170.57	6.64	5.60
8 Months (Oct 2005 – May 2006)	189.30	126.20	232.16	188.87	7.02	5.82
12 Months (May 2005 – May 2006)	172.56	197.46	511.71	456.23	9.82	9.49

Adj is an abbreviation of Adjusted_estimate.

Inspection of **Figure 28** and **Figure 29** indicate that it is perhaps the accuracy measures of the Bayesian network estimates that increase at a slower rate when compared with those of the adjusted estimates. These graphs plot the MAE and MSE values for increasing quantities of test data, comparing the Bayesian estimate approach with the adjustment approach.

Consider how the MAE of the of the adjusted estimates is somewhat smaller than that of the Bayesian estimates when using 3 months' test data, but is almost the same as that of the Bayesian network estimates when using 12 months' test data. One would expect the opposite to occur if the adjustment algorithm was, in fact, more effective for limited training data.

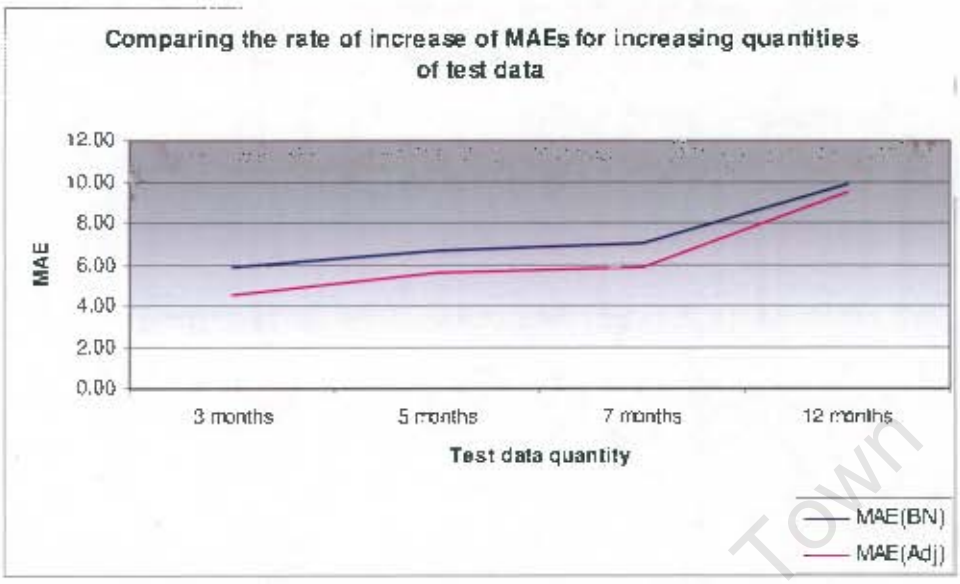

Figure 28: Graph displaying the rate of increase of the MAEs for increasing quantities of test data

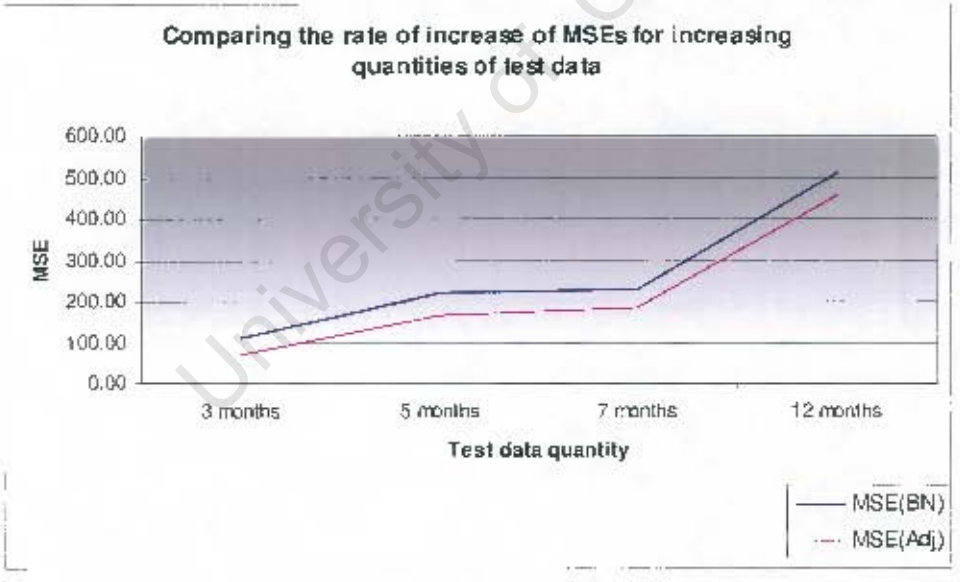

Figure 29: Graph displaying the rate of increase of the MAEs for increasing quantities of test data

From **Table 11**, it is evident that the $MAPE_{BN}$ decreases, while the $MAPE_{Adjusted\_estimate}$ increases, for increasing quantities of test data.

The mean Theil U-statistics, presented in **Table 12**, again disprove the assumption that the adjusted estimates are more effective for limited training data. In **Table 12**, the mean U-statistic for the adjusted estimates increases for increasing quantities of test data, while the mean U-statistic for the Bayesian estimates decreases.

Table 12: Mean U-statistic comparisons for increasing quantities of test data

Test Data Quantity	Mean U-Statistic$_{BN}$	Mean U-Statistic$_{Adj}$
3 Months (Jan, Mar, May 2006)	1.62	0.92
5 Months (Jan 2006 – May 2006)	1.68	1.05
8 Months (Oct 2005 – May 2006)	1.57	1.01
12 Months (May 2005 – May 2006)	1.38	1.66

Based on these findings, we conclude that the learning adjustment component does not necessarily increase the learning speed of the system.

8.3.2 Merits and Limitations of the Learning Adjustment Component

When considering the MSE and MAE, the learning adjustment was found to significantly improve accuracy. However, while the MAPE of the adjusted values were at times lower than the MAPE of the Bayesian estimates, overall the Bayesian estimates were found to have a preferable MAPE. We now examine this behaviour and draw conclusions regarding possible merits and limitations of the learning adjustment component.

Figure 30 compares the various sales estimates with the actual sales for loin chops at the Table View branch, for test fold 2. It is evident from the graph that in the initial months, where sales remain consistently low, the Bayesian estimates are adjusted appropriately towards the actual sales. From December 2004 to February 2005 there is a spike in sales, where the sales of loin chops are substantially greater. This is not detected by the Bayesian network or regression analysis as the training data contains no such spikes. In fact, examination of the Bayesian network estimates, for this data, shows a constant estimate, as the estimated interval remains the same throughout.

