Universidade do Porto

Faculdade de Ciências

FCUP

Alexessander da Silva Couto Alves

Internet Traffic Engineering
An Artificial Intelligence Approach

MI - Departamento de Ciência de Computadores

Porto, Março de 2004

TM

Alexessander da Silva Couto Alves

Internet Traffic Engineering
An Artificial Intelligence Approach

Departamento de Ciência de Computadores
Faculdade de Ciências da Universidade do Porto

March / 2004

O orientador:
[illegible]

O co-orientador
[illegible]

Alexessander da Silva Couto Alves

Internet Traffic Engineering
An Artificial Intelligence Approach

Tese submetida à Faculdade de Ciências da Universidade do Porto para obtenção do grau de Mestre em Ciência de Computadores

Departamento de Ciência de Computadores
Faculdade de Ciências da Universidade do Porto
March / 2004

Abstract

Due to the increasing popularity of the Internet, problems with current mechanisms for its management are becoming apparent. In particular, it is increasingly clear that the Internet network does not provide sufficient support for the efficient control and management of traffic, i.e. for Traffic Engineering.

This dissertation addresses the problem of traffic engineering on the Internet from an Artificial Intelligence perspective. It argues that Internet traffic engineering should be automated instead of being performed reactively as response to undesirable network states. It presents and evaluates mechanisms for Internet traffic engineering based on Artificial Intelligence methods.

This dissertation also discusses a suitable architecture for the application of the proposed mechanisms. It argues that the proposed mechanisms are able to support a wide range of services useful for both users and operators. Finally, in a network of the size of the Internet consideration must also be given to the deployment of the proposed solutions. Consequently, arguments for and against the deployment of these mechanisms are presented and the conclusion drawn that there are a number of feasible paths toward deployment.

The work presented argues the following: firstly, it is possible to implement mechanisms within the Internet framework that extend current protocols and algorithms and enable traffic engineering to be carried out by operators; secondly, that applying these mechanisms reduces human intervention together with the management problems faced by operators and at the same time the efficiency with which the network can be run is improved; thirdly, that these improvements can correspond to increased network performance as viewed by the user; and finally, that not only the resulting deployment but also the deployment process itself are feasible.

Resumo

Devido à crescente popularidade da Internet, torna-se aparente alguns problemas com os mecanismos actuais da sua gestão. Em particular, é cada vez mais notório que a rede Internet não apresenta o devido suporte para controlar e gerir o tráfego. Isto é, para a realização de Engenharia de Tráfego.

Esta Dissertação aborda o problema da engenharia de tráfego na internet com enfase nas questões de inteligência artificial. Argumenta que a engenharia de tráfego deve ser automatizada e não realizada apenas em resposta a estados indesejaveis da rede. Apresenta e avalia mecanismos para a realização de engenharia de tráfego na internet baseada em metodos da inteligência artificial.

Esta dissertação também discute uma arquitectura apropriada para a aplicação dos mecanismos propostos. Argumenta que os mecanismos propostos são capazes de suportar diversos tipos de serviços úteis tanto para os utilizadores como para os operadores. Finalmente, numa rede com as dimensões da internet deve ser tomada em consideração a aplicação das soluções propostas. Consequentemente, argumentos a favor e contra a aplicação destes mecansimos são apresentados e conclui-se que existem alguns caminhos viáveis para a sua aplicação.

O trabalho apresentado argumenta que: Primeiro, que é possivel implementar mecanismos dentro do quadro da Internet que extendem os protocolos e algoritmos actuais e permitem que a engenharia de tráfego seja realizada pelos operadores; Segundo, que aplicando este mecanismos reduz-se a intervenção humana nos problemas de gestão enfrentados pelos operadores, ao mesmo tempo que é melhorada a eficiência com que a rede é gerida; Terceiro, que os melhoramentos correspondem a uma melhoria do desempenho da rede vista pelo utilizador e; Finalmente, não só a resultante interligação destes mecanismos na rede mas também o próprio processo de interligação é viável.

to my parents.

Acknowledgements

I welcome the opportunity to express my gratitude to Eugénio Oliveira and Rui Camacho for their guidance, commitment and collaboration, making it possible for me to advance and extend my knowledge within the field of Artificial Intelligence.

I would like to thank Maria Eduarda Silva for introducing me in the field of Time Series Analysis and Forecasting.

I would also like to thank Jorge Martins de Carvalho for providing me with related work on Distributed Control; José Fernando da Costa Oliveira for insightful comments on the optimisation model proposed; Pavel Brazdil for providing me with related work on time series forecasting using neural networks; João Gama for the discussions that helped to improve the consistency of our proposals on machine learning; Luis Torgo for his support on the R language and; Joaquim Pinto da Costa for his advice on statistics.

I am grateful to John Smith at Wisconsin University for discussions on his paper about the relationship between Likelihood and the Mean Square Error.

I wish to thank Alan Montgomery at Carnegie Mellon University for providing me with the datasets and a hardcopy of his paper on time series analysis and forecasting of the USA unemployment.

I wish to acknowledge the work of Olivier Marin at University of Le Havre on the DARX framework and thank him for kindly donating his code to implement our agencies.

I wish to thank Boriss Silvertovs at German Institute for Economic Research and Dick van Djick at Erasmus University, for providing me with the datasets used in their technical report on time series analysis and forecasting the Industrial Production of the G7 countries.

I am grateful to André Restivo, Daniel Moura and Guilherme Pereira for their many helpful comments and suggestions on the review of the text of this dissertation.

I wish to thank my parents, my family and my friends for their support during the elaboration of this dissertation.

Preface

One of the global challenges that nations face at the beginning of the new millennium is the construction of the so called Information Society. The Information Society, as discussed today in the United Nations, is a vision of a new society centred on people, inclusive, and development-oriented.

In this context, the Information and Communications technologies play a fundamental role on the shortening of distances and time, two traditional obstacles to progress and welfare of peoples. These technologies, by their own nature, can reach all corners of the world and benefit millions of people. Where all of them may create, access, use and share information and knowledge, allowing individuals, communities and peoples to realize their full potential in promoting sustainable development and improving their quality of life.

A key aspect on the construction of the Information Society is connectivity. The universal, ubiquitous, equitable and affordable access to information and knowledge constitutes one of the pillars of the Information Society and at same time a technological challenge. In this context, the Information and Communication technologies are a cornerstone on the undertaking of this challenge since they provide access in any place and almost instantly to information and knowledge by individuals, organisations and communities. It is therefore of fundamental importance to provide data, voice, image and video services in a single integrated telecommunication environment, in order to allow technically and affordably the access in a single technological support, to a wide range of services, from best effort to interactive real-time video communication. Here, the Internet and its protocol suite play a fundamental role, due to its technical characteristics and economical benefits.

A fundamental step on the convergence to an Internet-centric access to information and knowledge is the improvement of the Internet reliability as well as the guarantee of its quality of service. This shall be accomplished by service providers operating their networks accordingly, mainly through the realization of mechanisms that allow providing effectively and efficiently these services with the required quality levels. Thus, the research for these mechanisms becomes a fundamental step on the consolidation of the Internet as a feasible technology to access information and knowledge. In this context, we present in this dissertation a proposal based on an Artificial Intelligence approach to the realization of these mechanism on an automated basis focused on the management of traffic within the core of Internet networks.

Contents

Abstract 3

Resumo 4

List of Tables 16

List of Figures 18

1 Introduction 19

1.1 Main contributions 20

1.2 Thesis structure 21

2 Traffic Engineering Automation 22

2.1 Introduction 22

2.2 Traffic Engineering 24

2.3 Traffic Engineering Automation Systems 25

2.4 Conclusions 27

3 Multi-Agent System Architecture 28

3.1 Introduction 28

3.2 Related Work 30

3.2.1 Swarm-intelligence routing 30

3.2.1.1 Analysis of a Swarm-Routing System 31
3.2.2 Multi-Agent Systems for network management 33
3.3 Multi-Agent Systems Concepts 33
3.3.1 Agent 34
3.3.2 Multi-Agent System 34
3.3.3 Agent internal architectures 34
3.4 Architecture 35
3.4.1 Network Dimensioning Agent 38
3.4.2 Traffic Demand Forecasting Agent 39
3.4.3 Path-Restoration Agent 40
3.4.4 Routing Agent 40
3.4.5 Fault-tolerance mechanism 41
3.4.6 Interoperability with the network and external systems 42
3.5 Conclusions 42

4 Time Series Analysis and Forecasting **44**
4.1 Introduction 44
4.2 Time Series 45
4.3 Prediction Theory 45
4.4 Nonlinear Dynamics Theory 47
4.4.1 Overview 47
4.4.2 Delay Coordinate Embedding 48
4.4.3 Attractors 49
4.4.4 Mutual information 49
4.4.5 False nearest neighbours 51
4.4.6 Lyapunov Exponents and Fractal Dimension 52

4.4.6.1 Maximal Lyapunov Exponent 52
4.4.6.2 Fractal Dimension 52
4.5 Stationary Time Series Models 53
4.5.1 Autoregressive Moving Average Models 53
4.6 Non-stationary and Seasonal Models 54
4.6.1 Time series differencing 54
4.6.2 Stationarity and Invertibility 54
4.6.3 Autoregressive Integrated Moving Average - ARIMA 55
4.6.4 Seasonal ARIMA Process 55
4.6.5 Holt-Winters Models 57
4.7 Non-Linear Time Series Models 57
4.7.1 Simple Nonlinear Prediction 58
4.7.2 Local Linear Prediction 58
4.7.3 Global Function Fitting 59
4.8 Time Series Analysis Methodologies 60
4.8.1 Selecting variables to be forecasted 61
4.8.2 Selecting data transformations 61
4.8.3 Selecting outlier handling procedures 61
4.8.4 Selecting independent variables 61
4.8.5 Selecting model classes 62
4.8.6 Model Identification 62
4.8.7 Selecting parameter estimation method 63
4.8.8 Selecting model validation criteria 63
4.8.9 Model selection criteria 63
4.8.10 Selecting exceptional forecast handling procedures 63

4.8.11 Selecting models combination procedures 64

4.8.12 Selecting ways to use human judgement 64

4.9 Applying Time Series Analysis Methodologies 64

4.9.1 Experimental Settings 65

4.9.2 Results and Discussion 66

4.10 Conclusions 68

5 Automating Time Series Analysis and Forecasting 69

5.1 Introduction 69

5.2 Related work 70

5.3 Inductive Logic Programming Framework 71

5.4 Improving Numerical Reasoning In ILP 71

5.4.1 Search Improvements 72

5.4.2 Model Validation 73

5.4.2.1 Other Relevant Statistical Tests 75

5.4.3 Stopping Criteria 76

5.4.3.1 Gaussian White Noise 76

5.4.3.2 Strict White Noise 77

5.4.4 Model Selection 79

5.4.4.1 Hypothesis Complexity 80

5.4.4.2 Model Selection Criteria 81

5.4.4.3 Choosing a Model Selection Criterium 83

5.4.4.4 Dealing with Model Selection Instability 83

5.4.4.5 Model Averaging 84

5.4.4.6 Hypothesis Averaging and Theory Averaging 85

5.4.5 Search Strategy and Pruning at Theory-Level 85

5.4.5.1 Search Complexity 86

5.4.5.2 Pruning 87

5.4.6 Supporting Numerical Reasoning in ILP systems 89

5.5 Automating Time Series Analysis and Forecasting using an ILP System 90

5.5.1 Model identification 90

5.5.2 Selecting variables to be forecasted 90

5.5.3 Selecting data transformations 91

5.5.4 Selecting model classes 91

5.5.5 Selecting model structures 91

5.5.6 Selecting independent variables 91

5.5.7 Selecting parameter estimation methods 92

5.5.8 Selecting model validation criteria 92

5.5.9 Selecting outlier handling procedures 92

5.5.10 Selecting exceptional forecast handling procedures 93

5.5.11 Selecting model combination procedures 93

5.5.12 Selecting ways to use human judgement 93

5.6 Experiments 93

5.6.1 Overview 93

5.6.1.1 Summary of experiments 94

5.6.2 The Learning Task 94

5.6.3 Datasets 95

5.6.4 Benchmark models 96

5.6.5 Background Knowledge and General Settings 96

5.6.6 Startup experiment - An artificial dataset 96

5.6.6.1 Recall Number 97

5.6.6.2 Results 98

5.6.6.3 Results Discussion 99

5.6.7 Assessing Model Validation and Model Selection 100

5.6.7.1 Overview and Goal 100

5.6.7.2 Procedure 101

5.6.7.3 Experimental Settings 101

5.6.7.4 Results and Discussion 101

5.6.8 Assessing Theory Level Search 102

5.6.8.1 Overview and Goal 102

5.6.8.2 Procedure 102

5.6.8.3 Experimental Settings 103

5.6.8.4 Results and Discussion 103

5.6.9 Assessing Hypothesis Averaging and Theory Averaging 103

5.6.9.1 Overview and Goal 103

5.6.9.2 Procedure 104

5.6.9.3 Experimental Settings 104

5.6.9.4 Results and Discussion 104

5.7 Conclusions 105

6 Network Dimensioning **106**

6.1 Introduction 106

6.2 Network dimensioning characterisation 106

6.2.1 Scalability 107

6.2.2 Link overloading 108

6.2.3 Load balancing 108

6.2.4 Feasibility 108

6.2.5 Traffic QoS guarantees 109

6.3 Related work 109

6.4 Network dimensioning model 110

6.5 Simulation results 112

6.6 Conclusions 113

7 Planning Connections Re-routing **115**

7.1 Introduction 115

7.2 Planning under uncertainty 116

7.3 Decision Theory Methodologies 117

7.3.1 Relaxing Independence 119

7.3.2 Relaxing Completeness 119

7.4 Problem Formulation 120

7.5 Probabilistic Model 121

7.6 Decision model 125

7.7 Experiments 125

7.7.1 Experimental setup 125

7.7.1.1 Traffic Patterns 126

7.7.2 Experiments results 127

7.7.3 Results analysis 129

7.8 Conclusions 129

8 Conclusions **131**

8.1 Summary 131

8.2 Future Work 132

8.3 Conclusion 133

A Acronyms **135**

References **138**

List of Tables

4.1 Results on forecasting the link workload of the *Abilene* communications network with a time horizon of 60 hours. The link workload is measured in bits per second (bps) 66

5.1 Model Selection Criteria 82

5.2 Models induced 98

5.3 Summary of results 99

5.4 Summary of results of the Relative FRMSE of the ILP algorithm and other benchmark models for the selected datasets. ILP^* stands for the ILP system with our proposals activated and ILP is the same ILP system with our proposals deactivated 102

5.5 Relative improvement on the FRMSE of Theory Level Search 103

5.6 Relative FRMSE resulted from averaging hypothesis and theories after applying ILP using a model selection criterium 104

List of Figures

2.1 *Bandwidth Broker* and the *DiffServ* architecture. (Adapted from [TAPF01]) 23

3.1 Schematic representations of the internal architecture of two agent types: (a) Model of an utility-based agent; (b) Model of a learning agent . . . 35

3.2 Multi-Agent System Architecture 36

4.1 Diagram of a time series prediction problem 46

4.2 Plots of several attractors examples: (a) Lorenz Attractor - The "Butterfly Wings" problem in weather forecasting; (b) Rossler Attractor ; (c) Cisco IOS TCP/IP sequence number attractor - Attack vulnerability 20 %; (d) FreeBSD TCP/IP sequence number attractor - Attack vulnerability 0.0 % 50

4.3 Plots of results on forecasting the link workload of a communications network with a time horizon of 60 hours: (a) SARIMA with seasonal period of 1 week; (b) Local linear fit; (c) Global function fits: Radial basis function and Polynomial Function; (d) Holt-Winters model with seasonal period of 1 week 67

5.1 Typical behaviour of the test and train error of a learning algorithm as a function of the model complexity 80

5.2 Pruning hypothesis based on the descriptive accuracy 88

5.3 Plots of results on forecasting the link workload of a communications network: (a) Step 1 - No model validation, selection and evaluation; (b) Step 3 - Bad parameterisation of the ILP system; and (c) Step 4 - All features enabled and ILP system configured using recall number formulae100

6.1 Plots of average, standard deviation and maximum link load of a communications network for several traffic profiles: (a) load at 40%; (b) load at 70%; (c) load at 80%; (d) load at 90%; (e) load at 100%; (f) load at 120%. The dark colour represents the results obtained by our approach. The bright colour represents the results obtained using a Shortest Path Algorithm 113

7.1 The definition of $\Phi(b)$ 123

7.2 The effect of uncertainty on the distribution parameters: (a) The effect of estimation error on the mean of the forecasted error; (b) The effect of estimation error on the variance of the forecasted error 124

7.3 Plots of the traffic load of several links: (a) Traffic load on the Chicago-Indianapolis link; (b) Traffic load on the Atlanta-Houston link; (c) Traffic load on the Cleveland-Indianapolis link; (d) Traffic load on the Houston-Kansas link 127

7.4 Plots of the load discarded: (a) A comparison of maximum expected utility and conservative decision policies applied to planning under uncertainty; (b) A comparison of both SPF and MCF optimisation algorithms for a planning scenario; (c) A comparison of both maximum expected utility and conservative decision policies applied to planning under uncertainty (uncertainty=0.31); (d) A comparison of both maximum expected utility and conservative decision policies applied to planning under uncertainty (uncertainty=2.37) 128

Chapter 1

Introduction

The emerging information society demands communication services that allow integrated use of audio, image, video and text data within a single telecommunication environment. Network services providers must therefore operate their networks to provide such services to end-users. The process of managing the allocation of network resources to carry traffic subject to user-specified quality of service constraints is known as Traffic Engineering. The goal of Traffic Engineering is to increase efficiency of network resources utilisation, while assuring that quality of service constraints are met.

Traditionally, Internet traffic engineering activities have been performed with direct human intervention. However, these activities are becoming more demanding and data intensive. This is specially noticed in large-scale public IP networks, due to the growing network size, complexity and multi-service operation. On this scenario, the fundamental challenge that traffic engineering poses to artificial intelligence is the realization of automated control capabilities that adapt quickly and cost-effectively to significant changes in network state, while maintaining stability. This would ideally be accomplished pro-actively using forecasting techniques to anticipate future trends and applying actions to obviate the predicted undesirable future states.

This dissertation addresses the problem of Internet traffic engineering from an Artificial Intelligence point of view. It argues that the traffic engineering process model may be formulated as an adaptive feedback control system, in which a team of agents act on behalf of the traffic engineer, monitoring the network state and intending actions with the goal of driving the network to a desired state, or a state with maximised expected utility.

The work presented argues that a Multi-Agent System architecture that extends current network management systems provides mechanisms for efficient control and management of traffic in a scalable, resilient and fault-tolerant way. It argues that such architecture encompasses activities such as: (i) sensing the network, (ii) forecasting future traffic demands, (iii) mapping traffic demands onto network resources, (iv) allocating spare capacity to the traffic demands projections, (v) and keeping network resilience.

As far as forecasting future traffic demands is concerned, this dissertation argues that an ILP system extended with numerical reasoning capabilities is suited for automating time series analysis and forecasting. It argues that such an ILP system can integrate expert knowledge and human judgement naturally. It improves on current approaches based on Expert Systems because it provides extended search capabilities, reducing human judgement through the reliance on empirical decisions. It also argues that it can outperform current statistical methods, in the experiments made and, moreover, can discover new time series forecasting models. Thus providing means by which software agents can build forecasting models autonomously.

In the context of mapping traffic demands onto network resources, this dissertation discusses network optimisation algorithms for network management. It presents a formulation that copes with multi-service traffic using linear optimisation algorithms.

In what relates allocating spare capacity to the traffic demands projections, the work presented argues that a planning algorithm for reasoning under uncertainty that maximises expected utility may be extended to allow the implementation of conservative and risk aversion policies. It argues that such policies improve on simple maximisation of the expected utility under specific conditions of uncertainty, providing means by which an agent can autonomously derive a plan for his actions, under diverse uncertainty scenarios, and, hence, cope with the uncertainties arisen from traffic demand projections.

1.1 Main contributions

The contributions are organised into three groups: the overall system architecture; the forecasting automation method and; network optimisation process.

This dissertation proposes a multi-agent architecture that copes with fault-tolerance, survivability, resilience, scalability, data spatially distributed and extensibility issues of telecommunication applications.

It also proposes a mechanism for the realization of automated proactive traffic engineering using forecasting techniques. The forecasting techniques are also automated based on a Machine Learning formulation that merges research on Expert Systems and Inductive Logic Programming for automating time series analysis and forecasting. Several proposals are made to extend Inductive Logic Programming so that these tasks could be accomplished. Some of these proposals are even generalisable to other Machine Learning techniques. The results obtained in experimental data provided evidence supporting the proposals and even have shown improvements on current forecasting techniques.

The process of managing pro-actively the allocation of network resources is based on a network optimisation model that supports the formulation of quality of service constraints and rationalises network resources utilisation. The experimental results have shown an increase on the efficiency of resource usage.

This dissertation also proposes a resource allocation mechanism based on a planning algorithm that considers forecasting uncertainties. Several proposals are made to enhance the planning algorithm that are generalisable to other applications. The experimental results have shown a reduction on the service degradation and on the number of configuration changes on the network.

1.2 Thesis structure

The structure of this dissertation is as follows. Chapter 2 introduces Internet traffic engineering and makes a comparison of several traffic engineering systems, relating them with this work. Chapter 3 presents the architecture of the system and highlights the advantages of using a multi-agent approach to the system architecture. Chapter 4 introduces methodologies for time series analysis and forecasting and presents an experiment made with a traffic demand time series. Chapter 5 presents our time series forecasting automation methodology. Chapter 6 presents our approach to network dimensioning. Chapter 7 presents our approach to plan network configuration changes. Finally, chapter 8 draws the conclusions, presents a summary of each chapter, and points out future work.

Chapter 2

Traffic Engineering Automation

2.1 Introduction

The telecommunications infrastructure is a world in transition. There are a number of trends that contribute to this fact: convergence of traditional telephony and data network worlds; shifting of boundaries between public and private networks; complementary evolution of wireline, wireless, and cable network infrastructures; the emergence of next generation integrated broadband multimedia networks; and the information superhighway.

In such a changing environment service providers try to preserve their investment on the network infrastructure by choosing network technologies that are optimised for fast, profitable service creation and service delivery across data, optical, wireless and voice networks, making new services possible and deliverable while seeking profitability, minimising future capital investments and reducing network operation expenses.

Classical IP does not meet many of these requirements, which lead to the development of the Multi Protocol Label Switching (MPLS) and the Differentiated Services (DiffServ) framework. [ACE+01].

MPLS DiffServ reduces complexity of network operation, addresses scalability, facilitates traffic engineering, and interoperability of multi-vendor equipments on the one hand, and supports diverse service levels on the other. This technology enables the network carriers to provide differentiated services and customers to develop network applications that require different service levels such as HTML web pages, video stream and voice over IP. The access to these services is dependent on the user profile established in the Service-Level Agreement (SLA).

SLAs specify the service contract, which covers a wide range of issues, including network availability and bandwidth guarantees, payment models and other legal and business formalities. The SLA contains a Service Level Specification (SLS) that characterises aggregated traffic profiles and the Per Hop Behaviour (PHB) to be applied to each aggregate. The PHB identifies a particular forwarding treatment that is applied to an aggregated traffic profile in order to assure a given Quality of Service (QoS).

To automate the process of SLS negotiation, admission control and configuration of the network devices, a new component called Bandwidth Broker (BB) may be added to DiffServ networks. Figure 2.1 is a schematic representation of two DiffServ Networks managed by two BBs that interact to negotiate SLSs.

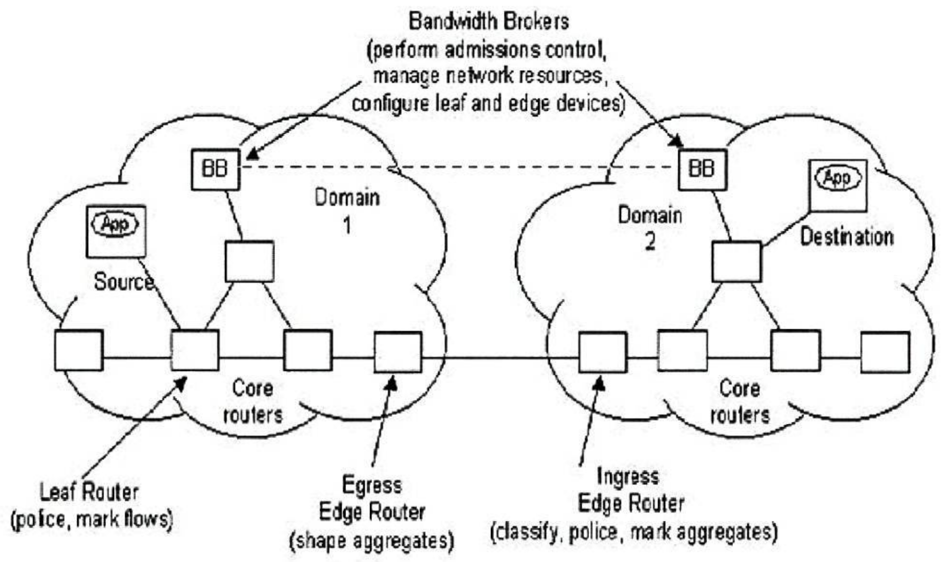


Figure 2.1: *Bandwidth Broker* and the *DiffServ* architecture. (Adapted from [TAPF01])

Due to the complexity and large number of functionalities of a bandwidth broker, the overall behaviour is abstracted into a functional architecture composed by a Traffic Engineer (TE), Policy Management and SLS Management components.

The SLS Management is responsible for negotiating SLSs with other peer Autonomous Systems and users. The Policy Management is responsible for providing a high level interface that dynamically extends the management system functionality. The TE is responsible for selecting paths which comply with the contracted SLS.

The SLS is mapped into Per-Hop Behaviours that assure a given QoS using DiffServ mechanisms for the aggregate flow. However, in order to provide end-to-end QoS guarantees, DiffServ mechanisms should be extended with intelligent Traffic Engineering functions. These Traffic Engineering functions will be the focus of this dissertation.

2.2 Traffic Engineering

Traffic engineering is defined by the Internet Engineering Task Force (IETF) as the aspect of network engineering dealing with the issue of performance evaluation and performance optimisation of IP networks [ACE$^+$01]. Traffic Engineering must efficiently map traffic demands onto network topology and adaptively reconfigure the mapping to changing network conditions.

Following Awduche [ACE$^+$01], traffic engineering methodologies are classified into two basic types: (i) time-dependent and (ii) state-dependent.

In time-dependent traffic engineering, traffic control algorithms are used to optimise resource utilisation in response to long time scale traffic variations (hours, days, weeks). Time-dependent mechanisms use historical information on traffic patterns to pre-program the Label Switch Path (LSP) [1] layout and traffic assignment. An example of a time-dependent algorithm for global resource optimisation is proposed by Mitra [MR99a]. The algorithm formulates the traffic engineering problem with multiple classes of services and multiple paths as a linear multi-commodity problem.

State-dependent mechanisms are used to deal with considerable variations in the actual traffic load that could not be predicted using historical information. These mechanisms are used to deal with adaptive traffic assignment to the established LSPs according to the current state of the network (traffic utilisation, packet delay, packet loss, etc). Examples of state-dependent algorithms are constraint based routing and load balancing among multi-paths. Constraint based routing attempts to route new connection requests minimising delay subject to the resource constraints. Load balancing attempts to distribute traffic across multiple paths between an ingress and egress

[1]The setup of explicit routes, also known as Label Switch Path (LSP), between two nodes in a network reduces operation complexity and is one of the MPLS features which faciliatates traffic engineering.

nodes. A packet enters the network in an ingress node and exits the network in an egress node. A route is a sequence of nodes between the ingress and egress nodes. A packet transverses the network using a given route.

In our approach, we have constrained ourselves to use only MPLS features for traffic engineering. Therefore, we cannot change the traffic control mechanism that already exists in routers. We must, otherwise, deploy our system externally to the network and carry out the control actions using embedded MPLS functionalities.

In this context, our control actions are subject to a relatively high time delay, since they must be sent via network messages to all routers on the network. Therefore, an approach that operates in larger time scales is preferred. Consequently, we propose automating time-dependent traffic engineering mechanisms. We should remark that state dependent and time dependent are complementary techniques. If applied together, they mitigate the effect of short and long term traffic variations on the network performance.

2.3 Traffic Engineering Automation Systems

Time-dependent TE systems may be categorised under two dimensions: (i) off-line and on-line systems along one dimension, and (ii) centralised and distributed systems along a second dimension.

On-line TE systems calculate LSPs as incoming requests arrive. Off-line TE systems use all paths information to calculate the minimum cost path placements.

Centralised TE systems make all routing decisions in a centralised manner. In descentralised TE systems, the ingress node is responsible for calculating a route to the egress node.

Rates [AKK+00] is an online routing server that calculates LSP placements in a centralised manner using a minimum interference algorithm. This approach can lead to degenerate scenarios where the network is in a very sub-optimal state. In this case, Rates does not re-route currently established LSPs for network re-optimisation in an automated manner. It provides a manual mechanism instead. Constraint Based Routing (CBR) shares similar characteristics with Rates' approach, but is a distributed solution. Rates is reportedly faster and more reliable on re-routing upon link-failure than link state algorithms like CBR. Both approaches calculate the LSP placements as incoming requests arrive. Thus, an optimal solution cannot be guaranteed. This has been the reason pointed out for developing off-line routing servers [XHBN00, TAPF01]. By taking all LSP requirements, link attributes, and network topology information into account, the centralised routing server calculates the global optimum LSP placements for the projected traffic utilisation. Nevertheless, since the offline routing goal is to minimise global resource usage, this may lead to routing plans with saturated links, specially if a linear cost function is chosen, as it may be difficult to accommodate new requests. This problem can be avoided if we can re-route existing connections to accommodate the new requests. However, this is often infeasible since it may be necessary to re-route a large amount of LSPs. This issue has no standard solution although it has been tackled by currently deployed off-line routing servers.

Generally, off-line routing supports more efficient protection mechanisms than CBR, since it shares protection capacity that is assumed to be exempt of simultaneous failure [XHBN00]. An important feature of centralised routing is to be faster on re-routing upon link-failure than link-state algorithms like CBR [AKK+00].

An example of an off-line routing server is TEQUILA [Fle02]. In this approach, routing plans are calculated based on forecasts of traffic demand. Several LSP configurations are calculated for each period of the day (morning, evening, night). One inconvenient of this solution is that it cannot adapt to changing traffic patterns caused by topology changes, new LSP requests and variations on user's behaviour. While deploying static configurations is suboptimal re-routing large amounts of LSPs may be infeasible. Chapter 7 describes our approach to this issue. We have developed a planning algorithm to derive a plan for LSPs placement that minimises re-routing accounting for forecasting error uncertainties.

Mitra [MR99a] models the problem of intra-domain provisioning as a linear multi-commodity problem, where the objective is to maximise revenue. This approach has the advantage of using polynomial time algorithms to calculate LSP placements, making it applicable to on-line traffic engineering. One drawback of using a linear function of link load to drive the optimisation is the possibility of saturating LSPs. This has two consequences: (i) increase in the queueing delay, and (ii) if the method is used in off-line routing there is the possibility of not being any remaining capacity left to accommodate the placement of new LSP requests. In our approach we propose to mitigate the problem of link saturation by introducing constraints in the maximum link utilisation, setting it to a predefined level or estimating it adaptively, during problem solving. TEQUILA [PT02] uses a non-linear multi-criteria formulation of the multi-commodity problem. This approach has the advantage of having a better load distribution over network links but it is a computationally more demanding solution.

We propose an intermediary solution between these two that is capable of operating with on-line requirements and coping with load balancing and differentiated services.

Our approach to network dimensioning is based on a linear multi-commodity formulation, which has a polynomial time solution. The issues of load distribution are handled by introducing constraints in the maximum link utilisation, as noticed by Mitra [MR99a]. To cope with more restrictive demands of real-time traffic and differentiated services, we transform the delay and packet loss requirements into constraints for the maximum Hop count, delay and packet loss for an admissible LSP. This approach is similar to [PT02]; however our proposal is more general because it allows the incorporation of several QoS restrictions. A comprehensive discussion of our approach to the issue of network dimensioning is presented in Chapter 6.

TEQUILA uses exponential smoothing to forecast traffic demands. We have compared several methodologies for forecasting traffic demands. The results are presented in Chapter 4. Due to the dynamic and diversified nature of traffic demands, several traffic patterns may exist. The usage of a single model in such cases may be inadequate. We thus propose to automate traffic forecasting in order to allow the consideration of several forecasting models. Chapter 5 presents our approach to the forecasting automation problem.

2.4 Conclusions

The traffic engineering automation problem has several dimensions with no optimal solution for all aspects. This gives rise to trade-off solutions. We have given an overview of the currently available solutions and provided a comparison of traffic engineering automation systems. We have discussed the issues of automating TE centred on the aspects of network dimensioning and traffic forecasting. We have demonstrated, using references to related work, that on-line routing algorithms like CBR do not guarantee optimal LSP placements, and off-line network dimensioning algorithms are constrained by the amount of LSP re-routings that are feasible on an MPLS network. We have discussed trade-off solutions to the problem. The problem of traffic forecasting has been addressed in this chapter as one of the dimensions of traffic engineering for network provisioning. We have analysed other approaches and compared them to our work. The intent of this chapter was to introduce the problem and establish a parallel with similar work in the area, in order to give the reader a context for the subsequent chapters, where our approach is further developed.

Chapter 3

Multi-Agent System Architecture

3.1 Introduction

Telecommunications applications pose strong requirements to software systems such as: reliability, real-time performance, openness, security and other integrated management and mobility. These challenging requirements fit naturally in the Software Agent paradigm [HB98b] and are the motivation for intelligent agent technology in this application domain.

In this chapter we present the system architecture and justify the multi-agent system approach. The goal of this chapter is to situate the system into a network scenario and articulate the multi-agent system functionalities with those currently provided by the network.

We have followed an approach based on the *Principle of Minimum Intervention*[1] in a sense that we extend the already available network control architecture with a *Network Management Layer*. This layer relies on the currently available control functions to perform reactive actions. An example of such functions are the path restoration mechanisms currently available on MPLS networks.

The Network Management Layer is called upon a fault or any extraordinary change in the network state to perform a contingency plan in order to drive the network to an optimised state of utility. This contingency plan will be kept until the fault or the extraordinary event is deemed solved.

[1]Also known as the *principle of subsidiarity* which states that higher layers should not interfere with lower layers taking over its functions

The Network Management Layer periodically monitors network state through performance and fault monitoring facilities of communications networks. Examples of such facilities are: Fault and performance alarms (see SNMP Protocol traps [Ros91]); and performance data collected from network elements like Routers, Interfaces, and telecommunication devices responsible for the data transmission at lower levels.

The periodical monitoring of the network state allows to keep track of the overall system dynamics, which gives the possibility of building up models of the traffic patterns. These models allow the Network Management Layer to perform advanced network management functions like proactive network optimisation as suggested in [ACE+01].

The Network Management Layer specifies a behaviour based on the *Principle of Rationality*[New01]. The principle of rationality states that if the agent knows that one of its actions will lead to a situation preferred according to its goal, then it will intend that action, which will then be taken if it is possible [PW]. However, in dynamical environments, under uncertainties, actions have different levels of confidence on the one hand, and the knowledge about the world has temporary validity, on the other. These factors interact continuously over time, changing the level of confidence on actions. Thus, the expected result of an action, at a given state of the world, may vary with time. Therefore, we complement the rationality component with *adaptive fit capabilities* [And90, Boy01] which also consider learning and uncertainty on the reasoning process. Thus the solution proposed includes both components of rationality and adaptive fit.

The Network Management Layer is comprised by Deliberative Agents that optimise, plan, and build forecasting models in order to keep the network in an optimised state of utility. The optimisation is a rational component while planning under uncertainty and building forecasting models are adaptive components.

The overall architecture includes Deliberative Agents deployed in the Network Management Layer, and Routing Agents deployed in the Network Elements. Such an architecture devises an hybrid-hierarchical multi-agent system.

Our purpose in this chapter is not to provide a detailed review of intelligent agent in telecommunications and communications network management, we leave this to others [Nan97, HB98b]. Rather than presenting an in-depth analysis and critique on the field, we introduce in turn the key concepts and indicate how they are related to the application domain. Where appropriate, we provide references to more detailed work.

Section 3.2 discusses related work found in the literature. Section 3.3 introduces software agents and multi-agent systems. Section 3.4 presents the architecture of the multi agent systems and articulates the requirements of the application domain with multi-agent system abilities. Finally, Section 3.5 draws our conclusions.

3.2 Related Work

In the artificial intelligence literature we may find two different approaches to the problem of network management: One is swarm-intelligence routing [CD98, KA99, SR97, MJ93, Whi97], another is multi-agent systems for network management [WR90, fDANM96, HB98a, CCL98, Vil02].

3.2.1 Swarm-intelligence routing

Swarm-intelligence [DGP+87b, DGP+87a] is an emergent behaviour found in some natural species that exhibit *self-organising* [YG62, NP77] capabilities. There are several mechanisms that lead to self-organisation. Random (or directed) changes can instigate self-organisation, by allowing the exploration of new state space positions. These positions exist in the basins of attraction of the system (see Section 4.4.3 for definition) and are inherently unstable, putting the system under stress of some sort, and causing it to move along a trajectory to a new attractor, which forms the self-organised state. Noise (fluctuations) can allow metastable systems (i.e. those possessing many attractors - alternative stable positions) to escape one basin and to enter another, thus, over time, the system can approach an optimum organisation or may swap between the various attractors, depending on the size and nature of the perturbations [Luc03]. For the interested reader Self-Organisation is a branch of Complex Systems Theory. The importance of the study of self-organising systems is to discover which are the properties of systems that lead them to organisation and stability.

There are several applications of self-organisation mechanisms in artifical intelligence. Examples are un-supervised learning (discovery of patterns in data, learning how data is organised) in artificial neural networks [JKL90, Koh84] and organising groups of autonomous robots [GDMO92, TGGD90].

Self-organising mechanisms may be applied for discovering stable states (with optimum organisation) in systems, for which we may not have an analytical approach. The usage of such methods in network routing is debatable. Network routing is a problem of graph theory, which has many network optimisation algorithms with well supported mathematical analysis. See [RAO93, Ber98, Bie02] for a comprehensive introduction, discussion and analysis of the several methods.

One of the drawbacks of Swarm-intelligence approaches is that it may take a considerable time to learn the best routing path between network nodes [SR97, MJ93]. Another specific problem to stochastic routing algorithms such as Ants-Routing and AntNet is the routing-lock [KA99] problem, which prevents the algorithm to respond on topology changes, locking on a previously learned route.

All these approaches to swarm-routing [CD98, TA02, SR97, MJ93, KA99] explore the environment, collecting state information from the network upon which they attempt in some sense to find the shortest-path to a destination. The information collected characterising the state of each network link may be structured in order to comply to a link adjacency matrix with the correspondent distance metric assigned to each link. In this scenario the Dijkstra's algorithm finds provably the shortest path to a destination. However, all approaches did not follow an analytical formulation.

We propose to analyse a swarm-routing system to illustrate the approach presented in the above paragraph.

3.2.1.1 Analysis of a Swarm-Routing System

AntNet [CD98] approach to network communications routing may be categorised as a vector-distance algorithm, where the distance to a node is measured using the round trip delay. This is an extension to the RIP [Rek88a] routing protocol that incorporates probe packets to check link delays. The probe packets are in this case replaced by ants.

AntNet provides better delay and roughly the same figures for network utilisation than SPF [Rek88b] and OSPF [Moy89] routing protocols. However, the solution has a control overhead 25 times greater than OSPF [CD98] requiring, on average, a proportional increase in processing power at each node, for processing routing messages. It is not possible to assure the ant concentration at nodes near the network minimum cut, which overloads router's CPUs on that region. Up to now the only provable guarantee about the number of ants in a node or a link is that it will be finite [SSY$^+$].

AntNet convergence time is greatly larger than OSPF [TA02]. OSPF algorithm and other link-state algorithms have poor convergence rates because they rely on the exchange of information among routers to propagate network state information. Since both rely on the same principle then, their convergence rate will be approximately equal. OSPF only updates the information that changed while AntNet uses the same propagation mechanism to learn the new route network-wide. Therefore, in case of link failure OSPF has a faster convergence rate. In a worst case scenario AntNet will keep routes because of the routing-lock problem [KA99].

A non-intuitive idea of routing in communications networks is that the shortest path (whatever distance means here) is not the goal of network management. The goal of network management is to optimise resource utilisation while assuring a given service level to the carried traffic. That is the reason why RIP, SPF and OSPF approaches have been replaced by MPLS: to provide means for traffic engineering to optimise resource utilisation and to reduce processing power in network nodes as so as to speed-up recovery time after link failure. Therefore, approaches like [TAPF01] are preferable because using simple routing protocols, with very low overhead, they provide Traffic Engineering with functions that increase resource utilisation on OSPF by 30% while assuring the required service levels. In our approach, we have obtained an increase of roughly 25% using linear programming algorithms.

AntNet improves on SPF, a link-state routing algorithm that also uses probe packets, because it uses a window algorithm to calculate the delay statistics, which smooths the variations, and avoids oscillation on the network routing. However, measuring round-trip delay is a poor estimate of the network delay, since the traffic in the up and down stream links are not balanced on the one hand, and on the other hand, measuring one way delay is difficult because network routers' clocks are not perfectly synchronised. A better approach is to estimate the link utilisation (see [TA02] for proposals on this field) and infer the delay using a queue theory model. An example of this technique is the Kleinrock function [Kle64]. A discussion and assessment of delay on the Internet with recent queue models explained and experimented may be found in [KT01]. These techniques would potentially reduce the overhead of AntNet because only updates are broadcasted. An increase in the accuracy of delay estimation would also be expected.

A possible improvement on this vector-distance algorithm is to allow each ant to store in the node it passes the time spent travelling the path between adjacent nodes and the sequence id. This allows each node to create a matrix with the experienced delay and packet-loss of each link. This would improve overall performance and reduce convergence time because we would have complete information on network links, which allows the usage of Dijkstra's algorithm to find the route to an egress node that has the shortest end-to-end delay. In this scenario, Dijkstra's[2] algorithm finds provably the route with shortest delay, which is, in last analysis, the goal of AntNet.

3.2.2 Multi-Agent Systems for network management

Telecommunications management has traditionally been tackled via a tried and trusted centralised approach. For circuit-switched type telecommunications networks, the dynamics of the logical configuration was kept to a minimum with, for example, fixed bandwidth reservations [HB98a]. MPLS DiffServ allows the realization of specialised services which requires the monitoring of the provided Quality of Service while keeping an optimised level of resource utilisation.

An approach used to tackle this problem is to distribute control and allow specialised agents to handle localised tasks [WR90]. Another approach for reducing complexity is to consider hierarchical control [fDANM96] Our approach explores the strengths of each of these methods and distinguishes them from the used on [HB98a] not only because it adapts to control functions already performed by agents deployed in network elements but also because the maximised utility factor is resource usage efficiency instead of QoS. In [Vil02] an architecture for coordinating bandwidth management and path restoration is proposed. It distinguishes from ours because it uses a subsumption architecture at each network node, while our approach is based on hybrid hierarchical control architecture with reactive agents deployed at each node plus a network-wide *Network Management Layer*, including deliberative agents.

3.3 Multi-Agent Systems Concepts

This section introduces key concepts like Intelligent Agents, Multi-Agent Systems (MAS) and a description of the internal architectures considered in this particular MAS.

[2]There are other algorithms for finding the shortest path to a destination. We focus on this algorithm since it was used in [CD98] to compare with AntNet

3.3.1 Agent

An agent is a computer system that situated in some environment is capable of flexible autonomous action in order to meet its objectives [JSW98]. An agent should receive sensory input from its environment and perform actions that change the environment. It should act without direct intervention of a human or other agents, maintaining control over its actions and internal state. Agents must respond in a timely manner to changes that occur in their environment and maintain a goal-directed behaviour. Nevertheless, they should interact, when appropriate, in order to complete their goals or to support other agents or human activities.

3.3.2 Multi-Agent System

A Multi-Agent System (MAS) may be defined as a loosely coupled network of problems solvers that work together to solve problems which are beyond the individual capabilities or knowledge of each single problem solver [DL91]. In a MAS, each agent has incomplete information or capabilities for solving the problem. There is no global system control, data is decentralised and computation is asynchronous, which means the normal functioning of each agent is independent of each other.

3.3.3 Agent internal architectures

The Agent Internal Architectures considered in this work will be described according to Norvig and Russel approach [RN03]. The agents developed for this MAS may be classified into learning and utility-based agents.

Utility based agents allow deciding which course of action to take in scenarios of choice with multiple conflicting goals, with different importance and different likelihoods of success. Figure 3.1 (a) presents an utility-based agent model. An utility based agent embodies Sensors, Effectors, a memory representation of the State of the World, the capability to predict how the world evolves according to the execution of its actions, information regarding the set of desirable States of the World and the utility of each one. According to this model, the agent should be able to compute a course of action to a goal state with maximum expected utility.

Learning agents allows to decide adaptively the course of actions to take in scenarios of dynamic, incomplete or unknown characteristics of the environment or domain.

Figure 3.1 (b) presents a learning agent model. A learning agent embodies Sensors, Effectors, a Performance Element that maps a State of the World into an Action to take, a Learning Element that discovers the mapping function between a state of the world and the action that maximises the performance measure, a Problem Generator or Database to train the Learning Element and a Critic Element that evaluates the performance of the actions executed, driving the action and response of the Learning Element. According to this model, the agent should be able to learn a mapping between a state of the world and an action that satisfies the performance standard externally established.

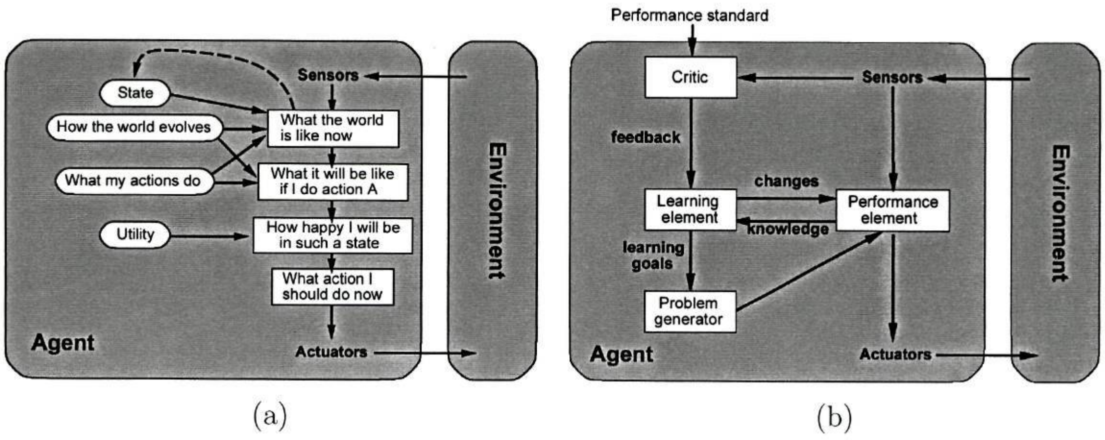


Figure 3.1: Schematic representations of the internal architecture of two agent types: (a) Model of an utility-based agent; (b) Model of a learning agent

3.4 Architecture

The goal of our architecture is to achieve maximum integration with regular network mechanisms and protocols, extending these mechanisms to provide dynamic bandwidth management, proactive optimisation and efficient path restoration mechanisms. The system itself must be scalable, reliable and fault tolerant. This section presents an architecture that tackles these issues.

The system performs dynamic bandwidth management based on an optimisation algorithm, which is executed periodically and recalculates network routes. The routes are modified according to the expected utility of each calculated route. Simulation studies [KMS$^+$01] demonstrated that a bandwidth management system in a DiffServ architecture can be effectively used to provide QoS for real time applications like *Voice* *over IP*. Moreover, these studies also indicate that an admission control mechanism, similar to the proposed in Chapter 6, increases profits by improving network resource utilisation.

The tasks of network dimensioning, forecasting and path restoration are performed by three types of agents. All these agents are deployed at the Network Management Layer. There is one network dimensioning, path-restoration and traffic forecasting agent logically associated with each network element. However, only a single network dimensioning agent is active network-wide at a given time. On each router a routing agent is deployed. Figure 3.2 presents an overview of the multi-agent system and illustrates some of the interactions among its agents and the managed network. A description of the specific agents are given in the next sub-sections.

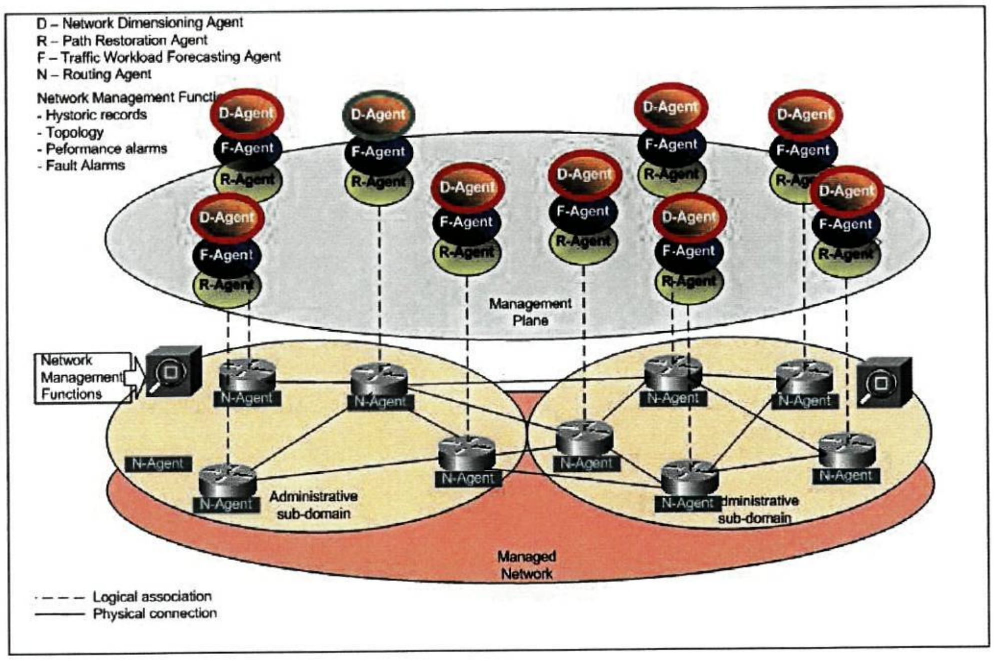


Figure 3.2: Multi-Agent System Architecture

In this work we deal with core networks. The traffic is aggregated at the edge of the network into flows according to the service level. The routes will be calculated for these aggregated flows, so the term flow is always referred to the aggregated flow, unless otherwise noted. A network administrated by an operator is usually divided into administrative sub-domains. Each domain has Operating Support System (OSS) with a performance monitoring system that includes databases with historic measurements of the experienced traffic demand of each flow. Those databases are used by forecasting agents to build forecasting models of the traffic demands. Each forecasting agent is associated with a network element. The agent builds forecasting models and receives forecast requests for all the flows originated in the network element with which the agent is associated.

The performance optimisation is carried out by a single network dimensioning agent for efficiency and efficacy reasons. By taking all network information into the optimisation process, the agent is able to calculate global optimum LSP path placements, while reducing communication overhead in the computation of the solution. The network dimensioning agent is elected at the system start-up and switched back to another agent in case of communication or software fault. The details regarding the fault-tolerance mechanism are described in section 3.4.5. The network dimensioning agent requests to each forecasting agent the forecasts for the expected traffic demands of each aggregated flow. The set of all flows comprises the traffic demands matrix, which is the input of the optimisation algorithm. The matrix with optimal routes is compared to the matrix of currently deployed path placements. The routes are modified according to the expected utility, which considers forecasting uncertainties and current traffic demand.