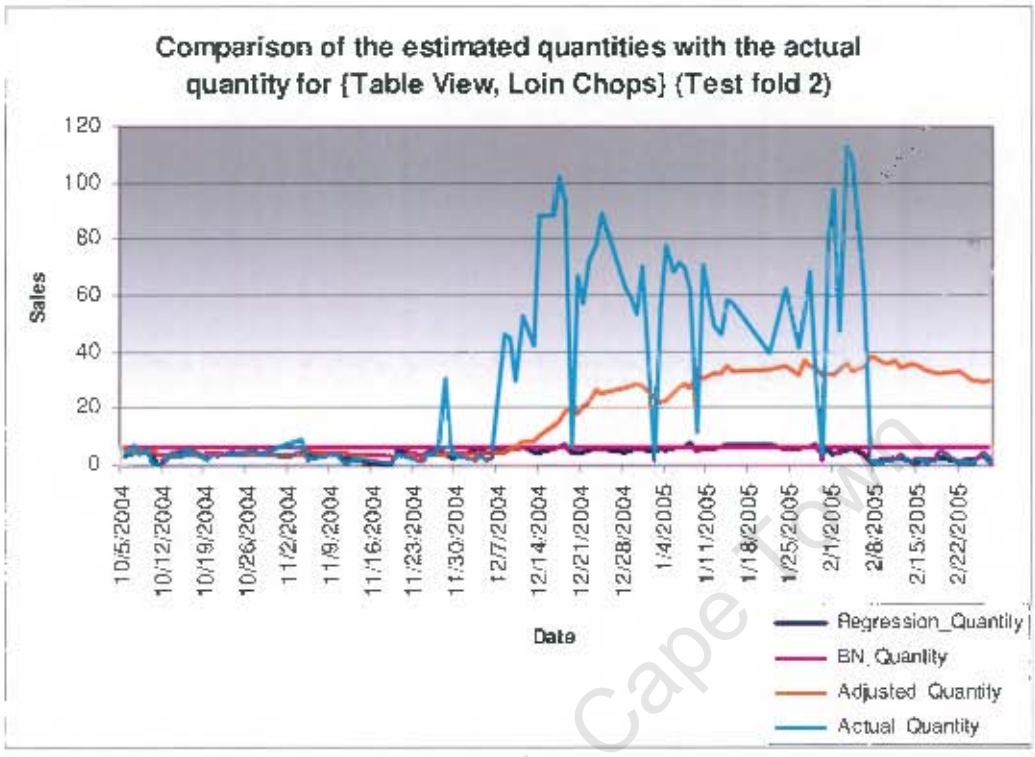

Figure 30: Graph of the comparison of estimates with the actual quantity sold for loin chops at the Table View branch, test fold 2.

The adjustment is effective in adjusting values towards the higher sales. However, where sales dramatically decrease again, the adjustment algorithm still calculates high values based on the recent behaviour. High APEs are calculated towards the end of February where the relatively high sales are divided by substantially lower sales. This results in a higher MAPE than the accuracy would suggest.

The adjustment algorithm is therefore effective where sales remain fairly consistent or when sales follow an upward or downward trend. It is less effective, and perhaps undesirable, where sales exhibit dramatic fluctuations.

8.4 Conclusion

In this chapter we investigated the effectiveness of a Bayesian network as the internal model of a sales forecasting system, and the performance of our learning adjustment component.

The Bayesian network was able to forecast the correct discretised sales interval for 68.65% of the test data. For each test fold, the correct interval percentage was above 60%, indicating the Bayesian network's ability to estimate the correct sales interval on a majority of occasions. The highest correct interval percentage of 77.22% was found for test fold 5. These months showed to have the most consistent sales behaviour.

We then investigated the accuracy of the Bayesian network relative to a typical exogenous data sales forecasting approach, in the form of linear regression. We compared the Mean Absolute Percentage Errors (MAPEs) of the Bayesian network estimates; adjusted estimates; and the regression analysis estimates. Overall, the MAPE for the Bayesian network estimates was found to be slightly lower than the adjusted estimates and substantially lower than the regression estimates.

However, the test folds with a higher percentage of correct interval predictions had a higher MAPE relative to those with a lower percentage of correct interval predictions. This was explained by considering a {Branch, Product} combination which displayed high initial sales and low sales in the latter months. Therefore, although the correct intervals were being estimated in the latter months, the intervals were quite large and produced a high Absolute Percentage Error (APE) relative to the low sales. With a few such combinations present in the data, a higher MAPE was calculated for the latter months.

We therefore considered alternative statistical performance measures to provide a better gauge of accuracy. These were in the form of the Mean Squared Error (MSE) and Mean Absolute Error (MAE). The regression forecasts were found to be more accurate than the Bayesian network forecasts for the initial folds, with the Bayesian network estimates superior during the latter months. Overall, the t-tests revealed a significant difference between the MAEs of the Bayesian network estimates and regression estimates at the 99% level, but no significant difference between their MSEs even at the 95% level. Therefore, on the whole, the statistics suggested that the Bayesian network slightly outperformed the regression analysis.

The adjusted estimates exhibited lower MAEs and MSEs for all folds when compared with the Bayesian network and regression analysis results. Overall, they were found to be significantly different from the other approaches at the 99% significance level.

Theil's U-statistic, which calculates a ratio of the accuracy of a forecast relative to the naïve forecast's accuracy, was determined for each {Branch, Product} combination using the three approaches. A U-statistic below 1.0 indicates an acceptable forecast. Of the 45 combinations, the Bayesian network approach had 27 values (60%) below 1.0, the Bayesian network with learning adjustment had 25 values (55.56%) below 1.0, and the regression analysis had only seven values (15.56%) below 1.0. The Bayesian network approach and Bayesian network with adjustment were therefore acceptable for a majority of the combinations. The regression approach was less acceptable. In addition, the Bayesian network mean of 1.06, was found to be not significantly greater than a naïve model which would calculate a U-statistic of 1.0 for each {Branch, Product} combination. The Bayesian network with adjustment produced a mean of 1.18, which was significantly different from the naïve model at the 95% significance level, but not at the 96% level. The regression analysis U-statistic mean of 1.95 was significantly different from the naïve model at the 97% significance level.