A fast restoration mechanism is a desirable feature of communication networks. When a fault affects a path, the traffic is switched to the backup path. To achieve fast restoration there are schemes based on pre-planned paths. In these schemes, backup paths can share their bandwidth to increase efficiency. The path-restoration agents calculate backup paths for each link originated on its router as well as the router itself. The path restoration agent communicates to the routing agent the pre-planned backup paths. Routers switch to the pre-assigned backup path in case of failure. The network failures are signalled to the system that triggers the route re-optimisation procedure in order to drive the network to an optimised state of utility.

The agents internal architecture will be described in detail by specifying their perceptions, actions, goals and environment; using Russell and Norvig approach [RN03]. The network dimensioning agent is an utility-based agent - its description is presented in section 3.4.1. The traffic demand forecasting agent is a learning agent - its description is presented in section 3.4.2. The path-restoration is an utility-based agent - its description is presented in section 3.4.3.

The fault tolerance and security issues are addressed at platform-level. Section 3.4.5 presents a more detailed description of the followed approach. Security is achieved through Authentication, Authorisation and Accounting. Deliberative agents will login in the framework in order to communicate and exchange messages. The Routing agents is a third-party agent that uses a different login scheme based on SNMP. Thus, Deliberative agents will also need to make a login in the SNMP agents to which they are binded.

The Interoperability with the network and external systems are addressed at agent-level. Section 3.4.6 presents a more detailed description of the followed approach.

Scalability is achieved through the association of agents to each node of the network. This way the number of agents is directly proportional to the number of nodes, and therefore, to the network size, allowing the distribution of computations and processing. The network dimensioning agent is distributed across network nodes to increase redundancy, but for efficiency reasons, computations are performed centrally. Nevertheless, the results obtained during simulations using real network topologies allowed us to conclude that the response time is satisfactory for the size of problems considered.

3.4.1 Network Dimensioning Agent

The goal of the network dimensioning agent is to deliver the maximum traffic demand while assuring the service levels required by the traffic carried out.

This is done by looking at the future traffic demands and finding for each flow a network path between the ingress and egress nodes that satisfies the stated goal. The optimisation is based on linear programming model described in chapter 6. The output of this algorithm is a plan for routing the future traffic demands.

Now, let us consider the present traffic demands and the uncertainties of our forecasts. In this scenario, the agent faces the problem of how to mix the current and future routing plans minimising the number of re-routed connections and the probability of discarding traffic (failure to carry all demands). The formulation proposed for this problem was based on a decision-theoretic planning algorithm, which decides which connections should be re-routed in order to achieve the intended goal. The utility of a plan is measured by the delay and probability of discarding the traffic. Chapter 7 presents our proposal for such an algorithm taking into consideration several decision-theoretic approaches.

The output of the algorithm is a routing plan that mixes routes that are currently deployed with those that were pro-actively optimised, reducing the number of re-routes and simultaneously maximising the expected utility of the initial deterministic plan that considers average demands.

The forecasts for the traffic demands are calculated by the traffic forecasting agent. The information on how the world evolves is, therefore, obtained interacting with the forecasting agents. The information regarding what the world is like now is obtained from sensors that supply current average levels of traffic in each flow. The effectors are logically coupled to the network through routing protocols that re-configure routing tables located at each router.

3.4.2 Traffic Demand Forecasting Agent

The goal of the traffic demand forecasting agent is to keep the accuracy of the forecasts at a required level of performance. This is done by inducing forecasting models from data. The time evolution of the traffic demand is called a time series. Chapter 4 introduces the key concepts of time series analysis and forecasting. Chapter 5 presents our approach to the task of automating the modelling of the traffic demand time series.

The agent tasks are composed of three steps: Inducing a model, doing forecasts and assessing the accuracy of the forecasts.

Inducing a model takes two input parameters: the level of desired accuracy for the model and the significance level of the statistical tests that assess the degree of fitness of a given model to the data. The level of desired accuracy (maximum generalisation error) is estimated using our proposed theoretical limit stated in section 5.4.3. The level of significance states how probable it is for the induced model to capture the deterministic structure[3] of the traffic demand time series.

The assessment of the forecasts accuracy is based on the same principle used for evaluating the induced models. When the forecasts accuracy becomes under a given threshold, the model is deemed outdated and a new one is induced.

The traffic demand forecasting agent purpose is to create models of the environment for supporting the decision process of the network dimensioning agent and the path-restoration agent. This agent is connected to sensors that monitor average traffic demand on each flow. The problem generator is replaced by a database of historic records of the average traffic demands. This database is assessed through the

[3]In the Wold's theorem sense in which a stochastic process under general conditions is composed of two parts: one completely deterministic and the other completely stochastic

3.4.3 Path-Restoration Agent

The goal of the path restoration agent is to calculate backup paths and global routing plans that maximise the expected utility of the network in case of link or node failure.

This is done by a systematic generation of *what-if-scenarios* of link and router failures. Only one link or router failure is considered at each time. For each scenario the backup paths are re-calculated for all flows. The structure of this agent is similar to the network dimensioning agent extended with the *what-if-scenarios* generator and the network alarms handling procedures.

The agent has two procedures that work at a larger and shorter time scales.

At a longer time scale, the agent calculates backup paths for *what-if-scenarios*. The agent pre-programs these backup paths in the reactive agents (routing agents) deployed in the network elements. This avoids communication latency during the network failure, potentially minimising information (packet) loss during that period. The agent also calculates global routing plans optimised for each failure scenario. This routing plan may be configured in the network after a failure had been detected, thus reducing the recovery-time.

At shorter time scale, the agent acts upon a network failure to implement a pre-planned routing scheme calculated specifically for that fault scenario, which maximises the overall expected utility of the network. The short time scale operation preempts the longer one, including the network dimensioning agent.

The information regarding the current state of the environment is obtained from sensors that supply current average levels of traffic demand and alarms signalling faults or exceptional events, like uncommon bandwidth requests. The effectors are logically coupled to the network through routing protocols that re-configure routing tables located at each router.

3.4.4 Routing Agent

The routing agent is basically a reactive agent that provides a set of functionalities and acts as a response to a given environment state or a request from another agent that is authenticated and authorised for that. Those requests from external agents are performed through the standard communication interface using the SNMP protocol [CF90]. This agent and its communication protocol is documented in the IETF standards and will not be modified by our architecture.

The routing agent is responsible for forwarding traffic, exchanging routing information, calculating routes and performing path-restoration. Through SNMP we may access the following network management functions within the routing agent: (i) performance monitoring, (ii) alarm notification, and (iii) router configuration.

In this architecture, the routing agent communicates with both Path-Restoration and Network Dimensioning agents, namely it forwards alarms to the path restoration agent and receives configuration change requests from both. Although both Path-Restoration and Network Dimensioning agents may perform configuration requests to change the route layout, only the path restoration agent may change the backup paths of the router.

3.4.5 Fault-tolerance mechanism

Multi-agent systems are distributed cooperative applications susceptible to both hardware and software failures. Thus, fault tolerance is required. By ensuring the continuity of computations in spite of failure occurrences, fault-tolerant mechanisms are essential in critical domains such as telecomunications.

It has been shown that replication is the most efficient reliability technique in the presence of failures [GS97]. Therefore we have selected a fault-tolerance mechanism based on this technique.

There are two main types of replication protocols: active and passive.

In the active protocol all replicas process every input message concurrently. While in the passive protocol all input messages get processed by a single replica which transmits its current state periodically to the other replicas in order to maintain consistency.

Active replication strategies lead to a high overhead; however, they provide a fast recovery delay, which is suitable for telecommunications applications with real-time constraints that require short recovery delays.

DARX [MSBG01] is a multi agent system platform that provides several schemes of replication. It supports different replication strategies and group membership management to dynamically add or remove replicas; and it also provides atomic and ordered multicast inside every replication group.

The number of replicas and the internal strategy of a specific task are totally hidden to the other application tasks. This is an interesting feature since it enables the software agent developer to abstract from the selected fault-tolerance mechanism.

3.4.6 Interoperability with the network and external systems

The best way to capture an up-to-date routing table is by configuring the intra-domain routing protocol on the Network Management System. The system appears as one of the nodes of the network to all other network devices as well as to the routing protocol. The very nature of the routing protocol enables the system to capture topological as well as routing information of the complete network. The system has access to performance events and historic traffic demand records through the interface with the network Operating Support System (OSS), which includes management systems available on each administrative sub-domain.

3.5 Conclusions

This chapter intended to substantiate the proposed architecture and to establish a parallel with similar work in the area, providing the reader with a context for the features claimed by this architecture.

We presented a multi-agent system architecture that tackles the scalability, reliability, fault-tolerance, openness, security and real-time requirements of this problem.

We presented the internal architecture of the agents considered in this work, and provided arguments that support the deployment of reactive agents inside network elements to reduce communication latency.

We argued that the adaptability of reactive agents, with local knowledge of the world, may be enhanced by exchanging information with deliberative agents that have network-wide knowledge. In this specific point, the proposed architecture distinguishes from other approaches. This allows the response of reactive agents to network faults to approximate those performed by agents with network-wide knowledge and thus, network-wide optimality.

For the full description of our proposals in the field of Time Series Analysis and Forecasting Automation, Network Dimensioning and Planning of Path Re-Configurations, please, refer to subsequent chapters 5, 6 and 7, where our approach is further developed.

Chapter 4

Time Series Analysis and Forecasting

4.1 Introduction

Time series analysis and forecasting has very important applications in the physical, environmental, social and life sciences domains. The purpose of time series analysis is to gain insight about the data underlying process in order to infer meaningful conclusions about the time series. An important application of time series forecasting is to prevent undesirable events to occur by applying corrective or contention measures. However, time series applications range from the design of discrete control systems to the analysis of the effects of unusual events to a system. This makes time series modelling an important topic of adaptive systems and machine learning disciplines.

The importance of time series analysis in the context of traffic engineering is to provide a means to capture trends and patterns in the usage of network resources. This knowledge allows traffic engineers, or a suited autonomous agent, to pro-actively respond to possible undesirable future states or optimise the management of the network resources.

In the following sections of this chapter we introduce the time series issues relevant to our study. Section 4.2 introduces the basic concepts of time series. Then, we provide an overview of the two most important theories of time series analysis and forecasting: Prediction theory, in Section 4.3; and Nonlinear Dynamics theory, in Section 4.4.

Section 4.5 introduces stationary time series models. Section 4.6 introduces non-stationary and seasonal models. Section 4.7 introduces nonlinear time series models. Section 4.8 introduces a procedure and provides insight about time series analysis. Section 4.9 presents the results of applying the above mentioned time series analysis methodology to network traffic forecasting. Finally, in Section 4.10, we draw the conclusions of this chapter.

4.2 Time Series

Time series is a sequence of observations of a given process' variables.

$$X_t, X_{t+1}, \ldots, X_{t+n}$$

The present work is focused on equally spaced observations, but it is possible to find time series randomly sampled over time.

An intrinsic time series feature is the dependence among observations. Time series analysis is concerned with techniques to explore those dependences, building dynamic models and drawing conclusion about data.

When solving a real world problem, we can make *a priori* inferences about the characteristics of the underlying process that has generated the observations. Therefore a systematic characterisation of time series analysis techniques is a good starting point for the model selection process. Two approaches to time series modelling can be tracked back to prediction theory and nonlinear dynamics theory. The following sections introduce both approaches to time series analysis and forecasting.

4.3 Prediction Theory

One of the most important problem in the study of time series is that of predicting a future value of a process, given a record of its past values. This problem is clearly of interest in economical systems and in the analysis of many types of physical systems. The pioneering work of Wiener [Pri81] on Prediction Theory was stimulated by the problem of predicting the future path of an aircraft in connection with the control of anti-aircraft guns (also known as the problem of fire-control). More recently [BR94] [Whi63], prediction theory has been playing a fundamental role in the theory of stochastic control systems.

The prediction problem can be stated as follows: Given the observed values of a (random) process at some of its past points, predict the value it will assume at some specific future time point. Thus, we are given observations up to time t, i.e. we are given values of X_v, $v \leq t$, and wish to determine the value of $X_{t+\tau}$, where $\tau > 0$. This situation is illustrated in Figure 4.1.

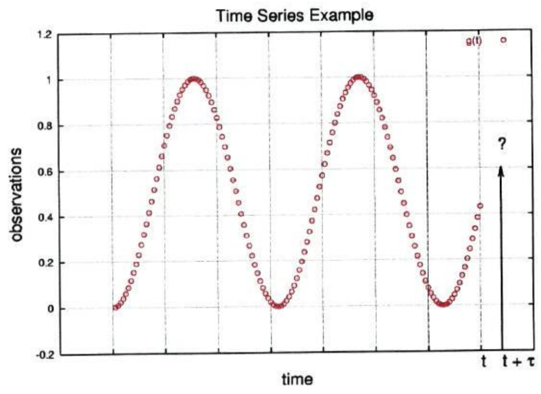


Figure 4.1: Diagram of a time series prediction problem

The predicted value of $X_{t+\tau}$ is calculated as a function of the observations, X_v, $v \leq t$. To determine the function that is the most accurate predictor, Prediction Theory generally uses the Mean Square Error (MSE) [Pri81]. We denote the predicted value of $X(t+\tau)$ by $\tilde{X}_{t+\tau}$. The accuracy is the inverse of the quantity MSE, where MSE is defined as the average over all realizations of the process' squares of the differences:

$$\text{MSE} = \text{E}\left[(X_{t+\tau} - \tilde{X}_{t+\tau})^2\right] \tag{4.1}$$

The problem of finding a function f of the given observations as a predictor for $\tilde{X}_{t+\tau}$ with best accuracy is trivial if we allow free choice of any function. Then, the best choice of f, in the sense that minimises MSE, is the conditional expectation of $X_{t+\tau}$ given X_v, $v < t$, which is denoted by $\tilde{X}_{t+\tau} = \text{E}[X_{t+\tau}|X_t, X_{t-1}, \ldots, X_{t-n}]$. However, this solution is of little use in practice since it is not possible to compute the conditional expectation unless we have a very detailed knowledge of the probabilistic structure of the process. We therefore consider very simple functions of the observations. The classical theory developed by Wiener and Kolmogorov deals only with linear functions [Pri81]. If the process is Gaussian, then there is no loss of generality because the conditional expectation of $X_{t+\tau}$ is a linear function of X_v, $v \leq t$; so the optimal predictor would be:

$$\tilde{X}_{t+\tau} = \sum_{v=0}^{\infty} a_v X_{t-v} \tag{4.2}$$

This mathematical problem was first solved by Kolmogorov and Wiener and can be regarded as filtering out the noise disturbance N_t so as to reveal the underlying process X_t.

More recently, Box and Jenkins [BR94] have developed a recursive algorithm to compute the predictors of discrete parameter processes that satisfy finite parameter linear models with a parsimonious number of parameters.

4.4 Nonlinear Dynamics Theory

The importance of nonlinear time series analysis techniques in the modelling of data communications traffic has been increasing since the discovery of self-similarity in Ethernet traffic traces [RK96a]. Today nonlinear analysis techniques are employed in several traffic modelling tasks. Rueda [RK96b] presents a survey of traffic characterisation techniques with applications of nonlinear time series methods. Willinger [WWE96], proposes a very comprehensive guide to self-similar traffic and performance modelling for modern high-speed networks.

We now introduce some fundamental concepts of nonlinear time series analysis like Lyapunov exponent, fractal dimension, delay embedding and attractor.

4.4.1 Overview

While time invariant linear systems can be described quite well with the methods presented in the previous section, non-linearity makes the application of those methods inappropriate.

Although a switching models methodology proposed by Tong [Ton90] can be constructed with a set of linear models, it contributes little to the understanding of how the system works. A possible approach to non-linear time series is non-linear dynamics theory, which encompasses nonlinear time series analysis.

The main theory underlying nonlinear time series analysis is the geometric theorem called embedding theorem by Takens and Mañé [TSC91][Tak81]. The theorem states that: Given an n-dimensional dynamical system with finite n, the internal dynamics of that system can be reconstructed from a time series with a single state variable $x_i(t)$, and it will be topologically identical to the true dynamics of the system.

If we can observe some scalar quantity of some vector function of the dynamical variables, then the geometric structure of the multivariate dynamics can be unfolded from this set of scalar measurements in a space made out of new vectors with components consisting of the scalar quantity shifted by some delay T. These vectors define a motion in a d-dimensional Euclidian space.

The importance of this theorem to time series analysis is that one can reverse engineer the full dynamics of a multi-dimensional system, by using the observed data of only one characteristic. For instance, one can predict the emitted power of a laser only by observing past data of the photonic emission.

The theorem does not provide ways to estimate the time lag T and what dimension d to use. We will present approaches for the calculation of d in Section 4.4.4, and for the calculation of m in Section 4.4.5. The next Section presents the delay coordinate embedding method.

4.4.2 Delay Coordinate Embedding

The most important phase space (or state space) reconstruction technique is the method of delays. Vectors in a new space, the embedding space, are formed from time delayed values of the scalar measurements:

$$\mathfrak{s}_j = \underbrace{\left[X_{n-(m-1)\tau}, X_{n-(m-2)\tau}, \ldots, X_n\right]}_{m\,\text{elements}} \tag{4.3}$$

The number m of elements is called the embedding dimension, the time is generally referred as the delay or lag. Embedding theorems by Takens [Tak81] and by Sauer et al. [TSC91] state that if the sequence $\{X_n\}$ consists of scalar measurements of the state of a dynamical system, then, under certain generic assumptions, the time delay embedding provides a one-to-one image of the original set x, provided m is large enough.

If N scalar measurements are available, the number of embedding vectors is only $N-(m-1)\tau$. Which increases the need for a reasonable choice of the delay so the structure we want to exploit should persist up to the largest possible length scale. Moreover, we always have to deal with a finite amount of noisy data. Both noise and finiteness will prevent us from having access to infinitesimal length scales. This issue gains special importance when analysing fractal dimension systems.

The embedding vectors obtained by this method, describe a motion trajectory in the m-dimensional space, arising the concept of *Attractor* that we present in the next section.

4.4.3 Attractors

Informally, an *attractor* is simply a state into which a system settles [Mei00]. Thus, in the long term, a dissipative dynamical system may settle into an attractor. Another definition of attractor is a set in the phase space that has a neighbourhood in which every point stays nearby and approaches the attractor as time goes to infinity. The neighbourhood of points that eventually approach the attractor is the *basin of attraction* for the attractor. See Figure 4.2 for motivational examples.

The boundary of a basin of attraction is often a very interesting object since it distinguishes between different types of motion. Typically a basin boundary is a saddle orbit, or such an orbit and its stable manifold. A crisis is the change in an attractor when its basin boundary is destroyed.

An alternative definition of attractor is sometimes used because there are systems which have sets that attract most but not all initial conditions in their neighbourhood (such a phenomenon is sometimes called riddling of the basin). Thus, Milnor defines an attractor as a set for which a positive measure of initial conditions in a neighbourhood is asymptotic to the set.

The definition of attractor is important to the understanding of the dynamics invariants described in the following paragraphs.

4.4.4 Mutual information

Kolmogorov and Sinai quantified, by means of an entropy, the rate of information generated by two orbits moving apart in the reconstructed phase space. Pesin related the rate of information generation with the rate of growth of distances by ergodic theorems [KS97]. Using the Kolmogorov Entropy $I(\tau)$ as a nonlinear correlation

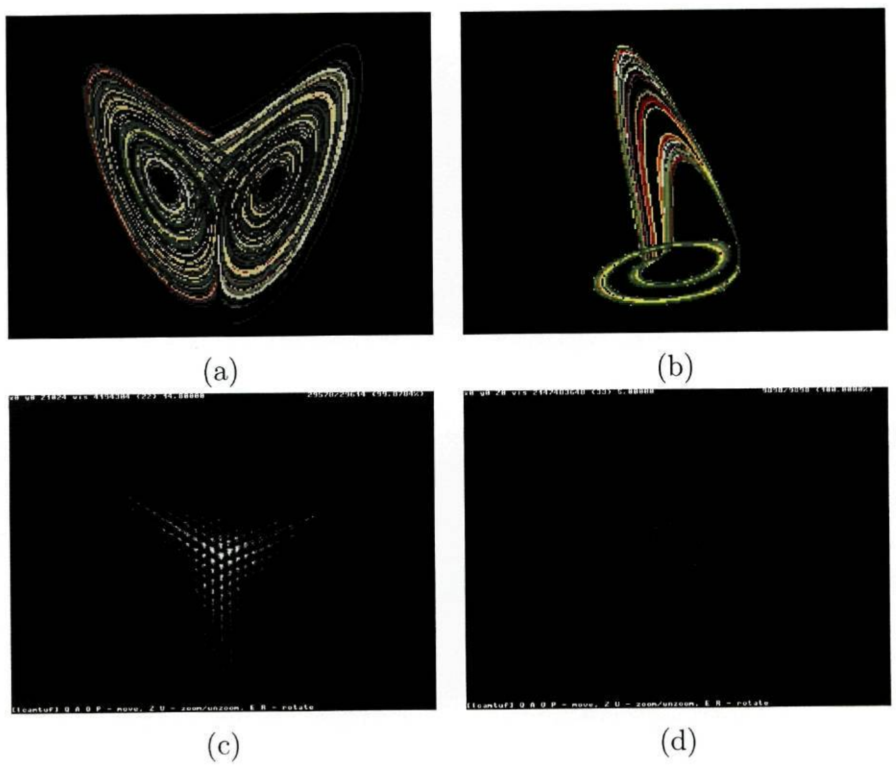


Figure 4.2: Plots of several attractors examples: (a) Lorenz Attractor - The "Butterfly Wings" problem in weather forecasting; (b) Rossler Attractor ; (c) Cisco IOS TCP/IP sequence number attractor - Attack vulnerability 20 %; (d) FreeBSD TCP/IP sequence number attractor - Attack vulnerability 0.0 %

function, one can determine when values of $\mathfrak{s}(n)$ and $\mathfrak{s}(n+\tau)$ are independent and useful as coordinates in time delay vector. This leads us to the definition of the *average mutual information* as a methodology for the choice of d in the delay embedding reconstruction.

The time delayed mutual information was suggested by Fraser and Swinney [FS86]. Unlike the autocorrelation function, the mutual information also takes into account nonlinear correlations. One has to compute

$$\mathrm{I}(\tau) = \sum_{ij} p_{ij} \ln \left(\frac{p_{ij}(\tau)}{p_i p_j} \right) \tag{4.4}$$

where for some partition on the real numbers p_i is the probability to find a time series value in the i-th interval, and $p_{ij}(\tau)$ is the joint probability that an observation falls into the i-th interval and at observation time τ falls into the j-th.

There are good arguments that if the time delayed mutual information exhibits a marked minimum at a certain value of τ, then this is a good candidate for a reasonable time delay, d.

However, these arguments have to be modified when the embedding dimension exceeds two.

4.4.5 False nearest neighbours

The *false nearest neighbour* method was first proposed by Kennel et al. [MBKA92] to determine the minimal sufficient embedding dimension m. The idea is quite intuitive. Suppose the minimal embedding dimension for a given time series X_i is m_0. This means that in a m_0-dimensional delay space, the reconstructed attractor is a one-to-one image of the attractor in the original phase space. More specifically, the topological properties are preserved. Thus the neighbours of a given point are mapped onto neighbours in the delay space. Due to the assumed smoothness of the dynamics, neighbourhoods of the points are mapped onto neighbourhoods again. Now, Suppose an m-dimensional space with $m < m_0$. Because of this projection, the topological structure is no longer preserved. Points are projected into neighbourhoods of other points to which they wouldn't belong in higher dimensions. These points are called false neighbours and are not typically mapped into the image of the neighbourhood, but somewhere else, so that the average "diameter" around the neighbourhood becomes larger. So, basically, by calculating the Euclidean distance among neighbour points $||\mathfrak{s}_i - \mathfrak{s}_j||$, and then calculating the radius

$$R_i = \frac{||\mathfrak{s}_{i+1} - \mathfrak{s}_{j+1}||}{||\mathfrak{s}_i - \mathfrak{s}_j||} \tag{4.5}$$

we can claim that this point is a *false nearest neighbour* if it exceeds a given heuristic threshold Rt [MBKA92].

A criterium to establish when a embedding dimension is high enough is the fraction of points for which R_i is zero, or at least sufficiently small.

The introduction of the false nearest neighbours concept and other *ad-hoc* instruments was partly a reaction to the finding that many results obtained for the genuine invariants, like the correlation dimension, has been spurious due to caveats of the estimation procedure. In the latter case, serial correlations and small sample fluctuations can easily be mistaken for nonlinear determinism.

4.4.6 Lyapunov Exponents and Fractal Dimension

The mathematical analysis of the dynamical invariants enables transformations of the data-like coordinate changes no matter how topologically the system is observed. The two most common invariants of a dynamical systems are: (i) the Lyapunov exponent λ, which measures how fast neighbouring trajectories diverge, and (ii) the fractal dimension d_c, which measures how much of $\mathbb{R}^n$ a trajectory occupies. The name fractal was coined because it can take a fractional dimension of $\mathbb{R}^n$.

4.4.6.1 Maximal Lyapunov Exponent

Let $\mathfrak{s}_i$ and $\mathfrak{s}_j$ be two points in the phase space with Euclidean distance $\delta_0 = \|\mathfrak{s}_i - \mathfrak{s}_j\|$. The distance at time n between the two trajectories emerging from these points is $\delta_{\Delta_n} = \|\mathfrak{s}_{i+\Delta_n} - \mathfrak{s}_{j+\Delta_n}\|$. Then the maximal Lyapunov exponent, λ is determined when:

$$\lambda = \lim_{\Delta_n \to \infty} \lim_{\delta_0 \to 0} \frac{1}{\Delta_n} \ln \left(\frac{\delta_{\Delta_n}}{\delta_0} \right) \tag{4.6}$$

A n-dimensional system has n λs, each measuring the expansion rate, in one direction (dimension), of the distance between two neighbouring trajectories. Positive λ implies exponential growth of the perturbation over that dimension, negative λ implies exponential convergence of the perturbation and a zero λ implies a less than exponential growth. The maximum Lyapunov exponent may be used to estimate the maximum time horizon of a prediction that will have error bounded to predefined value. For that, we may re-write eq (4.6) as $\delta_{\Delta_n} \leq \delta_0 e^{\lambda \Delta_n}$ in order to estimate the maximum time horizon Δ_n which keeps the distance between trajectories less than or equal to δ_{Δ_n}.

4.4.6.2 Fractal Dimension

The fractal dimension is defined as:

$$d_c = \lim_{\epsilon \to 0} \frac{\log\left(N(A, \epsilon)\right)}{\log\left(\frac{1}{\epsilon}\right)} \tag{4.7}$$

Where ϵ is an infinitesimal discretisation of the state space in a ϵ-box, let $\epsilon \to 0$, and count the resulting number of boxes. $N(A, \epsilon)$ is the number of closed balls of radius

$\epsilon > 0$ need to cover a set A. Fractal dimensions are flexible to characterise objects whose description is between dimensions, like the cantor set.

System attractors can be classified by dynamical invariants like the Lyapunov exponent and fractal dimension. Combinations of both invariants define three kinds of attractors:

1. A stable fixed point has n negative λs and fractal dimension d_c
2. A limit cycle has one zero λ and $n-1$ negative λs and fractal dimension d_c.
3. A chaotic attractor has one zero λ, at least one positive λ and generally non integer fractal dimension d_c

Inferences about a nonlinear system are generally less wide-range than linear methods. Nevertheless, linear methods are not applicable to nonlinear systems and the only analysis methods to choose are purely stochastic, which gives poor insight about the phenomena. The alternative is to centre the analysis in the target attractor type, basin geometry, dynamical invariants, etc.

4.5 Stationary Time Series Models

In this section the commonly used linear time series models are defined.

4.5.1 Autoregressive Moving Average Models

The fundamental assumption of time series modelling is that the value of the series at time t, X_t, depends only on its previous values (deterministic part) and on a random disturbance (stochastic part). Furthermore, if this dependence of X_t on the previous p values is assumed to be linear, we can write

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + \tilde{Z}_t \tag{4.8}$$

where $\phi_1, \phi_2, \ldots, \phi_p$ are real valued constants. $\tilde{Z}_t$ is the disturbance at time t, and it is usually modelled as a linear combination of zero-mean, uncorrelated random variables or a zero-mean white noise process denoted by $\{Z_t\}$

$$\tilde{Z}_t = Z_t + \theta_1 Z_{t-1} + \theta_2 Z_{t-2} + \ldots + \theta_q Z_{t-q} \tag{4.9}$$

The constants $\phi_1, \phi_2, \ldots, \phi_p$ and $\theta_1, \theta_2, \ldots, \theta_q$ are called autoregressive (AR) coefficients and moving average (MA) coefficients, respectively. The designations are justified by the resemblance of eq. (4.8) with a regression model and eq. (4.9) with a moving average (The average of a given time series during a time period q). Combining (4.8) and (4.9) we get

$$X_t - \phi_1 X_{t-1} - \phi_2 X_{t-2} - \ldots - \phi_p X_{t-p} = Z_t + \theta_1 Z_{t-1} + \theta_2 Z_{t-2} + \ldots + \theta_q X_{t-q} \quad (4.10)$$

This defines a zero-mean autoregressive moving average (ARMA) process of orders p and q, or ARMA(p,d). In general, a constant term may occur on the right-hand side of (4.10) meaning a nonzero mean process. A detailed description of this method can be found in [BR94]

4.6 Non-stationary and Seasonal Models

In this section we first introduce a special class of non-stationary ARMA method called the autoregressive integrated moving average (ARIMA). Then we define seasonal ARIMA (SARIMA) processes. We outline the key concepts of stationarity and state the conditions on the model parameters that ensure these properties.

4.6.1 Time series differencing

The trend of a time series $X(t)$ may be removed by differentiating the times series. The differencing operator may be defined as $\nabla_{X(t)} = X(t) - X(t-1)$ for all t. The differencing operator may be applied recursively over the previously differenced series, resulting in first order differencing, second order, ... , n order differencing. A first order differencing may remove a linear trend, a second order differencing removes the trend in the slope, and so on.

4.6.2 Stationarity and Invertibility

Applying a z transform to a model we obtain a function in the z-plane. The unit circle on the z plane characterise the behaviour of system regarding stability. A system with all zeros outside the unit circle is assumed to be stationary and invertible. Having zeros on the unit circle enables an ARMA model to be applied to non-stationary time series. The number of zeros in the unit circle d is equivalent to differencing the time series d times.

4.6.3 Autoregressive Integrated Moving Average - ARIMA

When the time series is not stationary, the methods of analysing stationary time series cannot be used directly. The time series should be differenced first before applying a stationary ARMA model. However, the stationary ARMA models introduced in Section 4.5.1 can be generalised to incorporate a special class of non-stationary time series models.

This class of non-stationary models is defined by

$$(1-B)^d\phi(B)X_t = \theta(B)Z_t \tag{4.11}$$

where d is a non-negative integer, $\phi(x)$ and $\theta(x)$ are polynomials of degrees p and q, respectively, B is the *backward shift operator* defined as $B^jX_t = X_{t-j}$ and the operation $(1-B)X_t = X_t - X_{t-1}$ is called *differencing the time series.*

Equation (4.11) defines an autoregressive moving average model of order (p, d, q), adapted from the ARMA model allowing the polynomial having d roots equal to unity, and the rest of the roots lying outside the unit circle. Or, in other words, if X_t is non-stationary, then after differencing the time series d times the differenced series is stationary and satisfies the restrictions of an ARMA process. Therefore, any ARIMA(p, d, q) series can be transformed into an ARMA(p, q) series by differencing d times.

The ARIMA model is used to model homogeneous non-stationary time series. For example time series that exhibit non-stationarity in level, or in level and in slope can be modelled as ARIMA$(p, 1, q)$ and ARIMA$(p, 2, q)$ respectively.

4.6.4 Seasonal ARIMA Process

Sometimes there can be seasonal or cyclic components in a time series. By this we mean the recurrence of some recognisable pattern after some regular interval that we call the seasonal period and denote by s. For example, in the monthly data of the atmospheric temperature there is a recurring pattern with a seasonal period of 12 months, which is clearly the period of the seasons.

A pure seasonal model is characterised by nonzero correlations only at lags that are multiples of the seasonal period s. This means that the time series at time t, X_t, depends on X_{t-s}, X_{t-2s}, X_{t-3s}, ...only. In general, we can define a pure seasonal ARMA model of orders P and Q and of seasonal period s by

$$X_t - \Phi_1 X_{t-s} - \Phi_2 X_{t-2s} - \ldots - \Phi_p X_{t-ps} = Z_t + \Theta_1 Z_{t-s} + \Theta_2 Z_{t-2s} + \ldots + \Theta_q X_{t-qs} \quad (4.12)$$

If we define the seasonal AR polynomial $\Phi(X^s)$ as

$$\Phi(X^s) = 1 - \Phi_1 X^s - \Phi_2 X^{2s} - \ldots - \Phi_p X^{ps} \quad (4.13)$$

and the seasonal MA polynomial $\Theta(X^s)$ as

$$\Theta(X^s) = 1 + \Theta_1 Z^s + \Theta_2 Z^{2s} + \ldots + \Theta_q X^{qs} \quad (4.14)$$

4.12 can be rendered more compactly using the backward shift operator as

$$\Phi(B^s) X_t = \Theta(B^s) Z_t \quad (4.15)$$

The pure seasonal models defined by (4.12) are often not very realistic since they are completely decoupled from each other. In reality, of course, few time series are purely seasonal and we need to take into account the interactions or correlations between the time series values within each period. This can be done by combining the seasonal and regular effects into a single model. A multiplicative seasonal ARMA model of seasonal period s and of seasonal orders P and Q and regular orders p and q is defined by

$$\phi(B)\Phi(B^s) X_t = \theta(B)\Theta(B^s) Z_t \quad (4.16)$$

Here $\phi(B)$ and $\theta(B)$ are regular AR and MA polynomials defined in (4.8) and (4.9).

To generalise the model defined by (4.16) to include non-stationary cases we define the seasonal difference to be $(1-B)^s X_t = X_t - X_{t-s}$. A multiplicative seasonal autoregressive integrated moving average (SARIMA) process of period s, with regular and seasonal AR orders p and P, regular and seasonal MA orders q and Q, and regular and seasonal differences d and D is defined by

$$(1-B)^d (1-B^s)^D \phi(B)\Phi(B^s) X_t = \theta(B)\Theta(B^s) Z_t \quad (4.17)$$

We will use SARIMA(p,d ,q)(P, D, Q) to refer to the model defined by (4.17). In typical applications $D = 1$, and P and Q are small.

4.6.5 Holt-Winters Models

The Holt-Winters method has proven through the years to be very useful in many forecasting situations. It was first suggested by C.C. Holt in 1957 [Hol57] and was meant to be used for non-seasonal time series showing no trend. He later offered a procedure (1958) that does handle trends. Winters(1960) [Win60] generalised the method to include seasonality, hence the name "Holt-Winters Method".

Holts initially modelled the series with one equation to the level a_t and one to the trend b_t. Winters added a third equation c_t to capture directly the seasonality. The implicit prediction model is

$$Y_t = a_t + b_t + c_t + \epsilon_t \tag{4.18}$$

Where a_t is the level of the trend, b_t the slope of the trend, c_t the correction term for seasonal variation, and ϵ_t a random term. For additive seasonality, a_t, b_t and c_t are estimated as:

$$\begin{cases} \hat{a}_t = \alpha(Y_t - \hat{c}_{t-s}) + (1-\alpha)(\hat{a}_{t-1} + \hat{b}_{t-1}) \\ \hat{b}_t = \beta(\hat{a}_t - \hat{a}_{t-1}) + (1-\beta)\hat{b}_{t-1} \\ \hat{c}_t = \gamma(Y_t - \hat{a}_t) + (1-\gamma)\hat{c}_{t-s} \end{cases} \tag{4.19}$$

For multiplicative seasonality each component a_t, b_t and c_t are estimated as:

$$\begin{cases} \hat{a}_t = \alpha(Y_t - \hat{c}_{t-s})(1-\alpha)(\hat{a}_{t-1} + \hat{b}_{t-1}) \\ \hat{b}_t = \beta(\hat{a}_t - \hat{a}_{t-1})(1-\beta)\hat{b}_{t-1} \\ \hat{c}_t = \gamma(Y_t - \hat{a}_t)(1-\gamma)\hat{c}_{t-s} \end{cases} \tag{4.20}$$

4.7 Non-Linear Time Series Models

In this section we introduce some simple nonlinear time series models. We present methods that fit locally or globally into the phase space. The local linear fits are very flexible, but can be ineffective on parts of the phase space where the points do not span the available space dimensions and where the inverse of the matrix involved in the solution of the minimisation does not exist. Global function results depend on how far the chosen *regression function* f_p is suited to model the unknown nonlinear function, and on how well the data are deterministic at all. However, they have the attractive feature of describing globally the dynamic of the system.

4.7.1 Simple Nonlinear Prediction

When the time series is expected to have more than a simple nonlinear component it is preferable to make local approximations in phase space because the autocorrelation of the system is more difficult to capture globally. Local approximations have the property of being robust. However, this approach has the drawback of short forecast lead times.

The simplest approach to local approximations is to keep only the zeroth order, that is, approximate the dynamic locally by a constant. In a delay embedding space all neighbours of $\mathfrak{s}_n$ are sought if we want to predict $\mathfrak{s}_{n+k}$. The forecast is simply the average over the future of the neighbours U_n:

$$\tilde{\mathfrak{s}}_{n+k} = \frac{1}{|U_n|} \sum_{\mathfrak{s}_j \in U_n} \mathfrak{s}_{j+k} \tag{4.21}$$

These methods require the delay coordinate embedding procedure to calculate $|U_n|$ neighbours and has input the minimal number of neighbours over which the predictions will be made.

4.7.2 Local Linear Prediction

If there is a good reason to assume that the relation $\mathfrak{s}_{n+1} = f(\mathfrak{s}_n)$ is fulfilled by the experimental data in good approximation for some unknown f and that f is smooth, predictions can be improved by fitting local linear models. They can be considered as the local Taylor expansion of the unknown f, and are easily determined by minimising the Sum of the Squared Errors (SSE)

$$\mathrm{SSE} = \sum_{\mathfrak{s}_j \in U_n} (\mathfrak{s}_{j+1} - a_n \mathfrak{s}_j - b_n)^2 \tag{4.22}$$

with respect to a_n and b_n, where U_n is the ϵ-neighbourhood of $\mathfrak{s}_n$, excluding $\mathfrak{s}_n$ itself, as before. Then, the prediction is $\tilde{\mathfrak{s}}_{n+1} = a_n \mathfrak{s}_n + b_n$. The minimisation problem can be solved through a set of coupled linear equations, a standard linear algebra problem. Locally linear approximation was introduced by Eckmann [JEC86] and by Farmer [FS87]. We should note that the straight least squares solution is not always optimal and a number of strategies are available to regularise the problem if the matrix becomes nearly singular and to remove the bias due to the errors in the "independent" variables. These strategies have in common that any possible improvement is bought with considerable complication of the procedure, requiring subtle parameter adjustments.

4.7.3 Global Function Fitting

The local linear fitting is very flexible, but can be ineffective on parts of the phase space where the points do not span the whole available phase space dimensions and where the inverse of the matrix involved in the solution of the minimisation does not exist. Therefore, many authors suggest the use of global nonlinear functions to fit the data. This approach requires the solution of:

$$\mathrm{SSE} = \sum_{\mathfrak{s}_j \in U_n} \left(\mathfrak{s}_{j+1} - f_p(\mathfrak{s}_j)\right)^2 \tag{4.23}$$

where f_p is now a nonlinear function in closed form with parameters p, with respect to which the minimisation is done. Polynomials, radial basis functions, neural nets, orthogonal polynomials, and many other approaches have been used for this purpose. The results depend on how far the chosen *function* f_p is suited to model the unknown nonlinear function, and on the degree of determinism of the data. We have adopted fitting radial basis functions [BL88, Smi92] and polynomials [Cas89] because in these two models the parameters p occur linearly in the function f_p and can thus be determined by simple linear algebra, and the solution is unique.

Although global models are useful to infer the structure and properties of the underlying system, they should be tested using an iterative procedure. The prediction errors, even if small in size, could be systematic and thus repel the iterated trajectory from the range where the original data is located. It can be useful to study the dependence of the size or the sign of the prediction errors on the position in the embedding space, since systematic errors can be reduced by a different model.

Global models are attractive because they yield closed expressions for the full dynamics. One must not forget, however, that these models describe the observed process only in regions of the space which have been visited by the data. Outside this area, the shape of the model depends exclusively on the chosen regression function. In particular, polynomials diverge outside the range of the data and hence can be unstable under iteration.

4.8 Time Series Analysis Methodologies

Time series analysis and forecasting requires the accomplishment of several tasks before successfully building a time series model. Some of those tasks obey to a precedence order. This may be further explored to establish a methodology for time series analysis. An instance of such a procedure is the *Box-Jenkins* methodology [BR94]. The Box-Jenkins methodology attempts to provide a versatile tool for time series analysis and forecasting so that one may obtain good results in practice. The methodology consists in 4 steps that may be iterated until an acceptable model for forecasting is found:

1. Tentative Identification;
2. Estimation;
3. Diagnostic Checking; and
4. Forecasting.

The Box-Jenkins methodology for time series analysis and forecasting usually employs ARIMA class of models. However, it is sufficiently general to be applied to other model classes with few enhancements required. Nevertheless, we here put in evidence some steps and enumerate some extensions to deal with more general cases:

- Selection of data transformations
- Selection of outlier handling procedures
- Selection of the forecasted variable
- Selection of independent variables
- Exceptional forecast handling procedures
- Model class identification
- Selection of parameter estimation method
- Model Selection
- Model Combination (or Model Averaging)
- Selection of ways to use human judgement

Although data transformation is intrinsically included in the *Tentative Identification* procedure of the Box-Jenkins methodology, it resumes to deal with non-stationarity. Some time series may include several seasonal components that should be identified before proceeding to model fit. We will present a brief explanation of the above steps in the next sections.

4.8.1 Selecting variables to be forecasted

Defining the variables to be forecasted usually is quite evident. However, in some cases there are no observations for the variable, and the variable has to be constructed. In this case it has to be decided what variables to forecast instead, in order to obtain information about the variable of interest. An example of a constructed variable is the *production index* of a country.

4.8.2 Selecting data transformations

Sometimes the variables must be transformed in order to remove seasonality or non-stationarity before they can be used for forecasting purposes. Some models cope with non-stationarity and non-seasonality. However, it is usually required to fully identify the non-stationarity order or seasonality period to parameterise such models adequately. Thus, whether or not data transformations are made, the procedures to identify such features are indispensable to check the conditions for applying forecasting models.

4.8.3 Selecting outlier handling procedures

Often the historical data contains exceptional values of variables. A central decision concerns how these should be detected and eliminated or otherwise handled. Some potential ways are replacing the values with smoothed ones [Wei90], or using robust estimators that don't place much weight on the outliers [Hub77].

4.8.4 Selecting independent variables

There are model classes where there are no explanatory variables. Such models are called extrapolation models (or time series models) in forecasting. They include classes such as ARIMA, SARIMA and Holt Winters models. However, external variables often improve dramatically forecasting performance, since they provide extra information about the deterministic features of the process. The problem is to decide which external variables are to be taken into account in forecasting.

4.8.5 Selecting model classes

An important decision is to select the model classes that are used when forming the forecasts. Often the attention is focused on only one or a few classes of models, but with the multitude of potential classes (ARIMA, transfer function, regression, exponential smoothing, various neural network models, etc.), each having its own strengths, weaknesses and applicability, which is very constraining. On the other hand, each new potential model class increases the amount of work required at execution time. There are several possible ways to determine the class of models more suitable to fit the data. Box and Jenkins [BR94], for example, advocate a pattern recognition approach based on inspecting the autocorrelation structure of the process. Another approach is to consider the problem as a search for an optimal model structure in the space of all possible model structures. However, exhaustive search through all possible model classes is computationally expensive, and usually not possible. Therefore, decisions have to be made on how to go through model classes in order to find the best according to a specified criterium used. Selecting this criterium is also a part of the time series analysis strategy.

4.8.6 Model Identification

A model structure is a set of models within a model class where the variables to be used have been selected, the equations connecting them together have been formulated, but where there is the possibility that some unknown parameters in the equations must be set [Lju87]. Many model classes, such as ARIMA and S-ARIMA, have an infinite number of possible model structures; other model classes, such as the Holt-Winters models, contain only a few. Selecting a suitable structure from a model class is called model identification [BR94].

4.8.7 Selecting parameter estimation method

Parameter estimation is the last stage in constructing a tentative model. There is a plethora of methods for parameter estimation for each model class, and what is best in a given situation depends on the data and usage context. Furthermore, parameter estimation methods implicitly or explicitly optimise a given criterium. If the optimisation is explicit, the criterium has to be supplied.

4.8.8 Selecting model validation criteria

Model validation is the essential criterion in deciding whether a model fits well data, and which of a given set of models is the best. However, it is usually not at all evident what criterion to use. When deciding whether a model is good enough, various criteria can be used; the most common ones relate to whether the residuals are a white noise process.

4.8.9 Model selection criteria

Choosing a model selection criterium is usually case dependent. For example, the least-squares error criterion is good when the data is relatively regular, but if the data contains many outliers, some linear criterion may be better.

There are also more sophisticated approaches to select a model. We can use, for instance, model selection criteria based on the Occam's razor principle. To cite some, we can select the Bayesian Information Criterium (BIC), The Akaike Information Criterium (AIC) and the Minimum Description Length (MDL) criterium. In which situation each one should be used depends on the type of data we have in hands.

4.8.10 Selecting exceptional forecast handling procedures

Sometimes the forecasts, despite being formed by well fitted models or renowned experts in the field, are out of range with expectations (also called catastrophes). Exceptional values should be detected, and, if possible, more reasonable estimates should be proposed [MLM93]. Decisions are needed on how to handle these: how to detect them (by simple rules, by some statistical tests, or by some other means), how to eliminate them (by using forecasts from alternative models, by filtering the forecasts afterwards, etc.) or whether to eliminate them at all and just warn the user.

4.8.11 Selecting models combination procedures

It is generally known [CH98] that combining forecasts from many models often yields more accurate forecasts than the forecast of a single model. Furthermore, some models might be used in cascade, the first model giving input to the second etc. Also multivariate models can be thought of as being composed of several models. Deciding whether, and in what way, to interlink models is a part of the time series analysis strategy.

4.8.12 Selecting ways to use human judgement

Defining whether and how to use human judgement is a difficult task. Although the accuracy of judgemental forecasts is, on average, inferior to statistical ones [MW89], it is often necessary to incorporate human judgement into forecasts; this is the case, for example, when forecasting the market performance of a new product. However, in many cases there is no advantage in using human expertise, or the costs and nuisance out weight the advantages. Deciding when forecasts or modelling would benefit from human judgement is a matter of judgement in itself. If it is to be used, it is to be decided how to avoid the pitfalls of human reasoning [MW89] and how to combine human and statistical forecasts.

4.9 Applying Time Series Analysis Methodologies

In this section we will apply the forecasting methods to a particular data set that describes the time evolution of a network link workload.

The importance of these experiments is to demonstrate the suitability of the described methods to forecast network load and to identify possible obstacles to the automation of time series analysis and forecasting.

We will explore several time series forecasting methodologies. The detailed description of the experimental setup, analysis techniques and model's parameterisation, can be found in [Alv02]. The models used in the experiments include: seasonal ARIMA, Holt-Winters and three nonlinear dynamics models: local linear regression; polynomial fitting and radial basis function fittings.

4.9.1 Experimental Settings

The time series under analysis consists in the workload of a link of the Abilene Data Communications Network [1]. The workload is defined as the transmitted information per second, expressed in *bit per second.* There are several granularities available for this data. Due to processing power and storage capacity issues, we have picked up the 2 hours aggregation period.

We have studied several intrinsic characteristics of this time series using linear and nonlinear time series analysis tools. We have calculated and plotted Autocorrelation Coefficients Function (ACF), Partial Autocorrelation Coefficients Function (PACF), the frequency spectrum, among other nonlinear time series analysis techniques.

We have observed in the ACF two major seasonalities: one with a period of 12 samples, which corresponds to a day cycle; and the other with a period of 84 samples, which corresponds to one week cycle. The sample ACF also suggested the presence of trend, more specifically a trend in the level of the time series. Based on this information, we may identify which classes of models are suited for this time series. We propose the usage of two stochastic models that cope with seasonality and non-stationary in time series: SARIMA and Holt-Winters. We have identified the model structure of each one and in Section 4.9.2 we present the forecasting results.

We applied nonlinear time series analysis techniques to estimate the embedding parameters. In a first approach, we have used approximately 1/3 of the total sample size and then all available data. In both studies, the embedding parameters maintained constancy. The delay of the embedding was 7 and the dimension was 11. The dimension of the embedding suggests the use of stochastic methods, since nonlinear dynamics methods are generally more suited for systems with a lower dimension. To look for evidence of nonlinearity in the time series we calculated the maximum Lyapunov exponent. The results were inconclusive due to a strong presence of "noise" in the time series that obfuscate the nonlinear signal component.

We have decided to apply nonlinear time-series analysis techniques even in the absence of solid evidence of nonlinearity in the time series because the sample size was not big

[1]Available from http://www.internet2.org

enough to reject this hypothesis. Supported with other results of applying nonlinear time series models to traffic time series [RK96b] [WWE96], we have completed our nonlinear analysis and fitted three models to the embedding. The results are now presented.

4.9.2 Results and Discussion

The results of the preliminary experiments are now presented and discussed. Table 4.1, presents the summary of results of fitting the models to the time series. To evaluate the models we measured on unseen data the Forecasting Mean Square Error (FMSE) and the Forecasting Mean Absolute Error (FMAE).

Figure 4.3 plots the predictions of the traffic workload. The fitted models were used to predict the time series with a forecasting horizon of 30 samples, which is equivalent to 2.5 days or 60 hours.

Forecasting Algorithm	FMSE (10^7 bps^2))	FMAE (10^3 bps)
SARIMA(0,1,1)x(84,0,1,1)	3,6	5,1
Local Linear Regression (k=90)	4,3	5,5
Polynomial Regression Fit (p=3)	6,4	6,1
Radial Basis Function (k=39)	6,5	6,9
Holt-Winters(T=84)	26,8	15,2

Table 4.1: Results on forecasting the link workload of the *Abilene* communications network with a time horizon of 60 hours. The link workload is measured in bits per second (bps)

The model with best performance is the SARIMA(0,1,1)x(0,1,1) parameterised with a seasonal period of one week and with regular and seasonal AR orders 0, with regular and seasonal MA orders 1 and with regular and seasonal D order 1. This can be regarded as a simple constatation of the rule of thumb that recommends applying nonlinear models only to low order dynamical systems.

The local linear regression model, fitted with 90 samples on the embedding, achieved a performance not very far from the SARIMA model. This model has a behaviour similar to a filter [MW89], removing the noise from the attractor orbits. This characteristic of the model helps revealing the deterministic dynamics of the system, which explains the FMSE of the model being only 18% worse than SARIMA.