Based on the investigated statistical measures, the performances of the Bayesian network and Bayesian network with adjustment component were found to be far superior to the regression analysis. Where a minimal average difference between actual sales and estimated sales is desired, the MSE and MAE indicated the Bayesian network with learning adjustment to be preferable. In this respect the Bayesian network (without adjustment) was slightly superior to the regression estimates. Where the goal is to minimise percentage error, the MAPEs indicated that the Bayesian network was the optimal approach of the three. The Bayesian network with adjustment was only slightly inferior, with the regression analysis being considerably worse. Finally, when considering the Theil U-statistics, both the Bayesian network approach and the adjustment approach were found to be generally acceptable, whereas the regression analysis had to be rejected for most {Branch, Product}combinations.

After comparing the various approaches, we turned our focus to the behaviour of the learning adjustment component. We anticipated that the learning adjustment would be more effective with limited training data and increased test data. This was not found to be the case. For increasing test quantities the MSEs and MAEs were found to be increasing at a slower rate for the Bayesian network estimates than the adjusted estimates, in contrast with what had been expected. The mean U-statistics for the adjusted estimates increased for increasing test quantities, while decreasing for the Bayesian estimates. We therefore concluded that, although often providing

more accurate results, the learning component did not necessarily increase the speed of learning. In addition, the adjustment algorithm was more effective where sales remained fairly consistent or when sales followed an upward or downward trend. It was less effective, and perhaps undesirable, where sales exhibited dramatic fluctuations.

Chapter 9

Satisfying the Requirements of an Adaptive Agent Architecture

The theoretical aspect of this research investigates the ability of modelling a sales forecasting system using an adaptive agent architecture. In this chapter we consider the behaviour of our system and analyse it in terms of the identified characteristics of adaptive agent architectures. We pay particular attention to its embodiment of Brooks's (1991) characteristics and consider additional characteristics highlighted in Chapter 2.

9.1 The Complex Behaviour of the System

Before analysing the characteristics of the system, we discuss its complex behaviour. **Figure 31** displays a graphical representation of the workings of the system, indicating how the Bayesian network functions as an internal model and how the competence agency integrates with the BaBe agencies.

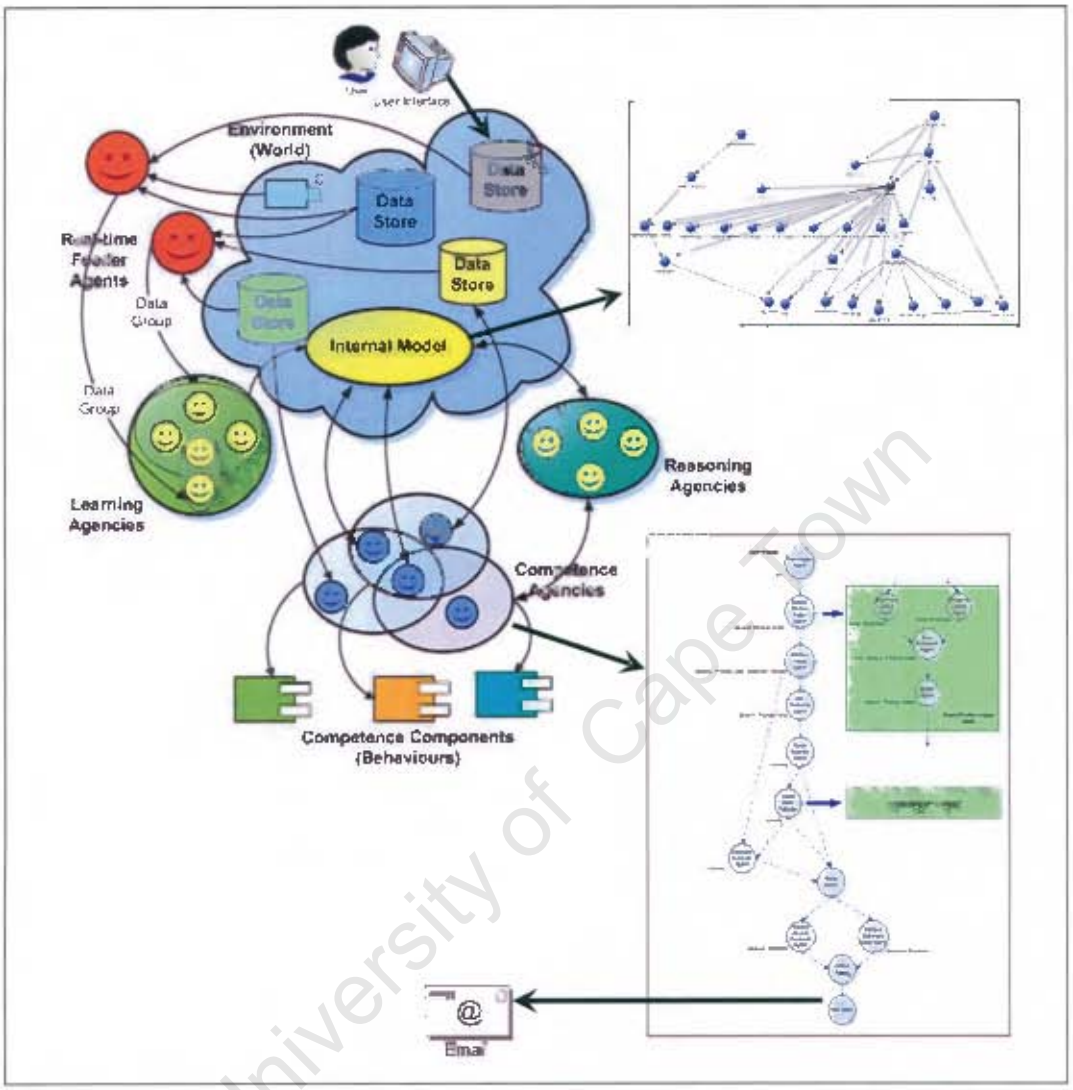

Figure 31: Behaviour of the system

- Exogenous data is captured through the interface and stored in a database, in the form of a data store, as shown in the diagram.
- The real-time feeder agents retrieve this data, together with sales data. They aggregate the data into data groups, which are passed to the learning agents.
- The learning agents use this data to train the internal model – the Bayesian network – using Bayesian learning.
- Each day the competence agents respond to a change in date.