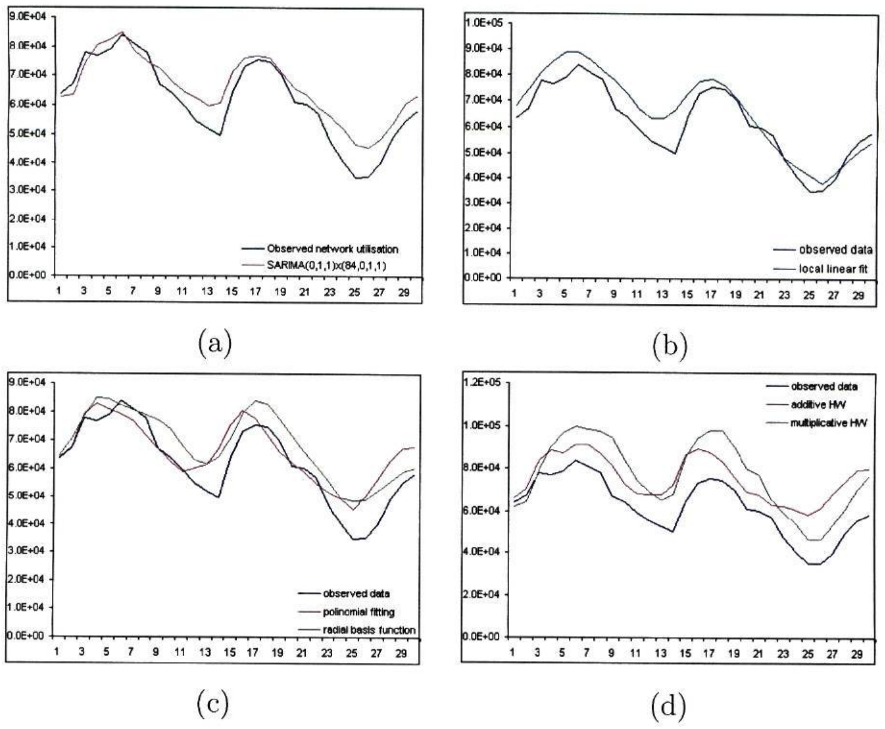


Figure 4.3: Plots of results on forecasting the link workload of a communications network with a time horizon of 60 hours: (a) SARIMA with seasonal period of 1 week; (b) Local linear fit; (c) Global function fits: Radial basis function and Polynomial Function; (d) Holt-Winters model with seasonal period of 1 week

Global function fits to the embedded time series were not able to capture so well the dynamics of the system. This method scored 47% worse than the local linear regression method and 75% worse than SARIMA FMSE. This can be explained by the fact that the dataset is too small and global function fitting describe the observed process only in regions of the space which have been visited by the data. Outside this area, the shape of the model depends exclusively on the chosen functional relationship, which seems not appropriate. An additional problem concerns the least squares is not always optimal, and can introduce some bias due to the errors in the "independent" variables. In particular, the polynomials tend to diverge outside the range of the data and hence can be unstable, which explains the poor fit to the data.

The Holt-Winters method was the last method applied. Holt-Winters methods are known to deal with seasonal data very well. However, in our experiment it achieved the worst performance because it has not been able to capture the trend in the time series.

4.10 Conclusions

As demonstrated in the first sections of this chapter, time series analysis and forecasting is a very broad research topic. To successfully forecast a time series one is faced with the requirement for a large amount of analysis and insight about the domain. Therefore, an autonomous-agent may find difficulties in collecting information to characterise the time series regarding some important characteristics like: seasonality, trend, nonlinearity, non-stationarity, uncertainty, etc. This information is essential to proceed with the selection of those classes of models that are best suited for fitting the time series. Those tasks require currently a considerable amount of human intervention analysing graphics and crossing the collected information with domain knowledge. Beyond this step, the tasks are more amenable to automation, since they consist essentially in comparing summary statistics, for instance: comparing confidence intervals of validation tests or selecting among models using some statistical criterion.

The preliminary experiments illustrated that it is possible to forecast the workload of a communications network, under some confidence interval. The explored methods were able to track, relatively close, the curve of observed workload of a network link during approximately 60 hours.

The work done for the preliminary experiments in analysing a specific time series and testing the reported methodologies was quite large. This led us to an important issue on this class of autonomous agents that need to make decisions pro-actively: *How to automate time series analysis and forecasting?* Next chapter describes our approach to deal with this issue.

Chapter 5

Automating Time Series Analysis and Forecasting

5.1 Introduction

Time series analysis and forecasting automation has been a research topic in Artificial Intelligence (AI) with the goal of reducing the human intervention and providing a tool for practitionaires and non-expert users. For example, Expert Systems (ES) have been used to automate time series forecasting in domains like: water supply [MB99], audit risk assessment [CS94], marketing planning [ECR90], electrical load forecasting [GVEDZ95] [HHC$^+$90] [MLM93], energy demand forecasting [CHL98] and product demand forecasting [HL94] [Lo94]. Other AI approaches include: Neural Networks [CS94, GVEDZ95, NLT99], Dempster-Shafer theory [ECR90], Fuzzy Logic [Hie94], probabilistic reasoning [HL94] and Case-Based Reasoning [NLT99]. These methods have been applied with different degrees of success and some of them compare better than models crafted by humans. For instance, Reilly [Rei03] claims that *autobox* outperform experts due to the exhaustive search-based approach, which considers alternative sequences to model augmentation.

Time series analysis requires decision-making based on theoretical and empirical results and also on human judgements. Expert Systems automate time series analysis and forecasting by dividing the task into a logical sequence of steps in which a Human may be prompted to make judgements based on its own knowledge. However, expert Systems have to rely on Machine Learning methods to estimate model parameters [Kar00], to select the amount of sample data for fitting models [Lan96], to forecast time series, to identify features in time series, and to select criteria to detect outliers [Kar00]. However, all the knowledge needed for time series analysis cannot be learned by forecasting systems in a sensible time. Hence, our proposal is to integrate that knowledge in the learning process, reducing the hypothesis space of the learning task. Inductive Logic Programming (ILP) is a Machine Learning paradigm that can naturally include domain knowledge in the learning process. A key aspect of ILP systems is the use of domain knowledge and training examples to induce models. Among the benefits of our approach is the possibility of inducing new knowledge, which results from relaxing model structures and letting the ILP system to "discover" the best arrangement for a given dataset. Another key aspect is the reduction of human judgement in time series forecasting by transforming it into decisions based on empirical results driven by systematic search.

The rest of this chapter is organised as follows: Section 5.2 presents related work. Section 5.3 gives a brief overview of the ILP framework. Section 5.4 presents our proposals to improve numerical reasoning in ILP systems. Section 5.5 introduces our approach to time series analysis and forecasting, using ILP to automate several steps of the methodology presented in the previous chapter. Section 5.6 evaluates our proposals and compares the experimental results with other statistical approaches reported in the literature. Finally, in Section 5.7 we draw the conclusions.

5.2 Related work

Several approaches to the task of automating time series analysis and forecasting may be found in the AI literature. Expert and Decision Support Systems have been used for selecting a model class [ECR90, Lo94, RJV96], to identify a model [BA97, MB99, HQ93] and to formulate the actual forecasts [GVEDZ95, Hie94, HHC$^+$90]. More specific applications include: handling of special events in data [HHC$^+$90]; automation of data manipulation [MB99]; plausibility checks of intentions data [MLM93]; judgemental estimation of external variables [CHL98]; evaluating forecasts [HQ93] and combining forecasts [FC92, BA97, HQ93].

ILP systems already have been used to induce knowledge for Expert Systems and to construct intelligible predictive models for data drawn from diverse domains. These include biochemistry (King *et al.* [KMLS92, KMSS96] and Muggleton *et al.*, [MKS92]), engineering (Feng, [Fen91], Dolšak and Muggleton, [DM92]), language processing (Zelle and Mooney, [ZM93], Cussens, [Cus97]), environment monitoring (Džeroski and Dehaspe, [DDRW94]), and software analysis (Bratko and Grobelnik, [BG93]). Numerical domains have received little attention in the ILP community, and the application of ILP to predict time series was not tackled until our work in [AAO03].

5.3 Inductive Logic Programming Framework

Inductive Logic Programming (ILP) is a Machine Learning methodology born from the intersection of Machine Learning, Logic Programming and Statistics research. The goal of ILP is the automatic induction of models from examples and domain knowledge. The hypothesis description language is First Order Logic (FOL). The motivation for using First Order Logic is twofold: use of a more expressive representational formalism than propositional logic, leading to an improvement in the intelligibility of the induced models; and facilitate the incorporation of domain (background) knowledge in learning process.

An ILP systems maps the hypothesis generation process into a search through an ordered space of hypotheses. The process of conjecturing *hypotheses* consists in combining different items of the background knowledge with the goal of explaining the provided examples. The set of hypotheses explaining the examples of a dataset is termed a *theory* and usually consists of a set of Horn clauses.

5.4 Improving Numerical Reasoning In ILP

Current ILP approaches [SC99] to numerical domains usually carry out a search through the model (hypothesis) space looking for a minimal value of a cost function like the Root Mean Square Error (RMSE). Systems like TILDE [BRR98] are of that kind. One problem with the minimisation of RMSE in noisy domains is that the models tend to be brittle. The error is small when covering a small number of examples. The end result is a large set of clauses to cover the complete set of examples. This is a drawback on the intelligibility of ILP induced models. This aspect is also an obstacle to the induction of a *numerical theory*, since we end up with small locally fitted *sub-models*, that may not correspond to the overall structure of the underlying process that generated data. Another drawback of most ILP systems, with the exception of Aleph [Sri04], is that the search procedure is usually an iterative greedy set-covering algorithm that finds the best clause on each iteration and removes the covered examples. This procedure may not find an optimal solution for the learning problem, thus leading to sub-optimal overall theories.

In this section, we propose improvements on the numerical reasoning capabilities of ILP systems. In particular we will address the problem of: noise handling; stopping criterium and theory-level search.

Our proposals for noise handling are based on a statistical approach to model validation; model selection; and model averaging. These techniques improve current ILP systems by checking the goodness-of-fit of the induced hypotheses, mitigating the over-fitting of hypotheses and model selection uncertainty.

The proposals made for theory level-search include a search framework and a pruning method. Theory-level search mitigates the problem of inducing sub-optimal theories in numeric domains by ILP systems that are based on a greedy-set covering algorithms.

The stopping criterium proposed is based on the PAC [Val84] method to evaluate the learning process. The stopping criterium derived is based on fundamental results established on the model validation step, allowing to abbreviate the hypothesis search based on a measure of performance.

5.4.1 Search Improvements

In ILP, each hypothesis generated during the search is evaluated to determine its quality. A widely used approach in classification tasks is to score a hypothesis by measuring its coverage, that is, the number of examples it explains. In numerical domains it is common to use the RMSE or Mean Absolute Error (MAE) as a score measure. Algorithm 1 presents an overview of the procedure and identifies the steps that were subject to modifications according to our proposals.

In this section we present two improvements at the clausal-level search. We propose an improvement to step 4 where a hypothesis is checked if it is a satisfactory approximation of the underlying process that generated the data. We propose the use of statistical tests in that model validation step. Our proposal for step 6 is inspired on the PAC [Val84] framework.

We use the terms hypothesis, model and theory with the following meaning in this chapter. A hypothesis is a conjecture after a specific observation and before any empirical evaluation have been performed (they are clauses in our framework). A model is a hypothesis that has at least limited validity for predicting new observations.

Algorithm 1: Basic cycle of a greedy set-covering ILP algorithm

```
1: repeat
2:   repeat
3:     synthesise a hypothesis
4:     accept a hypothesis (Model Validation)
5:     update best hypothesis (Model Selection)
6:   until Stopping Criterion satisfied
7:   Remove explained examples
8: until "All" examples explained
```

A model is a hypothesis that has passed the model validation tests. A Theory is a set of hypotheses that has been confirmed through empirical evaluation.

5.4.2 Model Validation

In most applications, the true nature of the model is unknown, therefore, it is of fundamental importance to assess the goodness-of-fit of each conjectured hypothesis. This is performed in a step of the induction process called *Model Validation*. Model Validation allows the system to check if the hypothesis is indeed a satisfactory model of the data. This step is common both in Machine Learning and Statistical Inference.

There are various ways of checking if a model is satisfactory. The most common approach is to examine the residuals, defined as follows.

Definition 1 (Hypothesis Residuals) *The residuals from an induced hypothesis h_j are the differences between the responses observed at each combination values of the explanatory variables and the corresponding prediction of the response computed using the induced hypothesis. Mathematically, the definition of the residual z_i for the i^{th} observation in the data set is written, $z_i = y_i - h_j(\overrightarrow{x_i})$, where y_i denotes the i^{th} response in the data set and $\overrightarrow{x_i}$ represents the list of explanatory variables at the corresponding values found in the i^{th} observation in the data set. Therefore, residuals are the random process formed from the differences between the observed and predicted values of a variable.*

As a consequence of the Wold's theorem [Lju87], the behaviour of the residuals may be used to check the adequacy of the fitted model.

Theorem 1 (Wold's Theorem) *Any real-valued stationary process may be decomposed into two different parts. The first is totally deterministic. The second totally stochastic. The stochastic part of the process may be written as a sequence of serially uncorrelated random variables z with zero mean and variance σ^2. The stationarity condition imply $\sigma < \infty$, thus z is a White Noise (WN) process denoted by:*

$$z \sim WN(0, \sigma) \tag{5.1}$$

According to condition (5.1) of the Wold's theorem, if the fitted model belongs to the set of "correct" functional classes, the residuals should behave like a white noise process with zero mean and constant variance.

Hypotheses whose residuals do not comply with condition (5.1) may be rejected using specific statistical tests that check randomness. The Ljung-Box test [Lju87], defined below, is one of such tests.

Definition 2 (Ljung-Box Test) *The Ljung-Box test is based on the autocorrelation function. However, instead of testing randomness at each distinct lag, it tests the "overall" randomness based on a number of lags. For this reason, it is often referred to as a "portmanteau" test. More formally, the Ljung-Box test can be defined as follows. The null hypothesis is the data is random. The test statistic is:*

$$Q_{LB} = n(n+2)\sum_{j=1}^{h}\frac{r(j)^2}{(n-j)}. \tag{5.2}$$

Where n is the sample size, $r(j)$ is the autocorrelation at lag j, and h is the number of lags being tested. The Significance Level is α. The hypothesis of randomness is rejected if:

$$Q_{LB} > \chi^2_{((1-\alpha);h)} \tag{5.3}$$

where $\chi^2_{((1-\alpha);h)}$ is the percent point function of the chi-square distribution.

The null hypothesis of the Ljung-Box test is a strict white noise process. Thus, residuals are independent and identically distributed (i.i.d.). According to the definition of statistical independence, namely condition (5.4), residuals are incompressible. Muggleton and Srinivasan [MSB92] have also proposed to check noise incompressibility for evaluating hypothesis significance but in the context of classification problems.

Other statistical tests may be incorporated to check our assumptions regarding error structure, like tests for normality. The use of residuals for model assessment is a very general method which applies to many situations.

Definition 3 (Statistical Independence) *Let $x_1, x_2, \ldots, x_i$ be a sequence of random variables. The variables x_i are statistically (mutually) independent if:*

$$\mathrm{E}\left[g_1(x_i)g_2(x_j)\right] - \mathrm{E}\left[g_1(x_i)\right]\mathrm{E}\left[g_2(x_j)\right] = 0 \quad \forall_{i \neq j}, \quad \forall_{g_1,g_2} \tag{5.4}$$

Where g_1, g_2 are any continuous function. Notice that this is much stronger condition than uncorrelation:

$$\mathrm{E}\left[X_i X_j\right] - \mathrm{E}\left[X_i\right]\mathrm{E}\left[X_j\right] = 0 \quad \forall_{i \neq j}. \tag{5.5}$$

Condition (5.4) means that there is no function that captures the relationship between these random variables.

The formulation presented in our approach considers that only noise is statistically independent, which is an assumption based on the Wold's theorem and verifiable by the Ljung-Box Test.

5.4.2.1 Other Relevant Statistical Tests

We have found two types of tests that are useful for the purposes established in this section: goodness-of-fit tests for measuring the compatibility of a random sample with a theoretical probability density function; and lack-of-fit tests for assessing the correctness of the functional part of a model.

Tests performed over the residuals of the fitted model can assure that no systematic error or process drifting exists. By verifying the autocorrelation function of the residuals one can infer if the residuals are random, or on the contrary, if they are correlated between them. Several tests can be computed for this purpose. The PACF [Que49] and ACF [Dan56] test over the bounds of the autocorrelation residuals can detect significant correlation spikes at medium and high lags. The BDS [WABL96] test can be applied to assure a non-linear independence of the residuals of a model driven by independent and identically distributed (i.i.d.) errors.

A statistic test that assures the functional part of the model is correctly specified and no significant terms are missing is the F-Test [SC89] as reported in [CR67]. The joint usage of those tests can prune the search space by rejecting models that fit poorly to data and therefore increase the chances of finding an hypothesis with the functional part correctly specified.

5.4.3 Stopping Criteria

A method to evaluate the learning performance is to define a bound δ on the probability of the learned hypothesis error being greater than ϵ. Thus, a stopping criterium would be when the probability of the error being greater than the accuracy ϵ is less than the confidence interval δ.

$$P(|z| > \epsilon) \leq \delta \tag{5.6}$$

This method for evaluating learning is called Probably Approximately Correct (PAC) [Val84].

Different degrees of "goodness" will correspond to different values of ϵ and δ parameterised in eq. (5.6). The smaller ϵ and δ are, the better the learned hypothesis will be. If we integrate the stopping criterium into a top-down breadth-first search strategy then, this algorithm finds the first clause with the smaller number of literals that satisfies the stopping criterium.

The calculation of $P(|z| > \epsilon)$ depends on residuals characteristics. To simplify the analysis we are assuming these random variables are indexed by a parameter, say $i \in I$, where I is called an index set. Therefore, our series of differences e_i constitutes a random process [Wei04]. We consider two processes:

1. Gaussian White Noise
2. Strict White Noise

White Noise residuals are an indicator that the hypothesis fits well to the data, that is, the residuals are uncorrelated. Gaussian White Noise is less general in practice because it implies testing residuals for normality besides uncorrelation.

5.4.3.1 Gaussian White Noise

In this approach, we set Model Validation step with a normality test besides the test for White-Noise process described in the previous section. A stationary Gaussian process is also strictly stationary and thus, the joint distribution of Gaussian process is also a Gaussian distribution. A Gaussian distribution may be fully characterised by the ensemble mean and variance [BD91]. Therefore, we may use the normal distribution function $\Phi(z)$ [Wei04] to calculate the probability of the error at data-point $|z_i|$ being higher than ϵ:

$$P(|z_i| > \epsilon) = 2 \times \left(0.5 - \Phi\left(\frac{\epsilon - \mu}{\sigma}\right)\right) \tag{5.7}$$

As we have pointed out $\mu = 0$ and therefore, equation (5.7) may be simplified removing the μ term. Equation (5.7) bounds the generalisation error because it is dependent only on the first and second order moments of the distribution, which are constant in a strictly stationary process.

5.4.3.2 Strict White Noise

In this section we propose to calculate the bound, δ, for any unknown distribution. First we address single-clausal theories and then, we extend the results to multi-clausal theories.

Theorem 2, proves the existence of the bound, δ, for an hypothesis and provides a procedure to calculate the error probability for a given accuracy level. Corollary 1, generalises the bound on the error probability to a complete theory. The proofs of both theorems rely on the convergence in probability mode, defined as follows.

Definition 4 (Convergence in probability) *Let $x_1, x_2, \ldots, x_n$ be a sequence of random variables. We say that x_n converges in probability to another random variable x, i.e. $x_n \xrightarrow{P} x$, if for any $\epsilon > 0$, $P(|x_n - x| > \epsilon) \to 0 \quad as \quad n \to \infty$*

Theorem 2 (Bounding Error Probability of an Hypothesis) *Let z be the residuals from the hypothesis h_i. Assume z is independent and identically distributed (i.i.d.) with distribution variance σ^2. Then the probability of the error being greater than ϵ is bounded by:*

$$P(|z| > \epsilon \mid h_i) < \delta, \quad \delta = \frac{\sigma^2}{\epsilon^2} \tag{5.8}$$

Proof 1 *Let the residuals $z_1, z_2, \ldots, z_n$ be a sequence of i.i.d. random variables each with finite μ and σ. if $\bar{z} = (z_1 + \ldots + z_n)/n$ is the average of $z_1, z_2, \ldots, z_n$, then, it follows from the week law of large numbers [BD91] that:*

$$\bar{z} \xrightarrow{P} \mu \tag{5.9}$$

Let the sample variance be

$$S_n = \frac{1}{n}\sum_{j=1}^{n}(z_j - \bar{z})^2 = \frac{1}{n}\sum_{j=1}^{n} z_j^2 - \bar{z}^2$$

, where $\bar{z}$ is the sample average. It follows from the Slutski's lemma [BD91] that:

$$S_n \xrightarrow{P} \sigma \tag{5.10}$$

Assuming the residuals z of the hypothesis h_i pass the null hypothesis of the Ljung-Box test, then they will comply with a strict white noise process with zero mean and finite variance, yielding thereby:

$$\mu = 0, \qquad \sigma < \infty \tag{5.11}$$

Following Conditions (5.9) and (5.10), each observation may be considered drawn from the same ensemble distribution. Thus, the sample mean and variance of the joint distribution converge to the ensemble mean and variance. Moreover, condition (5.11) states that both values are finite and, therefore, for all $\epsilon > 0$, the Chebishev's inequality bounds the probability of the residuals value, z, being greater then ϵ to:

$$P(|z| > \epsilon \mid h_i) < \frac{\sigma^2}{\epsilon^2} \tag{5.12}$$

We highlight the importance of the Ljung-Box test to assure i.i.d. of residuals. Only i.i.d. allows the application of the convergence axioms, which in turn allow the application of the Chebishev's inequality. The stationarity condition *per se* does not allow to consider that each observation was drawn from the same ensemble distribution, which prevents the direct application of Chebishev's inequality, and, therefore, to arrive at the derived conclusion presented here.

Corollary 1 (Bounding Error Probability of a Theory) *Let H be a set of hypothesis (clauses) induced for a given theory T. Assume:*

$$P(|z| > \epsilon \mid h_i) < \delta, \quad \forall_{h_i \in H} \tag{5.13}$$

then, for theory T, the probability of the error, z, being greater than ϵ, is bounded by:

$$P(|z| > \epsilon) < \delta \tag{5.14}$$

We recall that just one clause is learned at each time and the examples covered by that clause are removed, thus the same examples cannot be used to learn two clauses. When using a theory for prediction, only one clause of a non recursive theory predicts the output variable, that is, for a given instantiation of the input arguments of a predicate only one and the same clause is used for predicting the output argument. Thus all clauses of a non-recursive theory act as they were mutually exclusive regarding example coverage, i.e.

$$h_i \cap h_j = \emptyset \quad \forall_{i \neq j} \tag{5.15}$$

We also recall that the prior probability of h_i, $P(h_i)$ may be estimated calculating the frequency of h_i on the training set and dividing it by the coverage of the theory. Because the sum of the frequencies of all hypotheses is equal to the theory coverage, then

$$\sum_{\forall_{h_i \in H}} P(h_i) = 1 \tag{5.16}$$

Proof 2 *Let conditions (5.15) and (5.16) hold, then it follows from the* total probability theorem *that:*

$$P(|z| > \epsilon) = \sum_{\forall_{h_i} \in H} P(|z| > \epsilon \mid h_i) P(h_i). \tag{5.17}$$

Let condition (5.13) hold, then we may substitute $P(|z| > \epsilon \mid h_i)$ by δ in equation (5.17), yielding thereby:

$$P(|z| > \epsilon) < \delta \sum_{\forall_{h_i \in H}} P(h_i) \tag{5.18}$$

Since $\sum_{\forall_{h_i \in H}} P(h_i) = 1$ and because we assume $P(|z| > \epsilon \mid h_i) < \delta, \quad \forall_{h_i \in H}$, then:

$$P(|z| > \epsilon) < \delta \tag{5.19}$$

5.4.4 Model Selection

The evaluation of conjectured hypotheses is central to the search process in ILP. Given a set of hypotheses of the underlying process that generated data, we wish to select the one that best approximates the "true" process. The process of evaluating candidate hypotheses is termed *model selection.*

A simple approach to model selection is to select the hypothesis that gives the most accurate description of data. For example, select the hypothesis that minimises RMSE. However, model selection is disturbed by the presence of noise in data, leading to the problem of over fitting. Thus, an hypothesis with larger number of adjusted parameters has more flexibility not only to capture complex structures in data, but also to fit noise. Hence, any criterium for model selection should establish a trade-off between descriptive accuracy and hypothesis complexity. Figure 5.1 illustrates this trade-off.

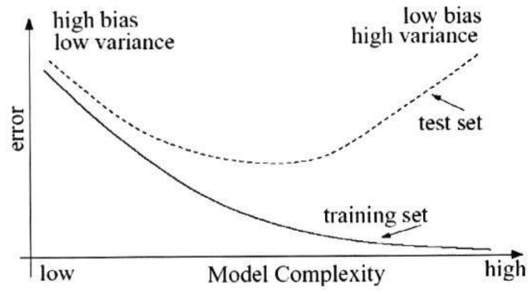


Figure 5.1: Typical behaviour of the test and train error of a learning algorithm as a function of the model complexity

We now present a statistical-based definition of hypothesis complexity for numerical domains.

5.4.4.1 Hypothesis Complexity

Defining a theoretically well-justified measure of model complexity is a central issue in model selection that is yet to be fully understood. In Machine Learning, some authors have proposed their own definition of complexity. Dzerovski [DT93], proposes a complexity measure based on the length of a grammar sentence in the Lagramge system. Muggleton [Mug96] proposes a complexity measure based on the number of bits necessary to encode an hypothesis.

Both of these complexity measures are sensitive to the hypothesis functional form. This is clear since both penalise each literal added. We argue that the functional form is not a good approximation to measure the complexity of a real-valued hypothesis, since any real-valued function can be accurately approximated using a single function class. This follows directly from Approximation Theory results. Two examples are the Weierstrass and Kolmogorov theorems, defined below.

Theorem 3 (Weierstrass theorem) *For any continuous function $f(x)$ and any positive ϵ, there exists a polynomial of degree m, $p_m(x)$, such that $|f(x) - p_m(x)| < \epsilon \quad \forall x$*

Theorem 4 (Kolmogorov superposition theorem) *Any continuous multidimensional function $f(x_1, \ldots, x_m)$ can be represented as the sum of $m + 1$ functions. These functions are called universal functions because they depend only on the dimensionality m and not on the functional form of f.*

Following theorem 4, the sum of universal functions is proportional to the dimensionality m. This highlights the role of dimensionality on a definition of hypothesis complexity. A few arguments on computational complexity and estimation theory also support this claim. Since the machine learning algorithm is given a finite dataset, models with fewer adjusted parameters will be easier to optimise as they will generically have fewer misleading local minima in the error surfaces associated with the estimation. They will be also less prone to the curse of dimensionality, defined as follows.

Definition 5 (Curse of Dimensionality) *For high-dimensional functions it becomes difficult to collect enough samples to attain high density. High-dimensional learning problems are more difficult in practice because low data density requires stronger, more accurate constraints on the problem solution.*

They will require less computational time to manipulate. A model with fewer degrees of freedom generically will be less able to fit statistical artifacts in small data sets and will therefore be less prone to the so-called "generalisation error". Finally, several authors (Akaike [BA02]; Efron [Efr86]; Ye [Ye98]) have proposed measures of model complexity which in general depend on the number of adjusted parameters. Consequentially, the adopted measure of complexity in this work is the number of adjusted parameters to data.

5.4.4.2 Model Selection Criteria

There are several model selection criteria suitable for the adopted measure of model complexity. Among these, we may find: (i) Akaike Information Criterium (AIC) [BA02];

Criterium		Definition
AIC	=	$-2\ln(L) + 2k$
AICC	=	$-2\ln(L) + 2k\frac{n}{n-k+1}$
BIC	=	$-\ln(L) + \ln(n)k$
MDL	=	$-\ln(L) + k + \ln(n)k$
Legend	n	number of samples
	k	number of parameters

Table 5.1: Model Selection Criteria

(ii) Akaike Information Criterium Corrected for small sample bias(AICC) [BA02]; (iii) Bayesian Information Criterium (BIC) [Sch78]; and (iv) the Minimum Description Length (MDL) [BA02].

Table 5.1 presents the definitions of the model selection criteria.

The estimation of an hypothesis likelihood function, L, with k adjusted parameters, requires a considerable computational effort and the assumption of prior distributions. In this context, the Gaussian distribution plays an important role in the characterisation of the noise, fundamentally due to the central limit theorem. Assuming error is i.i.d. drawn from a Gaussian distribution, then the likelihood of an hypothesis given the data [BA02] is:

$$\ln(L) = -\frac{n}{2}(1 + \ln(2\pi) + \ln(\hat{\sigma_r}^2)) \tag{5.20}$$

where $\hat{\sigma}_r^2 = \frac{1}{n}\sum_{i=1}^{n} z_i^2$, and z are the residuals.

Analytical model selection criteria like AIC and BIC are asymptotically equivalent to leave-one-out and leave-v-out cross-validation [Sha97]. However, they have the advantage of being incorporated in the cost function. Not requiring to split the dataset into groups or finishing the induction process before its application.

When these hypotheses have different coverages, Box and Jenkins [BR94](pg. 201) suggest the normalisation of those criteria by the sample size. This approach has the advantage of indirectly biasing the search to favour hypothesis with higher coverage, and consequentially, theories with less clauses.

Other authors presented similar work in this area. Zelezni [Zel01] derives a model selection criterium under similar assumptions that requires the Muggleton's complexity measure, which according to the adopted definition of complexity is unsuitable for our purposes. It also requires the calculation of the "generality" function for each induced hypothesis. His formulation does not estimate the modal value of the likelihood, so the final equation includes the usually unknown nuisance parameter of the hypothesis, which somehow limits its practical use.

5.4.4.3 Choosing a Model Selection Criterium

Model selection criteria have different characteristics, thus, it is essential to clarify its application conditions to numerical problems in ILP.

The use of AIC is recommendable if the data generating function is not in any of the candidate hypotheses and if the number of models of the same dimension does not grow very fast in dimension, then the average squared error of the selected model by AIC is asymptotically equivalent to the smallest possible one offered by the candidate models [Sha97]. Otherwise, AIC cannot be asymptotically optimal, increasing model complexity as more data is supplied [Sch78].

The use of BIC and other dimension consistent criteria like MDL is advisable if the correct models are among the candidate hypothesis, then the probability of selecting the true model by BIC approaches 1 as $n \to \infty$. Otherwise, BIC has a bias for choosing oversimplified models [Sha97].

Since except rare instances the true model is not known, or usually intractable, the use of AICC for prediction is then preferred.

5.4.4.4 Dealing with Model Selection Instability

The model selection process usually consists in choosing only one best description for the data discarding the remainder, even if there is evidence supporting all models. However, because there is noise in the data we cannot be certain that the ranking of models is correct. It would be possible to select another best model if another dataset was available, raising instability in model selection [Bre96]. In such case, a subset of models may be used for prediction instead of a single best model, reducing in this way variance in model selection.

Possible techniques for considering the prediction of a set of hypotheses are *bagging, boosting* and *model averaging*. In bagging, models are trained from bootstrap resamples of the data and weighted equally. This may prejudice in some cases both stable and unstable learning algorithms. In boosting, each additional model is chosen to (attempt to) repair the inadequacies of the current averaged model by resampling biased towards the mistakes. This method reduces both variance and bias at the cost of requiring a relatively large dataset. However, such framework is not applicable directly on time series problems. A possible alternative to both methods is *model averaging.*

5.4.4.5 Model Averaging

Model averaging consists in computing an estimate of the predicted value by weighting the predictions of candidate theories.

The issue of finding the weights to assign to each theory prediction is addressed by a modification in raw AIC values. First we compute the AIC differences [BA02] for all theories:

$$\Delta_i = \mathrm{AIC}_i - \min_i \mathrm{AIC} \tag{5.21}$$

We will select from the set of candidate theories those with substantial level of empirical support. Thus we have included only theories with $\Delta_i \leq 3$, as suggested in [BA02].

It follows that the relative likelihood of a theory h_i given the data is expressed by:

$$L(h_i \mid x) \propto \exp\left(-\frac{1}{2}\Delta_i\right) \tag{5.22}$$

if the relative likelihood is normalised by the sum of likelihoods we obtain the Akaike weights:

$$w_i = \frac{L(h_i \mid x)}{\sum_i L(h_i \mid x)} \tag{5.23}$$

The Akaike weight is considered the weight of evidence in favour of theory i being the actual best theory from the set. We may use the weights w_i to average predictions of each candidate theory. This method may also be applied to BIC values. For that we can convert the raw BIC values to "Schwarz" weights by replacing the AIC values in equation (5.21) by BIC values. Model averaging performs better than model selection and consistently reduces generalisation error. Section 5.6 presents an experiment that provides empirical support for that and compares model averaging and bagging in a time series prediction application.

5.4.4.6 Hypothesis Averaging and Theory Averaging

We have devised two techniques to reduce model selection uncertainty and bias: Hypothesis Averaging and Theory Averaging.

Hypothesis Averaging consists in selecting a set of hypotheses with substantial empirical support and averaging the predictions made by each hypothesis using Akaike weights.

Theory Averaging consists in producing several theories using a complete search strategy at *theory level* [Sri04], then averaging theories of the underlying process using Akaike weights.

We have made two experiments to assess the importance of the proposals. The experimental results show improvements on the model selection criteria (see section 5.6.9).

5.4.5 Search Strategy and Pruning at Theory-Level

Algorithm 1 (in Section 5.4.1) may lead to the induction of theories with very sub-optimal overall accuracy in numeric applications. This is even more critical in learning settings where a relatively large number of predicates are given as background knowledge and the ILP system is allowed to estimate several parameters at induction time. This may be attributed to two causes: (i) There could be an infinite number of models that pass on the ϵ neighbourhood of the observed examples in agnostic learning; (ii) In most ILP systems, the search procedure is usually an iterative greedy set-covering algorithm that finds the best single clause on each iteration. Thus, Algorithm 1 may not find the solution with best overall accuracy for the learning problem. Hence, we may need to rely on other search algorithms if one wants to assure that.

Srinivasan [Sri04] has proposed and implemented several Theory-Level Search methods based on randomised search techniques within the Aleph system. In fact, the Theory-Level Search (TLS) problem has an astonishing computational complexity, which usually prevents the application of a complete search strategy at theory-level. This directed us to an alternative approach based on three premises: (i) the final theory should have the prediction accuracy bounded by user-given parameters; (ii) the computational complexity of the search should be controlled by parameters that bound the search space; and (iii) The search strategy should make use whenever possible of justifiable pruning, without compromising condition (i).

This formulation has at least one important outcome. If the background knowledge provided to the ILP system does not allow the induction of at least one clause that comply with the accuracy level required by the user, the search is immediately stopped, allowing the user to reformulate the problem and to provide the ILP system with an alternative background knowledge definition.

An implication that follows directly from Corollary 1 is that condition (i) is possible to achieve if each clause of the theory also complies with the accuracy level required by the user. Second, because the upper limit on the error probability of a theory is comprised of a linear summation of terms and each term is always positive, then condition 5.17 may be considered a lookahead function since naturally one seeks for theories with the most stringent bounds on the error probability. Thus, we may implement a search strategy that allows the pruning of clauses based on the accuracy of the best theory found so far, while assuring a predefined level of accuracy for a theory.

In TLS, the search space may be represented as a tree, where each node corresponds to a clause, h_i. If one follows a complete search strategy, the search space would be huge since right at the first level of the tree, the number of clauses would be potentially all nodes of the lattice transversed by the greedy set-covering algorithm. Each clause would originate new sub-trees, in their turn, leading to an explosive combinatorial setup. In our approach only the first K clauses complying with the model validation tests and satisfying the stopping criterion are considered in each iteration of the set-covering algorithm. Moreover, since the user specifies the number of clauses to be retained at each iteration, the search is potentially even more abbreviated because only a partial sub-set of the lattice is explored on each iteration. IndLog uses a breadth-first search strategy to synthesise clauses. Thus, the partial sub-set explored consists of the upper levels of the lattice, where clauses are also more general.

Let K be the number of clauses specified by the user to be retained at each iteration of the set-covering algorithm. Then we may draw an upper bound on the computational complexity of the search.

5.4.5.1 Search Complexity

Let us study the complexity of the search in TLS, defined as a function of the number of set-covering algorithm iterations (reduction steps). Each clause induced by algorithm 1 is an iteration. Let n be the sample size and c be the minimum number of examples covered by an induced clause (c is usually configurable). Then, the maximum number of iterations, m, for covering the entire dataset would be:

$$m \leq \frac{n}{c} \tag{5.24}$$

Let K be the number of clauses retained in each iteration. Then, the maximum number of iterations, r, required for a complete search in TLS is bounded by:

$$r \leq \sum_{i=1}^{m} k^i \tag{5.25}$$

The maximum increase in the number of iterations is bounded by:

$$\lambda \leq \frac{\sum_{i=1}^{m} k^i}{m} \tag{5.26}$$

The overall maximum number of theories generated would be:

$$\tau \leq k^m \tag{5.27}$$

The above equations illustrate the importance of the K parameter in the definition of the TLS search space and consequentially in the control of the search space by the user. The number of iterations, is highly influenced by the parameter K, as well as the overall number of theories generated, τ. A popular approach to control the complexity in ILP systems is to limit the maximum number of nodes constructed. The number of nodes constructed depends on the lattice search procedure. Knowing the search type, the K parameter may be therefore translated to the number of nodes constructed using a simple algebraic relation, allowing to specify the search space using that popular parameter.

5.4.5.2 Pruning

The previous section developed mathematical relationships between the number of hypotheses generated and the computational complexity of TLS, which highlighted the importance of an efficient search framework, namely through the inclusion of pruning mechanisms wherever is possible. A pruning strategy would be to cut-off whole branches of the search space where it can be proved that the best theory lies else where. In this context, we propose to use branch-and-bound search algorithm to implement that pruning strategy. This algorithm would cut whole theories from the search space where it can be proven that the branches below a given clause h_m have provably a worst upper bound on the error probability than the best theory found so far, T^*. The proof that a whole branch may be excluded from the search if the lookahead function is condition (5.17) is given by theorem 5. Figure 5.2 presents an schematic representation of the theory-pruning method and the lookahead function. This is an intuitive lookahead function, since the goal is to search for theories with most stringent bounds on the error probability. The rules underlying the pruning method follows directly from condition (5.17) of corollary 1.

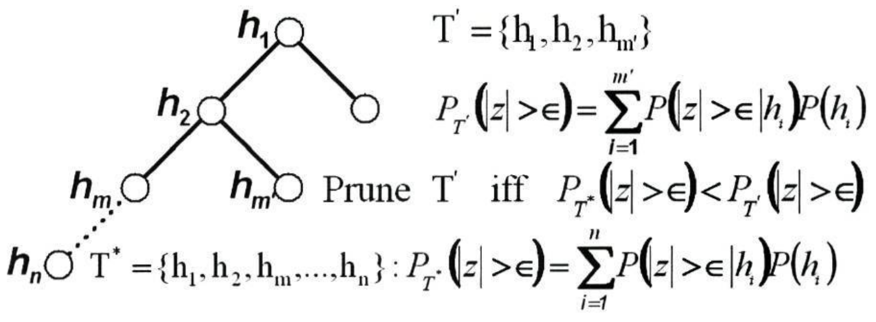

h_1
h_2
$h_{m'}$
h_m
h_n
$T' = \{h_1, h_2, h_{m'}\}$
$P_{T'}(|z| > \epsilon) = \sum_{i=1}^{m'} P(|z| > \epsilon | h_i) P(h_i)$
Prune T' iff $P_{T^*}(|z| > \epsilon) < P_{T'}(|z| > \epsilon)$
$T^* = \{h_1, h_2, h_m, \ldots, h_n\} : P_{T^*}(|z| > \epsilon) = \sum_{i=1}^{n} P(|z| > \epsilon | h_i) P(h_i)$

Figure 5.2: Pruning theories based on its descriptive accuracy

Theorem 5 (Pruning a branch) *Let $P^*(|z| > \epsilon)$ be the accuracy of the best theory found so far. Let $P'(|z| > \epsilon)$ be the accuracy of a partial theory T'. Let H_m be the set of clauses that forms theory T'. The best theory lies else where and the whole branch below theory T' may be excluded iff:*

$$P^*(|z| > \epsilon) < P'(|z| > \epsilon) \tag{5.28}$$

Proof 3 *The accuracy of the partial theory T' is given by:*

$$P'(|z| > \epsilon) = \sum_{\forall h_i \in H_m} P(|z| > \epsilon \mid h_i) P(h_i) \tag{5.29}$$

At best, the optimal error probability of any remaining clauses on subsequent branches would be:

$$P(|z| > \epsilon \mid h_i) = 0 \tag{5.30}$$

Thus, the optimal error probability of all remaining clauses of theory T' is:

$$\sum_{\forall h_i \in H} P(|z| > \epsilon \mid h_i) = 0 \tag{5.31}$$

If condition (5.28) holds, then condition (5.32) also holds due to the neutrality property of element zero. Which means that even if all remaining clauses would have zero error probability, the partial theory is already worse than the best theory found so far.

$$P^{*}(|z| > \epsilon) < P'(|z| > \epsilon) + 0 \tag{5.32}$$

5.4.6 Supporting Numerical Reasoning in ILP systems

Some ILP systems are implemented in Prolog, which is a language with a deficit in numerical reasoning capabilities, IndLog is one of this kind. We propose to overcome this Prolog drawback by providing an interface to a Statistical Computing System. That system provides a more powerful language for numeric processing - The R language [RP]. The R language is an interpreted language that runs in its own shell, therefore an interface between the Prolog interpreter and the R interpreter was built. The R language is released with many built-in numerical processing capabilities together with a large number of numerical analysis libraries. Its main strength is the easy inclusion of new packages. Many of the models referenced in the literature can be found in the packages of the R-Project. Furthermore, many of the mathematical functions and statistical algorithms that support decisions of a " time series analysis expert" are provided in the companion packages. This makes the R Language packages suitable to provide the necessary background knowledge definitions for time series domains and also to supply the numerical routines used in some steps of the ILP system.

We have also developed a library of numerical procedures in the C programming language which supplies extra algorithms, some of them not found in the R Language, which can be called from the Prolog interpreter.

This supporting software infrastructure is provided as a Prolog dynamic library, making the numeric algorithms accessible by the background knowledge of the ILP system.

5.5 Automating Time Series Analysis and Forecasting using an ILP System

The methodology for time series analysis and forecasting presented in section 4.8 is now addressed and techniques for its automation proposed.

We highlight the importance of two issues that may be easily tackled by an ILP system: the combination of judgement and quantitative methods, and the use of domain knowledge in time series forecasting.

The combination of judgement and quantitative methods in time series forecasting was first proposed by Makridakis and Hibon [MH79]. After this, a survey by Armstrong [FC88] identified the need to integrate judgement and extrapolation. Bunn and Wright [BW91], in literature review, concluded that judgement and statistical forecasting methods should be integrated . Furthermore, Lawrence, Edmundson and O'Connor [LC86] and Sanders and Ritzman [SR0b] have provided empirical support for that claim.

The use of domain knowledge in time series forecasting has been suggested to increase reliability in forecasting [RWL01]. Domain knowledge can be easily incorporated in any ILP system capable of numeric reasoning.

These tasks are naturally addressed by an ILP system. The next sub-sections presents our approach to the automation of a methodology for time series analysis and forecasting described in chapter 4.

5.5.1 Model identification

We provide two alternatives to implement Model Identification in ILP: (i) The use of statistical methods for model identification that are provided by the interface to the Statistical Computing System and (ii) the execution of a systematic search for model parameterisation. The first approach is usually less computationally intensive.

5.5.2 Selecting variables to be forecasted

The system must be informed of which variable is to be forecasted. Hence, the user must select and specifically declare in the configuration of the ILP system which variable will be predicted.

5.5.3 Selecting data transformations

The identification and processing of data transformations are possible in the ILP system by chaining lazy-evaluated predicates that perform data transformations. The result of each data-transformation-predicate feeds the input of the next one, in a pipeline fashion. This approach allows the successive execution of several data transformation operations to the time series.

5.5.4 Selecting model classes

We propose an hybrid approach to select which class of models to fit. The classes of models which are presumably suited for the data must be selected and activated. Then, the ILP system makes a systematic search in the models space for the best description of the sample data. Our system is provided with an interface to a large repository of time series models, which facilitates the human intervention in this process.

5.5.5 Selecting model structures

We propose an hybrid approach to select which model structures to use. The user must select and activate which structures are presumably suitable for the data. Then, the ILP system makes a systematic search in the models space for the best description of the sample data. This facilitates the discovery of new structures as illustrated in experiments (see Section 5.6.7). We provide mechanisms for reducing the search space, namely through the use of grammars.

5.5.6 Selecting independent variables

We propose to select the most appropriate variables automatically following a generate and test approach. This task is particularly important due to the requirement for an exponentially increase in sample size for a linear increase in the number of predictor variables in order to densely populate higher dimensional spaces. The choice of which variables should be used is made empirically by systematically searching the space of variables in the model. To cope with the dimensionality problem the model selection procedures use a penalty for the model complexity that indirectly bias the selection towards lower dimensional models. This facilitates the choice of the correct (asymptotically) model dimensions and the appropriate independent variable as illustrated in the experiments (see Section 5.6.6).

5.5.7 Selecting parameter estimation methods

The selection of Parameter estimation methods usually follow a trade-off between estimation accuracy and computational complexity. Some statistical methods like Burg AR Estimator [BD91] attempt to provide this trade-off, and others like Maximum Likelihood Estimator (MLE) are very demanding computationally. The preference of which methods to use is left, in last analysis, to human judgement. Nevertheless, we usually prefer estimators that make a trade-off between accuracy and complexity.

5.5.8 Selecting model validation criteria

Model validation may be performed by tests of lack-of-fit or goodness-of-fit. One of the most general ways to verify the degree of fitness to data of an hypothesis is to check if residuals are a white noise process. When fitting nonlinear models the BDS [WABL96] test checks for a non-linear dependence in the residual series. Both tests are provided in the background knowledge to be used in other experiments. We have also provided tests that check the lack-of-fitness of specific models. However, the usage of statistical tests are subject to significance level. We rely on human judgement to arbitrate such settings.

5.5.9 Selecting outlier handling procedures

The procedures for handling outliers consist, first of all, in selecting a statistical test for outliers detection. An example is the Grubbs's test [Gru69]. Such test requires the assumption of a given distribution that must be selected using human judgement. The significance level of the statistical test should be arbitrated using human judgement. The handling procedure after detecting outliers may be to remove the sample or to replace it by smoothing the sample region in the sample space. This procedure may be processed in the Statistical Computing System for which we have provided an interface.

5.5.10 Selecting exceptional forecast handling procedures

Exceptional handling forecasts procedures are domain specific. The procedure for handling exceptional forecasts may be coded in the background knowledge and is activated during the forecast to correct the exceptions.

5.5.11 Selecting model combination procedures

Combining forecasts from many models often yields more accurate forecasts than using the forecasts of a single model [CH98]. Model combination is fully provided using a generic statistical approach reported in Section 5.4.4.5.

5.5.12 Selecting ways to use human judgement

We propose the integration of human judgement through the incorporation in the background knowledge of rules that may represent judgement decisions on the sample data, used models, parameter values, forecast accuracy, and model structures. These features are supported through the usage of grammars, background knowledge and system settings.

5.6 Experiments

In this section, we report on the empirical evaluation of our proposals. We compare the results with statistical methodologies reported on the time series forecasting literature. We have selected non-linear time series datasets, since they usually present a greater challange to prediction. The datasets under study were collected from a broad range of domains, including engineering, social and life sciences. The predominant estrategy used was to compare the results obtained with an ILP system that was modified with our proposals and the same ILP System without the proposals activated.

5.6.1 Overview

The experiments are devided in two parts. The first part consists in a toy experiment which evaluates the model validation and model selection proposals. The second part consists of experiments 2, 3 and 4, which basically induces a model for predicting the future value of a time series. In these later experiments, the ILP system was provided with same background knowledge and datasets, with the purpose of evaluating the impact on forecasting performance of the proposals made. These later experiments illustrate the usage of an ILP system on scientific discovery tasks. In that sense, these experiments have been inspired in Colton and Muggleton [CM03] application of an ILP system to mathematical discovery. Although these experiments reports on learning multiple clause theories, they are related with Zelezni's [Zel01] work since they also learn a numeric function.

5.6.1.1 Summary of experiments

Experiment 1 consists of a functional class separation in the absence of noise. The goal is to evaluate the model validation and model selection techniques in optimal conditions, in order to evaluate the proposals before advancing for real data experiments.

Experiment 2 Consists of a time series forecasting problem. The goal is to evaluate the model validation and model selection techniques in real data.

Experiment 3 Consists of a time series forecasting problem. The goal is to evaluate the Theory Level Search techniques in real data.

Experiment 4 Consists of a time series forecasting problem. The goal is to evaluate the Theory and Hypotheses averaging techniques in real data.

5.6.2 The Learning Task

We propose to learn a class of non-linear time series models related with the Threshold Auto-Regressive (TAR) model [Ton90].

The TAR model is a nonlinear time-series process composed of linear sub-models. Each amplitude switched AR process is constructed for a specific amplitude range or subregion. The AR model to be used at time n is determined by the amplitude $x(n-D)$ where D denotes a time-delay. The AR model for sub-region m is activated if the following constraint is true: $R_m < x(n-D) < R_{m+1}$ The variable x is the time-series observed, m is the index denoting the sub-region, R_m denotes the threshold amplitude in the region m.

The goal of these experiments is to discover a new time series model structure. In a regular TAR model, the value for the time delay, D, is constant for all subregions m. In these experiments, we program the ILP system to discover whether allowing different D values for each sub-region results in improvements in forecasting performance on the test set.

5.6.3 Datasets

The following datasets were used in the experiments:

- Artificial dataset
- Canada's Industrial Production Index
- USA Unemployment rate
- ECG of a patient with sleep apnea
- VBR Traffic of an MPEG video

The artificial dataset was generated using two equations described in Section 5.6.6.

The Canada's Industrial Production Index was used in experiments reported in [SvD03]. We have used the logarithm of the first-differences from the period of 1960 until the year 2000. The train and test sample have of 430 and 49 examples respectively.

The USA Unemployment Rate was used in experiments reported in [lLMT98]. We have used the first-differences of the Quarterly Unemployment Rate from the period of 1939 until the year 2000. The train and test sample have 165 and 19 examples respectively.

The ECG of a patient with sleep apnea was used in experiments reported in [KRS$^+$99]. We have used heart rate sub-sampled in 1/4 ratio from the periods between 0 and 1500s. The train and test sample have 350 and 350 examples respectively.

The VBR Traffic of an MPEG video was used in experiments reported in [JT98]. We have used the bit count for I frames 31000 to 32100 of the full motion video "James Bond". The train and test sample have 1000 and 100 examples respectively.

5.6.4 Benchmark models

We have used the ILP system IndLog [Cam00] to induce switching models for non-linear time series. We have compared the results with structural, seasonal, linear and nonlinear time series models.

In this set of experiments the theories induced by the IndLog system (ILP), is compared with the following models: Auto-Regressive Integrated Moving Average (ARIMA); Threshold Auto-Regressive (TAR); Markov Switching Autoregressive (MSA); Autoregressive model with multiple structural Changes (MSC); Self-Excited Threshold Auto-Regressive (SETAR); Markov Switching regime dependent Intercepts Autoregressive parameters and (H)variances(MSIAH); Markov Switching regime dependent Means and (H)variances (MSMH); Bivariate Auto-Regressive models (Bivariate AR); and Radial Basis Functions Networks with structure optimization (RBFN). These models are used for comparison with the results discussed in the papers presented in section 5.6.3.