- For each {Branch, Product} combination, they retrieve the observed exogenous factors, for the given date, from the database and set this as evidence in the Bayesian network.
- They interact with the reasoning agents to query the Bayesian network, determining the sales interval with the highest probability.
- The Bayesian estimate is adjusted with the learning adjustment component – a complex agent, consisting of simple agents.
- The estimated turnover is calculated and all estimates are stored in a database, before being autonomously e-mailed to the Meatpackers manager.

The entire global behaviour results from the interactions of simple agents, forming diverse agencies. There is no coordinating authority. Learning and adaptation occurs by continually conducting Bayesian learning on the internal model, extracting patterns from environment data. Feedback is present in the learning adjustment component, which is affected by the output of previous Bayesian estimates. The system is almost completely autonomous (requiring only input of the exogenous data and sales data) – the competence agents are triggered by a change in date and email a result without any interaction with a human user.

9.2 Brooks's characteristics of *reactive planning* systems

9.2.1 Situatedness

Agents are situated in the world. Their behaviour is directly influenced by their environment, as they react to changes external to the system without dealing with "abstract descriptions" (Brooks, 1991: 3)

The system learns from real-world data, as BaBe's real-time feeder agents retrieve data from the environment. These, in-turn, have a direct influence on their communicating agents. Changes to the environment therefore directly influence the agents' actions and the state of the internal model.

9.2.2 Embodiment

Agents would need to have sensors to enable them to interact directly with their environment. Potgieter (2004: 6) explains how agents' "actions are part of the dynamic interactions with the world, and this changes their experience of the world."

The agents communicate through sensors, in the form of listeners, and emitters. Their actions are based on changes to their environment.

9.2.3 Intelligence and Emergence

Agents are observed to be intelligent. The intelligence emerges through the interactions with the environment, and between agents.

Based on the interactions of simple agents, an "intelligent" global behaviour emerges, enabling the system to estimate sales according to the observed environment conditions.

Emergence arises from the interactions of the simple agents and the use of a Bayesian network as an internal model. Through the interactions among feeder, learning, reasoning and competence agents, a sales estimate emerges based on patterns learnt from the external data. Through Bayesian learning, the Bayesian network identifies patterns and relationships which may not be obvious by human inspection.

Referring to the previously described framework for emergence, introduced by Baas et al. (1997), the system could be considered S^2, with the agents S_i^1. S^2 arises from the observation (Obs^1) of the interactions (Int^1) of the agents (S_i^1). The predicted sales are observed in the global behaviour of S^2, but are not apparent in the constituent agents. Therefore $P \in Obs^2(S^2)$ and $P \notin Obs^2(S_i^1)$, where P is the estimated sales and Obs^2 is an observation mechanism which regards the estimated sales.

9.3 Self-organisation

An additional characteristic of reactive and adaptive agent architectures is self-organisation. It implies an organisation through agent interactions without any single agent coordinating behaviour.

The system does not use a central control to coordinate behaviour. There are no algorithms stipulating the order of flow. Consider how the learning agents respond to the output of the feeder agents or how the competence agents respond to a change in date. Each agent performs a single role and is not concerned with the overall behaviour of the system.

Within the competence agents which we implemented, there is no central control. All that is defined is which agents interact with each other.

Consider how the requirement for a loop, which would require some coordinating algorithm, is avoided. This is achieved through the *BranchProduct Trigger* agent, whose role is to provide combinations of branches and products (for a given date) for other agents to manipulate. Having emitted a {Branch, Product, Date} combination, it is not concerned with the behaviour of the system, but simply responds to communicating agents to know when to emit another {Branch, Product, Date} combination. In this way sales are predicted for all branches and products, without requiring a loop to coordinate behaviour.

9.4 Internal Models

An adaptive agent architecture uses an internal model for prediction and decision making. Patterns are extracted from the environment and applied to the internal model in order to anticipate the consequences that follow when the pattern is again encountered.

The Bayesian network serves as an internal model for this system. The system observes patterns from its input stream and adapts the internal model through Bayesian learning. Agents respond to changes in the environment by querying the internal model and making predictions using Bayesian inference.

In addition, the Bayesian network incorporates nodes representing the causal relationships between sales and exogenous factors. It therefore serves Holland's (1995) requirement of reflecting a relationship between a system and its environment, as described in Chapter 2.

9.5 Feedback and Adaptation

The output of one iteration is fed back into the system to be used as input for the next iteration. Adaptive agent architectures continually adapt their behaviour according to the feedback they receive.

Feedback forms part of the learning adjustment component. The results of previous Bayesian estimates are fed back into the system and compared with the actual sales. The *Adjust Sales Estimate* agent learns an adjustment value, which is dependent on the results of previous Bayesian estimates.

In addition, the use of a Bayesian network as an internal model allows for adaptation. Bayesian learning enables the system to learn from its environment. It continually gathers data, in the form of event information; weather conditions; promotions data; pricing data; and advertising details, and relates these factors to sales data. As the system learns more about its environment, the results improve in accuracy and reliability.

9.6 Conclusion

This research proposed implementing a sales forecasting system using an adaptive agent architecture. In this chapter, we discussed the complex behaviour of our implemented system, and analysed it in terms of the identified requirements of adaptive agent architectures.

The system was shown to employ Brooks's (1991) characteristics of "reactive planning" systems and to satisfy the additional requirements of adaptive agent architectures identified in our research.

The agents are situated in the world, as changes to the environment directly effect their actions and the global behaviour of the system as a whole. They demonstrate embodiment in their

communication through listeners and emitters. An intelligence is observed through an emergent behaviour from the interacting agents. The system is able to estimate sales based on the state of its environment and its observed factors. The system demonstrates self-organisation as agents communicate without a coordinating authority. This gives rise to emergence.

Adaptive agent architectures additionally maintain internal models, used for prediction and decision making, and incorporate learning based on feedback and adaptation. The system uses a Bayesian network as its internal model. This enables it to gather patterns from its environment and make decisions when again encountering those patterns, based on probabilistic reasoning. It incorporates feedback through Bayesian learning as well as the learning adjustment component. This enables it to adapt over time to changing environmental conditions.

If we accept the premise that a system which incorporates Brooks's (1991) requirements for reactive planning, together with the ability to adapt and learn, is an adaptive agent architecture, then we may conclude that we have implemented a sales forecasting system using an adaptive agent architecture. We have therefore shown it to be possible to model a sales forecasting system using an adaptive agent architecture, thereby validating our initial hypothesis.