In all experiments the statistics used to compare the models is the Forecast Root Mean Square Error (FRMSE). All forecasts use lead time of one sample period.

5.6.5 Background Knowledge and General Settings

This section describes the general settings of all experiments except for the toy experiment presented in section 5.6.6.

We have provided IndLog with 12 AR(p) models, p ranging from 1 to 12; three model archetypes consisting of two, three and four amplitude intervals with 12 delay choices, D ranging from 1 to 12. The interval amplitude is calculated dividing the peak-to-peak value of each series by the number of intervals used in the archetype.

5.6.6 Startup experiment - An artificial dataset

To illustrate the impact of the proposals made for model selection and model validation we a simple experiment. The goal of the experiment was to confront the ILP system with a functional relationship separation problem using only one independent variable.

In the experiment, four parameters were varied: (i) Sample Size - Recall Number relationship of the ILP system, (ii) usage of the model validation step in the hypothesis search, (iii) significance level for the errors and deviations, and (iv) the usage of a model selection procedure.

The model validation tests selected were a Ljung-Box-Pierce test of the residuals correlation, the F-Test for lack of fit and the determination coefficient test for an informative output.

The Sample Size - Recall Number relationship is derived for usage in dataset separation in Section 5.6.6.1.

The equations modelled in the experiment are described in section 5.6.6.2.

The experiment uses the multi-linear regression and the equation-fitting algorithm, so the results will be expressed in the form of equations. A grammar of valid models was defined and provided scale type information to avoid nonsensical theories.

The following equations were used to generate the dataset:

$$y = \begin{cases} 0.5t + 0w + 0m & t \geq 6.0 \\ t^2 - 6t + 0w + 0m + 6 & t < 6.0 \end{cases} \tag{5.33}$$

Where t is time measured in seconds since 1970, w is the week day represented by the natural numbers between 1 and 7, m is the minute of the day represented as integer numbers between 0 and 1440.

5.6.6.1 Recall Number

The recall number is a limit on the number of possibilities of a non-deterministic predicate.

This is a guiding rule to estimate the sample size versus the data set size. (predicates that do some kind of sampling in the train dataset, during the saturation phase of the induction process):

$$r = \frac{|F|}{S} + \sigma \tag{5.34}$$

$|F|$ is the data set size, S is the chosen sample size and σ is the standard deviation of the uniform distribution given by the formulae:

$$\sigma = \sqrt{\frac{F}{12}} \tag{5.35}$$

Given that the samples are not deterministically partitioned but drawn from a uniform distribution pattern, we add the standard deviation to compensate some drifts around the deterministically calculated intervals.

Alternatively we can use the number of different examples of the dependent or independent variable. This would be a better guiding rule since it would potentially lead to a minor number of recalls to be used, and therefore speeding-up the induction process.

5.6.6.2 Results

The equations [1]. induced by IndLog are presented in table 5.2.

Step	Model
1	$y = \begin{cases} t + 2.075\sin(0.490t + 3.590) & t < 1.5 \\ 0 & t \geq 1.5 \end{cases}$
2	$y = \begin{cases} t^2 - 6t + 6 & t < 5.75 \\ 0 & t \geq 5.75 \end{cases}$
3	$y = \begin{cases} 0.5t + 3 & t \geq 8.5 \\ -0.533w - 3.887m - 0.002t + 27,684 & 4 \leq w \leq 7 \\ 2.805t - 12.561 & t < 8.5 \end{cases}$
4	$y = \begin{cases} 0.5t + 0w + 0m & t \geq 6.0 \\ t^2 - 6t + 0w + 0m + 6 & t < 6.0 \end{cases}$

Table 5.2: Models induced

The results from the experiment were compiled in table 5.3.

The inclusion of the model validation procedure during the hypothesis search is represented as "Model Validation". The confidence interval is represented as "Confidence Interval". The usage of the recall number formulae is represented as "parameterisation". The usage of the AIC criteria was described as "Model Selection". The error

[1]Models are represented as systems of equations to improve clarity of the text. The output of the ILP system is composed by first-order horn-clause theories

Step ID	1.0	2.0	3.0	4.0
Model Validation	NO	YES	YES	YES
Confidence Interval	99.9%	99.9%	95.0%	95.0%
Parameterisation	NO	NO	NO	YES
Model Selection	NO	NO	YES	YES
Train Results				
RMSE	0.95	~ 0	0.550	8.09x10-14
MAE	0.762	~ 0	0.328	3.98x10-14
Coverage	91%	34%	100%	100%
PE	1.13	~ 0	~ 0	~ 0
Test Results				
RMSE	2.749	0	0	0
MAE	2.631	0	0	0
PE	3.91%	0%	0%	0%
Coverage	100%	0%	100%	100%

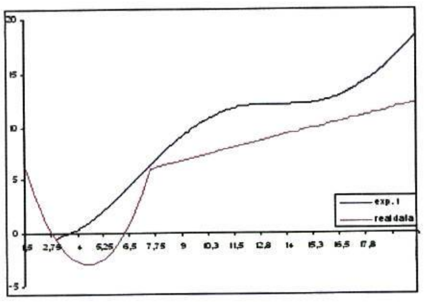
Table 5.3: Summary of results

statistics are the Mean Square Error represented as "MSE" and the Mean Absolute Error, represented as "MAE". Coverage is represented as percentage of the total data. The percentage error is represented as "PE".

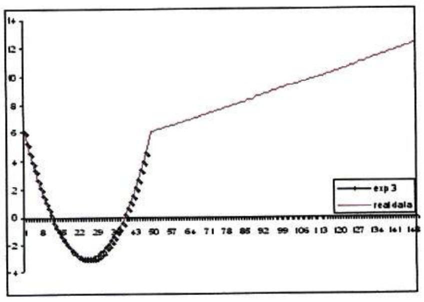
5.6.6.3 Results Discussion

A plot of the induced models is presented in figure 5.3. Figure 5.3 (a) shows a model with the functional part incorrectly estimated. Figure 5.3 (b) shows the effect of an incorrect parameterisation of the recall number and sample size in the ILP system. Figure 5.3 (c) shows the effect of correctly selecting the functional part of the model, to follow the principle of minimum description length and to parameterise correctly the ILP system.

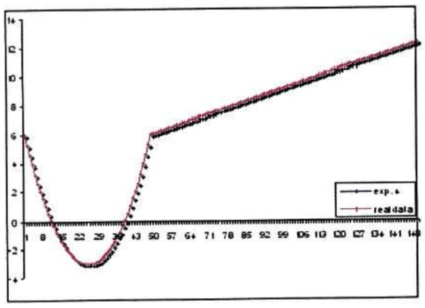
The step 2 was not plotted because the induced model has four dimensions. It can be analysed recurring to Equation 2. It shows a negative example of the Occam's razor principle. The model is too complex and the information retrieved from analysing the equations is not demonstrative of the process behaviour.



(a)



(b)



(c)

Figure 5.3: Plots of results on forecasting the link workload of a communications network: (a) Step 1 - No model validation, selection and evaluation; (b) Step 3 - Bad parameterisation of the ILP system; and (c) Step 4 - All features enabled and ILP system configured using recall number formulae

5.6.7 Assessing Model Validation and Model Selection

5.6.7.1 Overview and Goal

This experiment aims at the collection of empirical evidence for the proposals made on model validation and model selection as well as to illustrate the potential of ILP systems to deal with numerical problems.

5.6.7.2 Procedure

The experiment consists in two steps. The first step consists of running the ILP system IndLog without any of the proposals mentioned in this dissertation activated. In the second step, the ILP system is ran over the datasets with the model validation and model selection procedures activated. The results are collected and presented in table 5.4.

5.6.7.3 Experimental Settings

We have configured the model validation step with Ljung-Box test with a *p-value* set to 0.75. The model selection criterium used was the AICC. The presented values are the average RMSE of the test set, using the nonoverlapping block bootstrap with block lenght approximately of one fifth of the sample size.

5.6.7.4 Results and Discussion

This section presents the results obtained for each dataset described in Section 5.6.3. Those datasets were studied in several papers. Each paper used a different set of classes of models therefore the classes of models used in the comparison are different. Nevertheless, in Table 5.4 all datasets have an AR model that may be used as a reference when comparing the results across datasets.

The observed recall number on the test set for the Unemployment, Production, VBR Traffic, and ECG datasets is respectively: 100%, 96%, 94%, and 78%.

The ILP system consistently induced models with best forecasting performance in the test sets of all datasets presented. This allows us to conclude that the proposed modifications to the ILP search mechanism makes an ILP system adequate for learning time-series models and for discovering new model structures.

In all datasets, the ILP system was configured to allow the induction of the TAR model proposed in the cited papers. However, all models induced by the ILP system, used different D values in the constraint that selects each sub-region. This is perhaps an indicator of the superior flexibility of this structure to capture implicit seasonality and/or complex switching behavior in those time series.

A limitation of current ILP systems is that it is not possible to assure that an induced theory will have a total example coverage, which may lead to theories that do not

Model	Unemployment	Production	VBR Traffic	ECG
ILP*	0.91	0.85	0.93	0.82
ILP	1.11	1.04	0.96	0.93
AR	1.04	0.98	0.94	0.97
SARIMA	1.00	1.00	-	-
MSA	1.19	-	-	-
MSC	-	1.00	-	-
MSMH	-	0.98	-	-
MSIAH	-	1.20	-	-
SETAR	-	1.19	-	-
TAR	1.00	-	1.00	-
RBFN	-	-	-	1.00
Bivariate AR	1.20	-	-	-
Benchmark RMSE	1.59E-1	4.44E-3	12.93E3	4.53

Table 5.4: Summary of results of the Relative FRMSE of the ILP algorithm and other benchmark models for the selected datasets. ILP^* stands for the ILP system with our proposals activated and ILP is the same ILP system with our proposals deactivated

predict the complete set of unseen examples.

5.6.8 Assessing Theory Level Search

5.6.8.1 Overview and Goal

In this experiment we evaluate the importance of *Theory Level Search* (TLS) on the forecasting performance. The goal is to verify if the TLS approach improves on currently used greedy-set covering algorithms.

5.6.8.2 Procedure

The experiment consists in two steps. The first step is to run the ILP system IndLog over the datasets with model validation and model selection activated. The second step consists of running IndLog over the datasets with TLS activated altogether with model validation and model selection.

5.6.8.3 Experimental Settings

We have set the branching factor K, to 3, and configured the model validation step with the Ljung-Box test, which was set with a *p-value* of 0.75. The model selection criterium used was the AICC.

5.6.8.4 Results and Discussion

Dataset	RMSE Ratio	benchmark RMSE
Unemployment	0.99	$1.42\mathrm{E}-1$
Production	0.99	$3.77\mathrm{E}-3$
ECG	0.98	3.71

Table 5.5: Relative improvement on the FRMSE of Theory Level Search

Table 5.5, presents a summary of the experimental results. The benchmark RMSE regards to the values obtained using the ILP system with both model validation and model selection techniques as reported in table 5.4.

From this table we conclude that *Theory Level Search* usually improves the FRMSE over the tradional greedy-set covering algorithm with our proposals for model validation and model selection activated.

The limitations found are related with the trade-off between computational complexity and forecasting accuracy, mainly because there is no formal way to predict the optimal setup.

5.6.9 Assessing Hypothesis Averaging and Theory Averaging

5.6.9.1 Overview and Goal

In this experiment we evaluate the importance of Hypothesis Averaging and Theory Averaging on the forecasting performance.

We which to determine if there are some improvements in the prediction accuracy over model selection criteria like AIC and BIC used in theory level search, resulting from the usage of Hypothesis Averaging and Theory Averaging techniques for handling noise in numerical reasoning.

Dataset	Hypothesis Averaging	Theory Averaging	Benchmark RMSE
Unemployment	0.99	0.99	$1.41\text{E} - 1$
Production	0.99	0.98	$3.73\text{E} - 3$
ECG	0.98	0.99	3.63

Table 5.6: Relative FRMSE resulted from averaging hypothesis and theories after applying ILP using a model selection criterium

5.6.9.2 Procedure

The experiment consists in three steps: in first step the TLS search procedure is activated and all theories induced are collected. In the second step, theories with substatial empirical support are selelected and the akaike weights calculated for each theory. Finally, the array of theories is used to predict unseen examples in the test phase.

5.6.9.3 Experimental Settings

We have set $K = 3$ and configured the model validation step with the Ljung-Box test, which was set with a *p-value* of 0.75. The model selection criterium used was the AICC.

5.6.9.4 Results and Discussion

Table 5.6 present the results on the relative improvement on the best hypothesis found using TLS according to the setup used in the previous experiment, presented on Table 5.4.

The results of the experiment on Assessing Hypothesis Averaging and Theory Averaging procedures show improvements over the previous approach for model selection.

The importance of this technique is marked by its ability to make predictions that are better in absolute RMSE value than the best of the originating theories, which indicates a better ability to generalize and reduce model selection bias. The results were consistently better than model selection only which suggests a better ability to reduce model selection instability and variance.

The hypotheses averaging procedure introduces additional computational complexity as well as requires extra parameterization of the ILP system, which may turn its usage cumbersome in some cases.

5.7 Conclusions

In this chapter, we have proposed and evaluated improvements on the numerical reasoning capabilities of ILP systems. These improvements include: the incorporation of a noise handling mechanism based on model selection and model averaging; a proposal for a theory-level search framework; and a stopping criterium for abbreviating search inspired on the PAC learning formulation.

Our proposals were implemented in the IndLog ILP system and evaluated on time series modeling problems reported in the literature. The ILP results were better than other statistics-based time series prediction methods.

Model selection based on a information-theoretic approach reduced the complexity of the generated theories and improved prediction accuracy. Model Averaging improved the prediction accuracy by reducing bias and model selection variance.

Theory level search improved prediction accuracy by searching a larger space of theories, instead of using a greedy approach. The articulation of theory level search and model averaging improves prediction accuracy by reducing model selection bias and variance at theory level.

Experimental results indicate that including a model validation procedure during the hypothesis search assures that models will have the functional part correctly induced. No significant terms will be missing. No systematic error or process drifting will exist.

The proposals made for model validation, model selection and for measuring the learning performance can be generalised to other machine learning techniques dealing with numerical reasoning.

The ILP system discovered a new switching model based on the possibility of varying the delay on the activation rule of each sub-model of a TAR model.

We may conclude that ILP provides a good framework for combining expert knowledge, human judgement, and statistical methodologies for time series forecasting. The knowledge combination process has the potential to extract information about the time series features, while allowing the discovery of new knowledge.

Chapter 6

Network Dimensioning

6.1 Introduction

This chapter focus on provisioning the network for its traffic demands.

The importance of network dimensioning is to provide an approximation to global optimum routing while assuring quality of service requirements of the current and projected traffic demands.

This chapter is organised as follows. Section 6.2 provides an outline of the network dimensioning problem. Section 6.3 makes a comparison between several approaches to network dimensioning. Section 6.4 introduces our optimisation model for network dimensioning. Section 6.5 presents the results of simulation for the proposed optimisation model, and finally Section 6.6 presents the conclusions.

6.2 Network dimensioning characterisation

Network dimensioning has essentially two kinds of objectives: resource-oriented and traffic-oriented. The resource-oriented objectives envisage the optimisation and rationalisation of network resources utilisation in order to prepare network provisioning for future traffic demands. The traffic-oriented objectives aim at the fulfilment of the Quality of Service (QoS) requirements of the traffic demands.

A system that pursues those objectives is subject to technology restrictions and domain requirements. These restrictions may be configuration protocols latency, queue capacities, network capacity and dimension. The domain requirements to which a system is subject are availability, fault tolerance, scalability and response time to quick traffic variations.

The traffic and resource-oriented objectives lead to different solutions if optimised independently. On the one hand we want to minimise the resource usage for provisioning the network, on the other hand we want to provide the maximum QoS possible to the routed traffic. These objectives comply with a min-max formulation where, in some way, we must establish a trade-off to reach an acceptable solution.

The technology constraints limit the response times of the control actions and influence the architecture of the solution. On the one hand we need fast response time to accept new LSP requests and respond to quick traffic variations, on the other hand we need to keep the network operating in an optimal global state. These are the main lines of the problem; the solutions proposed may be categorised in three kinds of systems: on-line; semi-online and off-line systems. On-line systems may operate in a centralised or distributed manner, while semi-online and off-line systems are mainly centralised solutions. For details in the advantages and implications of each operating mode, please, refer to chapter 2.

Now that we have presented the problem let us characterise the solution. The solution must be scalable and capable of load balancing; besides that it must avoid link saturation and cope with network capacity overloading; finally it must guarantee the QoS of routed traffic and keep the network operating in an global optimum state guided by the resource and traffic oriented objectives.

6.2.1 Scalability

An important property of the solution is scalability so that a network with tens to hundreds of nodes can be handled. This feature constraints the operating mode of the system and the optimisation problem formulation. On-line and semi-online systems must have quick response-times, thus linear-programming formulations that lead to polynomial time solutions are preferable. Off-line systems have less stringent response times requirements, allowing computationally expensive solutions using non-linear or stochastic formulations like the one in [MR99b].

6.2.2 Link overloading

Link overloading degrades network performance by increasing delay [Kle64]. Saturating link capacities may render an LSP request infeasible even if, globally, the network resources are not completely consumed. This situation is particularly serious if we are running an off-line or semi-online system. In this case, the on-line routing algorithm may fail to find a route and will have no solution other than abort the request. To cope with this issue the optimisation problem formulation may follow two approaches: to use a non-linear objective function that is a power law of the link load, or restrict the maximum link load. The first solution enables the incorporation of a non-linear function derived from queue theory [Kle64] allowing to minimise the overall delay. Restricting link load has the advantage of enabling a linear programming formulation of the network dimensioning problem, which may be important if operating in on-line or semi-online schemes. However it may lead to sub-optimal resource allocation when the network is heavily loaded.

6.2.3 Load balancing

Load balancing is guaranteed by means of distributing traffic by multiple paths based on the network state, responding to fast traffic variations and offering the potential of better network-wide load balancing. MPLS supports multiple paths from source to destination and therefore it offers the potential of better network-wide load balancing. However, the LSP placement of those paths must be calculated by the network dimensioning algorithm. A formulation that takes this issue into account is therefore preferable.

6.2.4 Feasibility

There are situations in which the committed traffic demand surpasses network capacity, and in which it is not feasible to calculate a network routing plan that include all traffic demands. When the network enters such state, access control mechanisms will partially or totally discard some traffic demands. The problem formulation should be able to cope with this issue by indicating the ratio of discarded traffic demands, and preferably, minimising the financial cost of these actions.

6.2.5 Traffic QoS guarantees

The differentiated services architecture for network routing imposes different constraints in the way traffic is routed through the network. Each service class has different quality of service requirements that must be handled in the problem formulation. For instance, Real-time services such as voice and video may have admissible routes which are severely restricted in their number of hops. While on the other hand, "best-effort" traffic may be handled with less stringent requirements for delay and jitter. The solution should be aware of such differences and be capable of pursuing traffic and resource objectives subject to this constraints.

6.3 Related work

In this section we compare three time-dependent traffic engineering automation systems concerning the above described characteristics.

We have used a linear programming approach to the formulation of the network dimensioning problem. Linear programming formulations may be solved by primitives that are polynomial time algorithms [Ber98, RAO93] and that scale well with the number of nodes. This capability may be important if we expect to operate in a semi-online basis to perform reconfigurations for larger batches. In [MR99a], Mitra proposes the linear programming multi-commodity algorithm for on-line and semi-online routing. Aukia [AKK$^+$00] argues that global optimisations may be infeasible for practical reasons when large amounts of LSP must be re-routed.

Minimising a linear function may overload some links. This is contrary to the principle of load distribution. In [PT02] a multi-criteria nonlinear objective function based on the Kleinrock function [LFK73] is used to avoid overload links. Instead of minimising a convex function of the link load, we maximise network throughput and revenue while restricting the maximum load of a link.

The residual bandwidth may be managed by an on-line routing algorithm to accommodate new LSP requests. The solution proposed in [MR99a] is to maximise network revenues, but as alerted in His paper and more generally in [Ber98], this leads to overloading links with best revenues.

A minimum congestion formulation with link capacity constraints, may render the problem infeasible in overloading conditions [Bie02]. We propose the use of a Maximum Concurrent Flow (MCF) formulation [SM90]. The MCF and the minimum congestion problems are easily seen equivalent and both have important applications in telecommunications [Bie02]. The MCF approach has the advantage of coping with overload situations, where the minimum congestion formulation may be infeasible. Since MCF does not directly maximises a function of the link load, it will not overload links with best revenue.

Having multiple LSPs between an origin and source pair may be used by load balancing algorithms deployed in the nodes to distribute traffic more evenly between links on the network. We have adopted an alternative representation that allows the establishment of multiple LSPs between each egress and ingress nodes. In this chapter we will use flows as an alias for traffic trunk. A traffic trunk is an aggregation of a set of traffic flows characterised by similar edge-to-edge performance requirements [LR98].

Real-time services, such as voice and video may have admissible routes which may be severely restricted in their number of hops. We constrain the delay and packet loss to which an LSP is subject by defining constraints on the maximum transmission delay and packet loss for each LSP. This is possible by keeping statistics for the delay, jitter and packet loss rate per link; this approach is similar to the one presented in [PT02].

The next section present the formalism of the solution.

6.4 Network dimensioning model

We consider a network with $\mathcal{N}$ nodes and $\mathcal{N} \times \mathcal{N}$ directed links. Each link (i, j) connecting node i to node j has capacity c_{ij}, delay d_{ij} and packet loss p_{ij}.

For each service class $s \in S$ there are $\mathcal{K}$ bandwidth demands. We denote the bandwidth demand from source node i to destination node j by $b_{ij}(k)$ and the revenue as $r_{ij}(k)$.

The objective to be maximised is the network throughput and revenue as stated in equation (6.1), which has one component for each class of service s.

$$W_s = \sum_{(i,j)\in\mathcal{N}} \sum_{k=1,\ldots,\mathcal{K}} \lambda_{ij}(k) r_{ij}(k) \qquad \forall s \in S \tag{6.1}$$

Where $\lambda_{ij}(k)$ is the throughput of the demand $b_{ij}(k)$ carried. The throughput may be smaller than unity, meaning that only a $\lambda(k)$ part of the traffic demand $b_{ij}(k)$ was feasible to route.

In the case study, each class of service is oblivious to the presence of other service classes. We consider three layers, another for *real time* traffic, one for *prime rate* traffic and one for *best effort* traffic. This approach is quite similar to the described in [MR99a], however we do not consider a step for resource conservation.

In each iteration a service class is considered, beginning with the most delay sensitive service class. The residual capacity of each step is used for provisioning the next service class.

Without loss of generality, we may use the same model for all service classes [1]. Each flow may be customised tuning the constraints expressed by equations (6.5) to (6.7)

Putting all together we have the following model:

maximize:

$$\sum_{(i,j)\in\mathcal{N}} \sum_{k=1,\dots,\mathcal{K}} \lambda_{ij}(k) r_{ij}(k) \tag{6.2}$$

subject to:

$$\sum_j X_{ij}(k) - \sum_j X_{ji}(k) = \lambda(k) s_{ij}(k),\ \forall(i,j)\in\mathcal{N},\ k=1\dots\mathcal{K} \tag{6.3}$$

$$\sum_k X_{ij}(k) \le \mu c_{ij}, \quad \forall(i,j)\in\mathcal{N} \tag{6.4}$$

$$\sum_i \sum_j X_{ij}(k) \le \nu(k) b_{ij}(k),\ \forall(i,j)\in\mathcal{N},\ k=1\dots\mathcal{K} \tag{6.5}$$

$$\sum_i \sum_j d_{ij} X_{ij}(k) \le \delta(k) b_{ij}(k),\ \forall(i,j)\in\mathcal{N},\ k=1\dots\mathcal{K} \tag{6.6}$$

$$\sum_i \sum_j p_{ij} X_{ij}(k) \le \pi(k) b_{ij}(k),\ \forall(i,j)\in\mathcal{N},\ k=1\dots\mathcal{K} \tag{6.7}$$

$$X_{ij}(k) = 0, \quad \forall(i=j)\in\mathcal{N},\ k=1\dots\mathcal{K} \tag{6.8}$$

$$0 \le X_{ij}(k) \le b_{ij}(k), \quad \forall(i,j)\in\mathcal{N},\ k=1\dots\mathcal{K} \tag{6.9}$$

$$0 \le \lambda(k) \le 1, \quad k=1\dots\mathcal{K} \tag{6.10}$$

$$s_{ij}(k) = \begin{cases} b_{ij}(k), & i = i_k \\ -b_{ij}(k), & i = j_k \\ 0, \text{otherwise} \end{cases} \tag{6.11}$$

Our objective stated in equation (6.2) is to maximise flow revenue. If routing all flows is not feasible, then the solution will route preferably higher revenue flows. However,

[1]Multi-path formulations may depend on the algorithm selected for solving the problem see [Ber98, RAO93] for a detailed discussion

if it is feasible to route all flows, this reduces to a throughput maximisation problem.

In equation (6.3) we express the flow conservation constraints, where the term $\lambda_{ij}(k)$ expresses the throughput of flow $b_{ij}(k)$ that will be carried in the network. Equation (6.4) constrains the link load to be less than or equal to a predefined level μ of the total available capacity c_{ij} of each link.

In Equations (6.5), (6.6) and (6.7) we limit the maximum number of nodes $\nu(k)$, the maximum delay $\delta(k)$ and the maximum packet loss $\pi(k)$ which an admissible solution for routing flow $b_{ij}(k)$ should have.

We should notice that a continuous linear programming formulation subject to capacity constraints may divide flows by several paths. Which makes the solution of $\frac{\sum_i \sum_j X_{ij}(k)}{b_{ij}(k)} = \nu(k)$ the average number of nodes the traffic demand transverse.

Equations (6.8) and (6.9) remove redundant decision variables for links that do not leave the node and reduce the volume of the polyhedron, respectively.

Equation (6.10) limits the throughput of traffic demand k between 0 and 1. Where 1 expresses full bandwidth carried.