Chapter 10

Conclusion

10.1 Summary

Sales forecasting continues to gain importance, becoming recognised as an integral part of operational planning. However, with the increased use of sales forecasting and the expansion of global industry, forecasting systems are becoming increasingly complex. This is partly due to increased complexity and globalisation, as sales forecasting is required for more products, at more locations and for more business units (Lapide, 2006).

This research investigated an alternative exogenous data sales forecasting approach, based on adaptive agent architectures, to tackle the increased complexity of sales forecasting. In this project, we considered the feasibility of modelling a sales forecasting system using an adaptive agent architecture – an approach to which sales forecasting had seemed well suited. We achieved this using BaBe, an adaptive agent architecture that uses a Bayesian behaviour network as an internal model. This introduced an additional aspect to this research – the effectiveness of a Bayesian network as a sales forecasting technique. We found that while Bayesian networks have been used for forecasting in general, there has been little focus on Bayesian networks for sales forecasting, with perhaps the only real application being Hidden Markov Models, the simplest form of dynamic Bayesian network. The final aspect of this research investigated an agent based learning adjustment component, which employed feedback learning commonly found in adaptive agent architectures. The learning algorithm extended the mean error calculation used in sales forecasting for performance measurement, to include all exogenous factors. BaBe was unable to train the Bayesian network using continuous data and we were limited to discretised estimate intervals. The learning algorithm was required to adjust the discretised forecasts produced by the Bayesian network towards a closer approximation of actual sales, and to increase the speed of

learning.

10.1.1 Work Done

We implemented a sales forecasting system using BaBe – an adaptive agent architecture, designed by Potgieter (2004). The system focused on an existing company, in the form of Meatpackers – a meat wholesale company. Their sales are largely affected by exogenous factors, which we identified with the assistance of the Meatpackers manager. We used BaBe to mine a Bayesian network for the exogenous variables, and altered the network where dependencies seemed incorrect. This introduced a qualitative aspect to a quantitative sales forecasting approach.

The system consisted of a user interface, linked to a database, for capturing exogenous data. BaBe's learning agents would use this data to continually train the Bayesian network, while competence agents would interact with reasoning agents to retrieve an estimate from the Bayesian network for each product, at each branch, on a given date.

The high level behaviour of the system was achieved through implementation of competence agents, developed using the BaBe agent framework. These agents were responsible for retrieving the exogenous data from the database for each product, at each branch, on a given date, and setting the observed values as evidence in the Bayesian network. They would then interact with reasoning agents to query the Bayesian network and retrieve an estimate based on the observed values.

The learning adjustment component was implemented as a complex competence agent, which interacted with the other competence agents. It consisted of a number of simple agents, each concerned with retrieving an adjustment for a different exogenous variable, with separate agents for combining the adjustments and adjusting the Bayesian estimate. The algorithm was based on retrieving an average error of the Bayesian estimate for each of the observed factors, and combining them to form an overall average error. This reflected the amount by which the Bayesian estimate should be adjusted.

The behaviour of the system was almost completely autonomous. The competence agents would be triggered by a change in date, and a suggested estimate would be emailed to Meatpackers without any human interaction.

10.1.2 Testing and Findings

The research proposed a theoretical aspect as well as a practical aspect. The theoretical aspect concerned the ability of modelling a sales forecasting system using an adaptive agent architecture. This would be determined based on whether our system was able to adhere to the identified requirements of adaptive agent architectures, with particular attention being paid to Brooks's (1991) characteristics of reactive planning systems. The practical aspect analysed, quantitatively, the effectiveness of a Bayesian network for sales forecasting, and the performance of the learning adjustment component. For this purpose, we compared the accuracy of the system's forecasts with forecasts made using multiple linear regression, a commonly used exogenous data sales forecasting technique.

The system was found to predict the correct discretised sales interval on a majority of occasions, with an overall percentage correct interval prediction of 68.65%. The interval was correctly predicted on 77% of the data for the final five months, where the sales behaviour seemed most consistent.

An investigation into the commonly used statistical measures for sales forecasting performance measurement indicated favourable results for the Bayesian estimates and adjusted estimates. The Mean Absolute Percentage Error (MAPE) was found to be best for the Bayesian estimates, slightly worse for the adjusted estimates and considerably worse for the regression estimates. The Mean Absolute Error (MAE) and Mean Squared Error (MSE) both showed the average overall error to be lowest for the adjusted estimates, indicating the learning adjustment component's ability to adjust the Bayesian estimate towards a more accurate result for the majority of cases. For these statistics the Bayesian estimates were found to be slightly preferable to the regression estimates.

The Theil U-statistics, which provide an indication of whether a sales forecasting technique is acceptable for the given data, again indicated preferable results for the Bayesian estimates and adjusted estimates. Of the 45 combinations, the Bayesian network approach had 27 values (60%) below 1.0, the Bayesian network with learning adjustment had 25 values (55.56%) below 1.0, and the regression analysis had only seven values (15.56%) below 1.0. The estimates were therefore acceptable for a majority of the combinations when using the Bayesian network approach or Bayesian network with learning adjustment, but were less acceptable for the regression analysis.

In addition, the Bayesian network's mean U-statistic was not significantly greater than that of the naïve model, while the mean U-statistic of the adjusted estimates was significantly different from the naïve model at the 95% significance level, but not at the 96% level. The regression analysis U-statistic mean was significantly different at the 97% significance level.

Overall, the Bayesian network showed superior results to the regression analysis, indicating Bayesian networks as a possibly desirable option for exogenous data sales forecasting. The learning adjustment was found to be effective in adjusting the discretised values towards more accurate results when sales remained fairly consistent, or followed an upward or downward trend. It was less effective when sales fluctuated largely, and it was not found to increase the speed of learning.

Analysis of the system revealed that it behaved according to the identified characteristics of adaptive agent architectures. It incorporated all of Brooks's (1995) requirements for reactive planning, and adhered to the additional requirements described in the Adaptive Agent Architectures chapter. We concluded that one can model a sales forecasting system using an adaptive agent architecture, with this research showing a possible approach.

10.2 Contributions to Research

We have described a method for modelling a sales forecasting system using an adaptive agent architecture. This provides an alternative approach to sales forecasting, incorporating reactive planning and agent-based theory, which might be effective in accommodating for the increasing complexity of current sales forecasting requirements.

We have introduced an additional application to the study of reactive planning and adaptive agent architectures.

We have shown a Bayesian network to be more effective, in this research, than regression analysis, a common exogenous data sales forecasting approach. Bayesian networks have not been widely explored as sales forecasting techniques. This research has shown them to be an acceptable, possibly desirable, approach, worth further investigation.

We have presented a feedback learning algorithm that extends the mean error performance measurement to incorporate exogenous factors. This was found to be effective in adjusting discretised values towards more accurate results when sales behaviour is consistent, or follows an upward or downward trend.

10.3 Limitations and Future Work

The main limitations of this research are:

1. The investigation was conducted for a single company and the findings may not extend towards other companies.
2. The accuracy of the forecasts is dependent on the reliable capture of exogenous data and sales data. We believe that sales data was not always accurately recorded, particularly in the early data. In addition, we were limited to a small amount of promotional and advertisement information and almost no opposition data.
3. BaBe was unable to train the Bayesian network with continuous data. We were therefore restricted to price categories and sales intervals. This resulted in a loss of accuracy, and this research may not have uncovered the true effectiveness of a Bayesian network for sales forecasting.
4. The mined Bayesian network was static, rather than dynamic. Although it incorporated nodes which considered aspects of the date, it might not have dealt with temporal patterns as effectively as would a dynamic Bayesian network.

We therefore propose the following as future work:

10.3.1 Investigate More Companies

A number of diverse companies should be investigated to determine the suitability of Bayesian networks in various sales forecasting systems.

10.3.2 Increase Autonomy for More Accurate Data

An interesting extension to this system would be to attempt to achieve a higher level of autonomous behaviour. The implemented system is reliant on human interaction for the capture

of data. The use of technologies such as RFID tags or bar code readers could allow for the autonomous capture of sales data as soon as items are sold. Exogenous data would be more difficult to capture autonomously, as it would essentially involve agents which crawl the web in search of appropriate weather data, sports data, promotional data and advertisement data – which at some point must be captured by a human.

10.3.3 Introduce Continuous Data

Complex Adaptive Systems (Pty) Ltd. is developing BaBe's ability to handle continuous data. An interesting investigation would be to determine the improvement of accuracy when using continuous data rather than discrete data.

10.3.4 Use a Dynamic Bayesian Network

A comparison might be conducted between the static Bayesian network of this project and a dynamic Bayesian that incorporates the same exogenous data. This could provide a good indication of our Bayesian network's ability to determine temporal patterns.

Additional extensions, not arising from the limitations of this research, might be considered. We propose the following as large scale projects:

10.3.5 Implement a Generic Sales Forecasting System

With BaBe's ability to mine Bayesian networks, a system could be implemented that allows a user to select his/her own data sources and specify his/her own exogenous variables. Alternatively, a user interface could provide enough flexibility for the user to list the exogenous variables and capture the appropriate data manually. Complex Adaptive Systems (Pty) Ltd. is currently developing a "Data Administrator" which could be used to configure the data stores from multiple sources.

The competence agents would have to be adjusted to retrieve the appropriate data from the database for setting the evidence of the Bayesian network and a consistent node name should be

used for the competence agents to query. This would create a generic sales forecasting system that could be adapted to suit any company.

10.3.6 Introduce Policy Analysis

The system could be extended to include policy analysis apart from simply estimating sales. The Bayesian network could be used, for example, to consider the effect on sales if the prices decrease or adverts are introduced. As the system calculates turnover, it could be used to determine the influence of price to obtain an optimal turnover.

10.4 Concluding Remarks

We have suggested the investigation into adaptive agent architectures, in order to tackle the growing complexity of sales forecasting. Bayesian networks have shown to be effective means of conducting sales forecasting, particularly as they are less complex to design than econometric models and they are able to handle multi-collinearity better than regression techniques. An adaptive agent architecture that incorporates Bayesian networks, such as BaBe, is an effective alternative to implementing sales forecasating systems. This approach is well worth further investigation in not only sales forecasting, but other systems exhibiting similar complexity.

References

Australian Bureau of Statistics. 2006. *Commonwealth of Australia* [online]. Available from: http://www.abs.gov.au/AUSSTATS/abs@.nsf/97adb482c0aba769ca2570460017d0e7/cbed2b492aacee6eca2570d7001ad22a!OpenDocument. [Accessed 11 July 2006]

Baas, N. A., Emmeche, C. 1997. On Emergence and Explanation. *Intellectica*. 25, 67-83. Available from: http://www.nbi.dk/~emmeche/coPubl/97d.NABCE/ExplEmer.html

Bradfield, D., Underhill, L. 2000. IntroSTAT. Second Edition. Cape Town: JUTA.

Brooks, R. A. 1991. *Intelligence Without Reason. MIT AI Memo 1293* [online]. Available from: http://people.csail.mit.edu/brooks/papers/AIM-1293.pdf . [Accessed: 10 October 2005]

Bobrow, D.G., Marks, J., Weld, D. S. 2006. *The Role of Intelligent Systems in the National Information Infrastructure. The American Association for Artificial Intelligence* [online]. Available from: http://www.aaai.org/Library/Reports/nii.php#RTFToC55 [Accessed: 10 October 2006]

Cape Argus. 2006. *Cape Argus & Independent Online (Pty) Ltd* [online]. Available from: http://www.capeargus.co.za/ [Accessed 31 August 2006]

Chiva-Gomez, R. 2004. Repercussions of complex adaptive systems on product design management. *Technovation.* 24 (2004) 707-711. Elsevier Ltd.

Frank, E., Witten, I. H. 2005. *Data Mining. Practical Machine Learning Tools and Techniques. Second Edition.* San Francisco, CA: Morgan Kauffman Publishers.

Fruit & Veg City. 2006. *A Fresh Start* [online]. Available from: http://www.fruitandvegcity.co.za/tiki-index.php?page=fvc_story_02. [Accessed 16 October 2006]

Gell-Mann, M., Millikan, R.A. 1990. *The Santa Fe Institute* [online]. Available from: http://www.santafe.edu/research/publications/workingpapers/91-03-017.pdf [Accessed 4 June 2006]

Glasgow, N., Longstaff, D., Sibthorpe, B. *Complex Adaptive Systems: A Different Way of Thinking About Health Care Systems. Australian Primary Health Care Research Institute.* [online]. Available from: http://www.anu.edu.au/aphcri/Publications/Background_paper_stream1.pdf . [Accessed 2 June 2006]

Größler, A., Schieritz, N. 2003. Emergent Structures in Supply Chains – A Study Integrating Agent-Based and System Dynamics Modeling. *Proceedings of the 36th Hawaii International Conference on System Sciences.* 3(3), 94.1.IEEE Computer Society.

Holland, J. H. 1995. *Hidden Order: How Adaptation Builds Complexity.* Reading, Mass: Helix.

Hyndman, R. J., Makridakis, S., Wheelwright, S. C. 1997. *Forecasting: Methods and Applications.* Third Edition. Toronto: John Wiley & Sons.

Jensen, F. 1999. Gradient Descent Training of Bayesian Networks. Symbolic and Quantitative Approaches to Reasoning and Uncertainty. *European Conference, ECSQARU* . 99, 190-200.

Kauffman, S. 1995. *At Home in the Universe: The Search for Laws of Complexity.* Harmondsworth: Penguin.

Lane, D. M. 2006. *HyperStat Online Statistics Textbook* [online]. Available from: http://davidmlane.com/hyperstat/index.html. [Accessed 10 September 2006]

Lapide, L. 2006. Evolution of the Forecasting Function. *The Journal of Business Forecasting.* Spring, 22-28.

Maes, P. 1989. How to do the right thing. *Connection Science Journal.* 1(3), 291-323.

MacDonald, I. L., Zucchini, W. 1997. *Hidden Markov and Other Models for Discrete-valued Time-series.* London: Chapman & Hall.

Mauboussin, M. J. 1997. Shift Happens. On a New Paradigm of the Markets as a Complex Adaptive System [online]. Available from: http://www.capatcolumbia.com/Articles/FoFinance/Fof3.pdf. [Accessed 3 March 2006]

Mentzer, J. T., Moon, M. A. 2005. *Sales Forecasting Management. A Demand Management Approach.* Second Edition. Thousand Oaks, CA: Sage Publications.

Morrison, J. S. 2004. Preparing for Basel II: common problems, practical solutions: part 3: model validation. *The RMA Journal.* [online]. Available from http://findarticles.com/p/articles/mi_m0ITW/is_10_86/ai_n14897531 [Accessed 31 August 2006]

Murphy, K. 1998. *A Brief Introduction to Graphical Models and Bayesian Networks* [online]. Available from: http://www.cs.ubc.ca/~murphyk/Bayes/bayes.html [Accessed 18 September 2005]

Murphy, K. 2002. *Dynamic Bayesian Networks* [online]. Available from: http://www.cs.ubc.ca/~murphyk/Papers/dbnchapter.pdf#search=%22dynamic%20bayesian%20network%22 [Accessed 5 October 2006]

Niedermayer, D. 1998. *An Introduction to Bayesian Networks and their Contempory Applications* [online]. Available from: www.niedermayer.ca/papers/bayesian/bayes.html [Accessed 18 September 2005]

Nilsson, N. J. 1998. *Artificial Intelligence: A New Synthesis.* San Francisco, CA: Morgan Kauffmann Publishers, Inc.

Nwana, S. H. 1996. *Software Agents: An Overview* [online]. Available from: http://www.idi.ntnu.no/emner/dif8914/kompendium-2004/papers-2004/a-nwana96.pdf#search=%22deliberative%20agent%20architecture%20internal%20symbolic%20model%22. [Accessed 9 October 2006]

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* San Mateo. Morgan Kauffmann Publishers, Inc.

Potgieter, A., 2004. *The Engineering of Emergence in Complex Adaptive Systems.* Thesis (PhD). University of Pretoria, Pretoria

Scott, J. 2001. *Principles of forecasting: A Handbook for Researchers and Practitioners.* Boston: Kluwer Academic Publishers.

Shim, J. K. 2000. *Strategic Business Forecasting: The Complete Guide to Forecasting Real World Company Performance.* New York: St. Lucie Press.

Steel, R. 1999. *Complexity & Organisation Development – An Introduction* [online]. Available from: http://www.new-paradigm.co.uk/complex-od.htm. [Accessed: 23 August 2005]

Stacey, R. 1996. *Complexity and Creativity in Organizations.* San Francisco, CA.: Berrett-Koehler.

Sykes, A. O. Date Unknown. *The Inaugural Coase Lecture. An Introduction to Regression Analysis. Working Paper in Law and Economics.* [online]. Available from: http://www.law.uchicago.edu/Lawecon/WkngPprs_01-25/20.Sykes.Regression.pdf [Accessed 11 July 2006]

Webb, G. I., Yang, Y. 2002. A Comparative Study of Discretization Methods for Naïve-Bayes Classifiers. *Proceedings of PKAW 2002, The 2002 Pacific Rim Knowledge Acquisition Workshop.* Tokyo, 159-173.

Wooldridge, M. 1995. Conceptualising and Developing Agents. *Proceedings of the UNICOM Seminar on Agent Software.* 25-26 April, 40-54.

Wooldridge, M., Jennings, N. R. 1995. Intelligent agents: Theory and Practice. *The Knowledge Engineering Review.* 10(2), 115-152.

Appendix A

The Database Schema

The database is required for storing environment data, as well as forecasted sales used by the learning adjustment algorithm.

Figure 32 displays a schema diagram for the database. Tables exist for storing weather information, events, promotions, advertisements, prices and sales. MeatCity's advertising is distinguished from opposition advertising with separate tables – MC_Advertising and Opp_Advertising. Similarly, MC_Promotions and Opp_promotions distinguish the promotions.

In the diagram, PK indicates the table's Primary Key, while FK refers to a Foreign Key. An arrow from one table, A, to another table, B, indicates a Foreign Key constraint from table A to table B. For example, the EventID of the Events table is constrained by the EventID of the Eventtable table.

The Raintable, Windtable, Eventtable, Producttable and Branches tables all serve as lookup tables. Their purpose is to relate IDs, in the form of numbers, to descriptions, in the form of words.

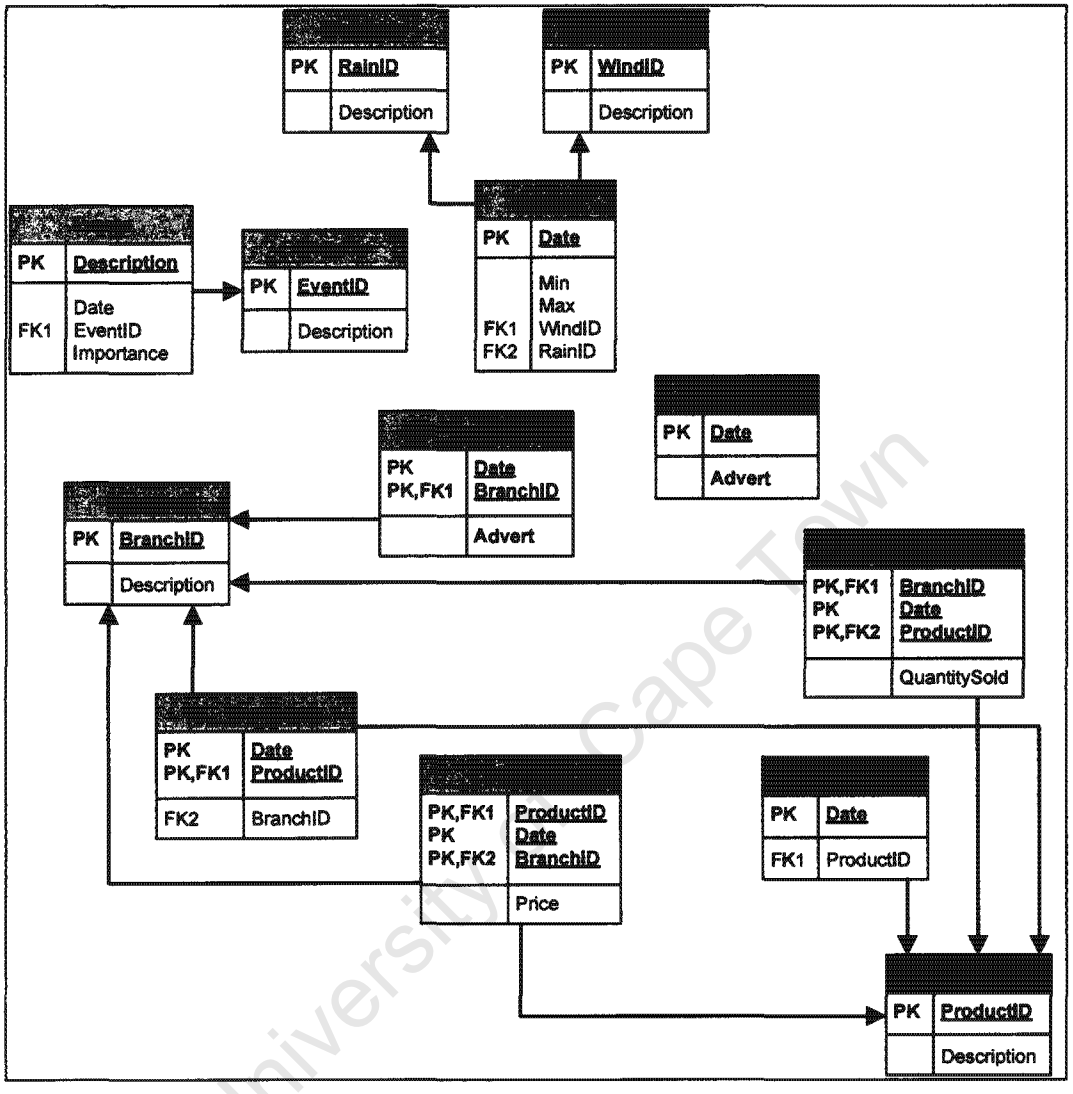

Figure 32: The database schema

Appendix B

The Theil U-Statistics for the investigated {Branch, Product} combinations

Table 13 displays the calculated Theil U-statistics for the {Branch, Product} combinations that were investigated.

Table 13: Theil U-statistics for all {Branch, Product} combinations

Branch : Product	BN U Value	Adjusted Estimate U Value	Regression U Value
Access Park : Beef Espetada	0.81	0.89	1.26
Access Park : Beef Prime Rump	0.82	0.78	1.05
Access Park : Beef Scotch Fillet	1.04	1.18	2.59
Access Park : Beef Stew	1.28	1.18	1.83
Access Park : Beef Tenderised Steak	1.08	0.93	2.56
Access Park : Beef Texan	0.79	0.99	1.13
Access Park : Lamb 1/4 Mutton Pack	0.86	0.83	1.13
Access Park : Lamb 1/2 Lamb Pack	1.01	2.02	14.37
Access Park : Lamb Knuckles	1.54	1.00	2.10
Access Park : Lamb Loin Chops	0.94	0.88	0.85
Access Park : Lamb Neck	2.51	4.14	3.76
Access Park : Beef Burgers	0.93	1.03	1.32
Access Park : Beef Rashers	0.85	1.17	1.29
Access Park : Grabouw B/Wors	0.99	1.10	2.05
Access Park : Prima B/Wors	0.96	0.92	2.34
Table View : Beef Espetada	1.09	0.95	1.17
Table View : Beef Prime Rump	1.06	0.95	1.22
Table View : Beef Scotch Fillet	1.09	1.09	1.29
Table View : Beef Stew	0.95	0.57	1.04
Table View : Beef Tenderised Steak	0.97	0.92	1.02
Table View : Beef Texan	0.94	1.01	1.26
Table View : Lamb 1/4 Mutton Pack	0.89	0.97	0.76
Table View : Lamb 1/2 Lamb Pack	1.00	1.00	1.06
Table View : Lamb Knuckles	0.85	1.55	2.44
Table View : Lamb Loin Chops	1.73	2.25	2.96
Table View : Lamb Neck	1.64	2.51	5.34
Table View : Beef Burgers	0.94	0.98	1.34
Table View : Beef Rashers	0.93	0.99	0.98
Table View : Grabouw B/Wors	0.95	0.95	1.08
Table View : Prima B/Wors	1.01	0.99	1.35
Tokai : Beef Espetada	0.85	0.80	0.87
Tokai : Beef Prime Rump	0.87	0.87	1.15
Tokai : Beef Scotch Fillet	1.33	1.48	2.73
Tokai : Beef Stew	1.26	1.16	1.44
Tokai : Beef Tenderised Steak	1.08	0.91	1.24
Tokai : Beef Texan	0.74	0.91	0.89
Tokai : Lamb 1/4 Mutton Pack	0.73	0.79	0.91
Tokai : Lamb 1/2 Lamb Pack	0.90	1.27	1.54
Tokai : Lamb Knuckles	1.04	1.17	1.66
Tokai : Lamb Loin Chops	0.81	0.73	0.86
Tokai : Lamb Neck	1.69	2.29	3.10
Tokai : Beef Burgers	0.87	1.20	1.02
Tokai : Beef Rashers	0.77	1.00	1.13
Tokai : Grabouw B/Wors	0.84	0.91	1.47
Tokai : Prima B/Wors	1.62	1.08	3.90
Average	**1.06**	**1.18**	**1.95**