6.5 Simulation results

In our experiments we have compared the link load distribution of a Shortest Path Formulation (SPF) with the model presented in section 6.4.

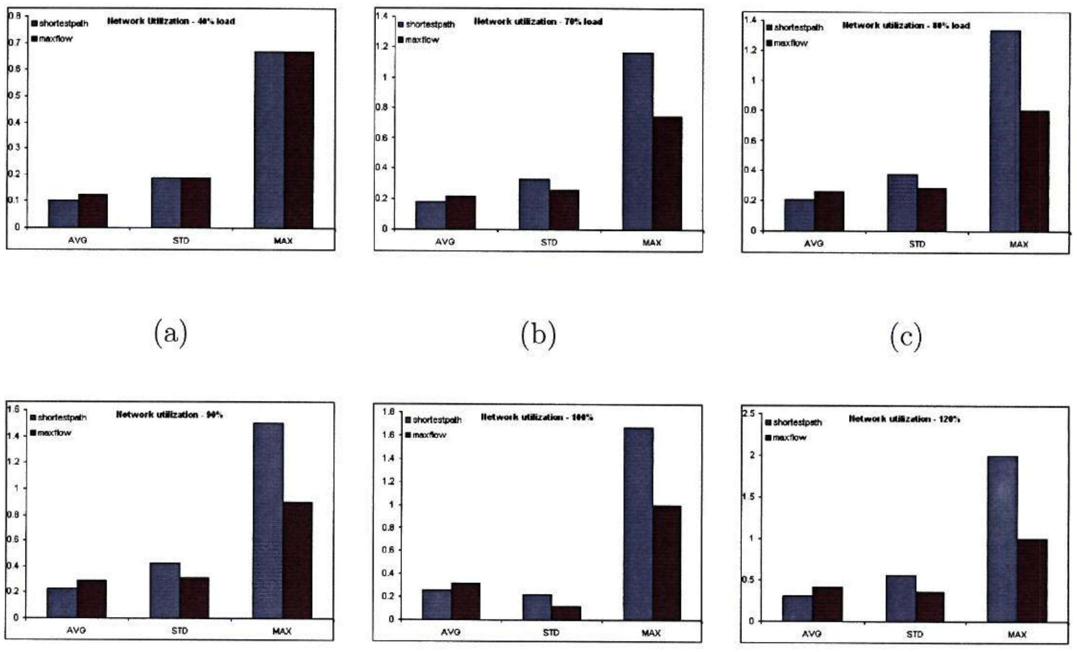
We have defined the total throughput of the network as the sum of the capacities of the first-hop links emanating from all edge nodes. In our experiments we have started with 40 % load of the total throughput as the medium load condition and scaled up to 120% for a pathogenic load condition.

For each link (i, j) we have measured the link load u_{ij}, defined in equation (6.12)

$$u_{ij} = \frac{\sum_k X_{ij}(k)}{c_{ij}} \tag{6.12}$$

Figure 6.1 shows the maximum, average and standard deviation of the link load distribution for the different traffic load profiles. We can see that the algorithm succeed in reducing the maximum link load below 1 for all cases, while SPF algorithm produces solutions up to 2 times the maximum link capacity.

While load increases, the average link load increases slightly, since the algorithm uses paths with larger number of links than the shortest path, and therefore the same load is accounted in more links than in the SPF case. We can also see that the traffic load is balanced over the network as the standard deviation of the link load utilisation is less than half of SPF for high load profiles.



(a) (b) (c)

(d) (e) (f)

Figure 6.1: Plots of average, standard deviation and maximum link load of a communications network for several traffic profiles: (a) load at 40%; (b) load at 70%; (c) load at 80%; (d) load at 90%; (e) load at 100%; (f) load at 120%. The dark colour represents the results obtained by our approach. The bright colour represents the results obtained using a Shortest Path Algorithm

6.6 Conclusions

Our proposed model is capable of mapping common QoS requirements, like packet loss, delay and bandwidth into the problem formulation. It is also possible to incorporate jitter in the model in the same way we have incorporated delay; this may be important for interactive traffic like Internet telephony. The proposed model is capable of distributing traffic across the network by introducing constraints that require link utilisation not exceeding a certain level, which may be either pre-specified or obtained adaptively. If the load committed to network exceeds the network capacity, the model discards traffic minimising economic loss. The formulation allows the placement of multiple LSPs between an egress and ingress node. The problem may be solved using polynomial primitives, which makes it suitable for on-line or semi-online operating modes.

Chapter 7

Planning Connections Re-routing

7.1 Introduction

Agents situated in dynamic and unpredictable environments require several capabilities for successful operation. Such agents must monitor the world and respond appropriately to important events. The agents should be able to accept goals, synthesise plans for achieving those goals, and execute the plans while continuing to be responsive to changes in the world. In addition, the agents should be able to re-plan as changes in the world render their plans obsolete and to reason about uncertain information.

This chapter describes our approach to the problem of selecting which routes should be changed whenever a new routing plan is calculated for the forecasted traffic demands. The algorithm aims at the minimisation of the overall number of connections re-routing subject to link capacities and traffic demands. Reducing the overall number of re-routes is particularly important in MPLS, since on the one hand, re-routing a high number of connections may be infeasible in practice or at least operationally undesirable [AKK$^+$00]; on the other hand, if the allocated capacity is greater than demand, the probability of congestion is smaller; thus, the utility of such network state is higher, therefore such routing configuration should be kept. An illustrative example where a connection re-routing shall be avoided is when the forecasted demand is less than the currently allocated bandwidth and there is spare capacity to maintain that configuration. By preserving previous routes whenever possible, the traffic engineer reduces the overall number of configuration changes, and if enough resources are available, no configuration changes will occur during several operation cycles.

In this approach, the planning algorithm establishes a trade-off between a large number of configuration changes and inefficiently affecting the resources of the network to traffic demands. These decisions are made under an uncertainty scenario, therefore, a decision-theoretical approach was followed.

The next section presents an overview of planning under uncertainty using artificial intelligence methods. Section 7.3 presents an overview of decision theory methodologies with applications in a very broad range of conditions. In Section 7.4 we present an overview of the decision problem addressed in this chapter. Section 7.5 shows the probabilistic model used. Section 7.6 shows the decision model used. Section 7.7 presents the experiments. Finally, in Section 7.8 we draw our conclusions.

7.2 Planning under uncertainty

Planning is a fundamental characteristic of intelligent behaviour. Therefore, a vast body of research has been devoted to the establishment of theories, problem representation and algorithms.

A plan is a sequence of actions that leads a system to a goal state. When the planning system knows its initial state, the exact effect of its actions and there are no exogenous interference in the environment, planning is exclusively a logic deduction problem. The agent is omniscient in its world and therefore, sensing the environment is not required because the agent can predict what state the world will be as it executes the plan.

However, there are several sources of uncertainty that affect planning systems. The agent may have partial or full ignorance of its initial state, the state transition description may be stochastic, the environment perception may be incomplete or subject to exogenous interferences. In brief, uncertainty arises in planning systems from incomplete models of the world. Either because the required information is inaccessible, too complex or impossible to include in the model of the system.

Therefore, when planning programs are used to provide courses of action in real-world settings, such as medical treatments, high-level robot control or disaster relief, they must consider that actions may have several different outcomes, some of which may be more desirable than others. They must ponder the potential of a plan achieving a goal state against both the risk of driving the system to an undesirable state and the cost of performing that plan.

Under these circumstances, decision theory [LR57] provides an attractive framework for assessing the strengths and weaknesses of a particular course of action.

The agent models the state of the world as a distribution over possible states. Given a probability distribution over the possible outcomes of an action in any state, and a preference function over outcomes, we can define an utility function on outcomes so that whenever the agent would prefer one plan over another, the preferred plan would have higher expected utility. The task of the planner then seems straightforward to find out the plan with the maximum expected utility.

Feldman and Sproul's [FS77] published one of the earliest works in decision-theoretic planning. More recently we have seen a resurgence in work on planning under uncertainty. We may put forward reasons for this. The recent advances in Bayesian inference [Pea82]. The success of Markovian approaches in areas such as speech recognition and the closely-related reinforcement learning techniques have encouraged work in planning using Markov Decision Processes (MDP) and most notably partially observable MDPs [Lit96].

An open issue seems to be the problem of separation of utility assessments from risk assessments in subjective evaluations and the attempt to place probability measures on situations that are, in fact, unknown [FL02]. We address this problem in section 7.5. But first, we will introduce some fundamental concepts of decision theory in the next section.

7.3 Decision Theory Methodologies

Mathematical decision theory is concerned with decision making under conditions of uncertainty. The central idea of mathematical decision theory is that a numerical utility function can be used to evaluate decisions [CM59].

The central theorem of decision theory shows that an optimal strategy can be selected using a single numerical value that summarises the advantages of a set of actions [Fis70]. This theorem states that: if various outcomes have known utilities and known probabilities of occurrence, then any admissible strategy is equivalent to one which maximises expected utility.

Decision methodologies under uncertainty can be grouped in Game-Theoretic, Probabilistic and methodologies that do not specify an utility or loss criteria like *procrastination.*

Game-Theoretical approaches involve uncertainty resulting from conflicting objectives and private knowledge of players.

Probabilistic approaches define a quantitative expression for the likelihood of an event, then apply the expected value maximisation principle, also called "Bayes principle".

The former approach is the most widely advocated rule in decision theory, and therefore, the followed approach in this chapter. It suggests that the option with the largest expected value should be chosen. Calculation of the expected value of a decision option requires the availability of the probabilities attached to each possible environmental state. In cases where this can not be done, subjective probabilities are attributed to each possible environmental state.

The following paragraphs are an overview of the probabilistic approaches to methodologies for deciding under uncertainty.

We start by the case where decision variables are deterministic. If A and B are two plans, plan A is preferred to B, that is $A \succ B$ if, and only if, the utility of plan A, $U(A)$, is greater than the utility of plan B, $U(B)$. This is simply formulated by:

$$A \succ B \iff U(A) > U(B) \tag{7.1}$$

If the decision variables are not deterministic, but the probabilities of the states of nature are known, the agent should then select the plan with better expected utility.

$$A \succ B \iff P(A)U(A) > P(B)U(B) \tag{7.2}$$

If the exact value of the probabilities of the states of nature are unknown, the problem can be formulated as a range of probability distributions over the possible states of nature. Assuming a continuous range of probability distributions, π, the lower bound for plan A can be defined by $p_L(A)$, and an upper bound as $p_U(A)$. The difference between $p_U(A)$ and $p_L(A)$ can be regarded as some measure of uncertainty, referred as Knightian uncertainty.

Since there is not a unique probability distribution, decisions can no longer be made by maximising expected utility because it could happen that an action that maximises an expected utility under some probability distributions could not be optimal under others.

To cope with this, two approaches to decision-making under Knightian uncertainty were derived: Relaxing Independence Assumption and Relaxing Completeness.

7.3.1 Relaxing Independence

Relaxing independence means that if A and B are two games, and A is preferred to B, that is $A \succ B$, then a mixture of A and a third game D is better than a mixture of B with D. In situations where there is no unique probability distribution, this argument will not hold, instead a weaker version of independence, which Gilboa and Schmeider [GS89] call constant-independence assumption, will hold. In this case a constant act is declared and a mixture of A and C will be preferred to a mixture of B and C.

This weaker definition of independence and uncertainty aversion provides a theoretical foundation for Wald's minimax decision rule. This rule formalises the idea that in situations where there is uncertainty about the probability distribution to apply, or no distribution at all, the action to apply should be the one holding their least possible loss. We denominate a decision policy based on the minimax assumption a risk-aversion [1] policy.

If A and B are two plans, plan A is preferred to B, that is $A \succ B$, if and only if, the minimum expected utility of plan A, regarding probability p, that is $E_pU(A)$, is greater than the minimum expected utility of plan B, for any probability p belonging to the set of probabilities π.

Formally, Gilboa and Schmeidler specify the decision rule as:

$$A \succ B \iff \min_p E_pU(A) > \min_p E_pU(B), \qquad p \in \pi \tag{7.3}$$

Under this assumption the theory allows points that are indifferent to a given constant act, C, to be represented by two indifference curves. The indifference act is represented by a line with 45°. The points above the indifference curve C will have slope equals to $\frac{p_U(A)}{1-p_U(A)}$. The points below the indifference curve C will have slope $\frac{p_L(A)}{1-p_L(A)}$.

7.3.2 Relaxing Completeness

An alternative approach to characterising preferences and describing the appropriate decision rule is to maintain the independence assumption but relaxing the completeness assumption. Completeness implies that a decision maker faced with two gambles will be able to compare them and state a preference relation between them. By

[1]in a sense that selects actions that comprises the minimum risk expressed in terms of utility

dropping the completeness, it is possible to make decisions that are not consistent with each other. In order to prevent intransitivity, Bewley introduces the *inertia* assumption [Bew88]. This assumption states that the *status quo* must be kept unless an alternative that is clearly better under all circumstances is presented. We denominate a decision policy based on the inertia assumption a conservative policy. Formally this decision rule is:

$$A \succ B \iff E_pU(A) > E_pU(B), \qquad \forall_p \in \pi \tag{7.4}$$

The Gilboa-Schmeidler preferences takes into account only the worst case scenario. The decision rule proposes actions that minimise the downside of the worst case. The Bewley preferences explores the outcomes under all probability distributions, not focusing in the worst case scenario. Given a range of probability distributions that are possible over the future period, it is theoretically possible to establish a range of utilities that are consistent with these probability distributions. The inertia assumption keep the current plan unchanged if the utilities lie in within this range. Consequentially, it is possible that the same routing plan will prevail for several periods if the utility of all connections satisfies satisfies condition (7.4). Therefore, the Bewley formulation of Knightian uncertainty has the potential to explain the behaviour of some network management policies.

7.4 Problem Formulation

In this application, we are interested in planning what connections should be re-routed. For that, we need to articulate the uncertainties of the forecasted demands with the success probability of performing a connection re-routing action. For the purposes of this chapter, a connection re-routing will have any of the following meanings: the action of allocating a bandwidth pipe, or LSP, between two points in the boundary of the network; the action of changing the route or bandwidth allocation of an already established connection. In this context, connection re-routing has the final goal of supplying a bandwidth demand. That action is considered successful if, during a given period of time, the allocated bandwidth is sufficient to supply the demand.

For planning the re-route of the connections, we have followed an approach based on the branch-and-bound algorithm, as devised by [FS77]. That algorithm explores the search space formed by the combinations of actions derived from both current and new routes. With the advantage of avoiding to explore portions of the tree where it is possible to show that the optimal plan lies elsewhere.

Since branch-and-bound compares partially complete plans, the utility model should be formulated as a sum of terms attributable to each step of the plan. So, a very important characteristic of the utility function is additivity, because it permits the calculation of the utility of the partial path to be local and incremental rather than global.

A function with that property is the expected utility of the plan. Calculated as the mathematical expectation of the utilities of the individual outcomes, in this case the outcome of an individual connection re-routing action. The partial utility of a connection is calculated as function of the traffic goal of reducing end-to-end delay. We are interested in synthesising a plan with the Maximum Expected Utility (MEU), or just the Maximum Utility (MU). The only special feature of this algorithm is considering a feasibility procedure that evaluates the capacity constraints of the network. Risk aversion and conservative decision policies may be coded in the feasibility procedure using decision rules reported in eq. 6.3 and eq. 6.4.

Notice that the expected utility is a measure of the merits of the strategy expressed by the plan. Thus, if we use the expected utility as a measure when searching for good plans, we do not guarantee good outcomes, only good strategies.

7.5 Probabilistic Model

This section provides an uncertainty model for assessing the success probability of a configuration action.

The uncertainty model considered for the forecasting errors (residuals) is the Gaussian white noise. The Gaussian white noise has also a Gaussian joint distribution with constant first and second order moments. Because the Gaussian Probability Density Function (PDF) is completely characterised by the first two moments, it is possible to model the success probability of a given configuration action.

In this section, we are particularly interested in exploring the importance of the residuals variance as well as the estimation errors of the first and second order moments. Namely, how uncertainty and risk influences choices of actions, and how it can be used to model the behaviour of the traffic engineer.

There is also the possibility of not considering only one single uncertainty model.

For example, when we do not know the distribution of the errors, or we do not want to assume a single distribution due to the lack of prior knowledge or empirical support, we may consider several uncertainty models and apply a decision-theoretic approach to planning based on Knightian uncertainty. The decision rules described in section 7.3 are useful for that purpose, Namely, Independence and Completeness relaxation assumptions.

To simplify our analysis, we will consider a single uncertainty model that considers uncertainty on the distribution parameters. Without loss of generality, the procedure devised here is applicable to the case where more than one uncertainty model is considered. For that, we only need to obtain a lower and an upper success probability that satisfies the conditions stated in section 7.3, then apply a procedure similar to the described in this section.

The present approach considers uncertainty on the mean and variance of the probability distribution of the actions. The mean and variance, are estimated from residuals using a finite sample size thus, subject to uncertainty. In this case, the variance's index of precision defines an upper and lower bound where the population variance lies, for a given confidence interval.

Let the bandwidth demand of flow k at time t be represented as $b_t(k)$ and the forecasted bandwidth demand for the same flow and time instant be represented as $\tilde{b}_t(k)$. Let the bandwidth allocated in the network to route flow k at time instant t be represented as $d_t(k)$.

The forecast errors $e_t(k)$, were modelled with a Gaussian distribution with zero mean and the variance $\sigma(k)^2$. This approach is similar to Boelhje and Lins [BL98] and Manfredo and Leuthold [ML01]for measuring and describing risk.

Since the error affects forecasts, the forecasted bandwidth is a stochastic variable with a Gaussian Probability Density Function (PDF), with mean $\tilde{b}_t(k)$ and variance $\sigma(k)$, i.e., $\tilde{b}_t(k) = b_t(k) + e_t(k), \quad e_t(k) \sim N(0, \sigma(k)^2)$.

The probability of the forecasted bandwidth being smaller than the currently allocated in the network, $P(\tilde{b}_t(k) < d_t(k)$, can then be estimated by evaluating the area between their respective intervals of the normal distribution function.

$$P(b_t(k) < d_t(k)) = \Phi\Big(\frac{d_t(k) - \tilde{b}_t(k)}{\sigma(k)}\Big) \tag{7.5}$$

Where Φ is the area between the respective intervals of the normal distribution function, as shown in figure 7.1.

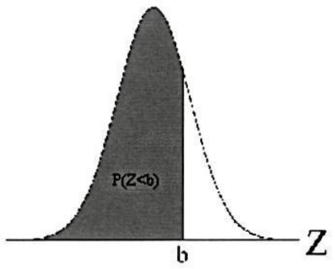


Figure 7.1: The definition of $\Phi(b)$

The parameters of the error distribution are estimated from historical forecast errors, therefore, from error theory, the standard error of the mean is:

$$\sigma_{\tilde{b}(k)} = \frac{\sigma(k)}{\sqrt{n}} \tag{7.6}$$

The standard error is most often used to express the uncertainty in the mean of a variable, but it is also more generally applied to express the uncertainty associated with any form of a central estimate. Thus one can speak of the standard error of a quantity whether or not it is the mean of a set of measurements.

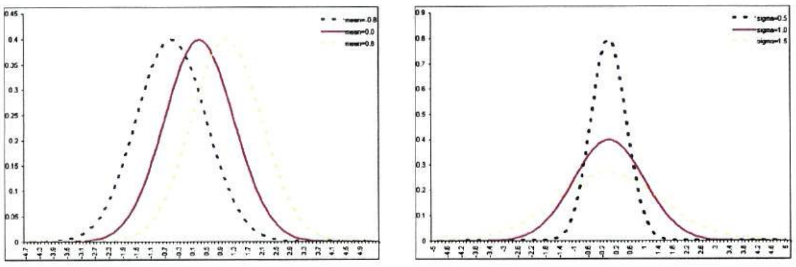
The standard error of the variance is:

$$\sigma^2_{\sigma(k)} = \sigma(k)^2 \sqrt{\frac{2}{n-1}} \tag{7.7}$$

Let us take a look in the standard error of the mean. As we can see, due to errors, the population mean can lie anywhere inside the interval $\pm\frac{\sigma(k)}{\sqrt{n}}$. Figure 7.2 plots the PDF of both *extrema* of the mean and compares it with the expected value - the PDF in the middle.

Now, Let us take a look in the standard error of the variance. As we can see, due to errors, the population variance can lie anywhere inside the interval $\pm\sigma(k)^2\sqrt{\frac{2}{n-1}}$ around the sample estimator. Figure 7.2 plots the PDF of both *extrema* of the standard deviation and compares it with the expected value - the PDF in the middle.

How can we articulate the sample errors and uncertainty? How can we define an upper and a lower probability of success?



(a) (b)

Figure 7.2: The effect of uncertainty on the distribution parameters: (a) The effect of estimation error on the mean of the forecasted error; (b) The effect of estimation error on the variance of the forecasted error

It seems that we can define a general upper and lower bound on the probability of success by simply taking both *extrema* of the standard errors and defining two PDFs for the forecasted demand. Doing that we will have two distinguished indexes of precision that give origin to two distributions, that originate the lower and upper bound on the probability of success. The PDF for the forecasted demand that originate the lower bound on the probability of success is defined by:

$$P_L(b_t(k) < d_t(k)) = \Phi\Big(\frac{d_t(k) - \tilde{b}_t(k) - \sigma_{\tilde{b}(k)}}{\sigma(k) + \sigma_{\sigma(k)}}\Big) \tag{7.8}$$

the upper bound on the probability of success is defined by:

$$P_U(b_t(k) < d_t(k)) = \Phi\Big(\frac{d_t(k) - \tilde{b}_t(k) + \sigma_{\tilde{b}(k)}}{\sigma(k) - \sigma_{\sigma(k)}}\Big) \tag{7.9}$$

Using equation 7.8 and 7.9 one can define an upper and lower bound on the success probability of a configuration action. Using these equations one may apply the decision rules presented in equations 7.3 and 7.4, to each plan or to each node generated by the branch-and-bound algorithm, implementing conservative and risk-aversion decision policies.

First we calculate the utility of the current routing plan for the forecasted demands, and then the utility of the new routing plan optimised for the demands forecasted. Then, one creates a model in which the probability of each state of the nature is considered, and for each state, the utility of each plan is calculated, giving us a framework for applying the enunciated decisions rules. We will call this framework a decision model.

7.6 Decision model

Instead of applying only one method for decision making in the planning algorithm we propose to evaluate each method separately and then try combination of methods in order to discover the framework which minimises the number of configuration changes and at same time maximises the network resources utilisation. We are also interested in modelling the behaviour of the traffic engineer facing the same decisions.

We apply this decision rules alternately and compare then each outcome with the *a posteriori* decision. Then we will compare the decision arisen from the combination of decision rules and compare with an *a posteriori* decision. The next section present the results of our experiments.

7.7 Experiments

These experiments aim at the study of the relationship among uncertainty, risk and performance in planning under Knightian uncertainty. The performance of the planning algorithm was controlled by the following parameters:

- The level of uncertainty - expressed in terms of standard error;
- The level of risk - expressed in terms of standard deviation of the forecast error;
- The standard deviation of the demand.

The performance metric is the *demand failed to supply*, caused by missed connection re-routing actions. The performance metric is calculated based on the real observed value of the demand.

7.7.1 Experimental setup

The experiment consists in generating plans in response to changes in bandwidth demand, that occurs at fixed granularity. These changes are denominated events.

Each session consists of sequences of fifty events with a predefined level of uncertainty, risk and bandwidth demand variation that allows to obtain the load discarded.

In each session, the range of standard error values is in $[0.1, 0.6]$, the range of the forecasting error values is in $[0.1, 1.0]$ and the range of the bandwidth demand variation is in $[0.2, 2.0]$. All intervals were subdivided in 10 values. All combinations of values were explored giving an outcome of 50×10^3 plans generated.

7.7.1.1 Traffic Patterns

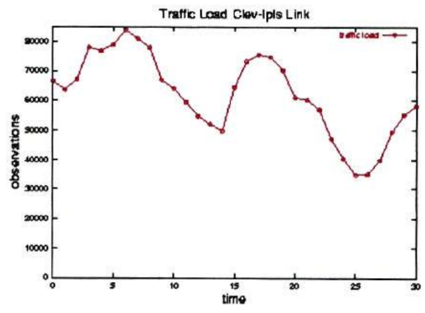
We have collected performance data from the Abilene network [Int] during 4 months. The data collected showed several traffic patterns on the demand. These traffic patterns are illustrated on figure 7.3.

The importance of these patterns on the realization of this experiment is concerned with issue of selecting a pattern to generate the demand flow of the 50 events that characterise each session.

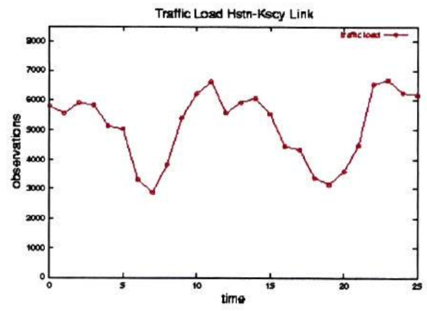
The observed traffic patterns of the demand consists mainly of two kind of patterns: Figure 7.3(a,b) show patterns with complex determinism and relatively low noise variance; Figure 7.3(c,d) show patterns with simple determinism but with low and high noise variance respectively;

We now discuss which patterns shall be selected to be included in the experiments. Figure 7.3(c) presents the pattern which is the most simple to predict, because as the simplest determinism. It potentially causes the least critical consequences in case of wrong prediction, due to the low noise variance, the magnitude of the prediction error would also be low, and consequentially, would have a low impact on the network state. The pattern of Figure 7.3(d) has the potential to cause the most critical consequences in case of a bad prediction. Although the determinism is simple, it has a relatively high noise variance, which may cause high magnitude prediction errors putting the network in a pathogenic state. The first two patterns presented in Figure 7.3(a,b), have a very low impact on real network scenarios due to the low noise variance, and also because the series presents good predictability as the results obtained on chapter 4 show.

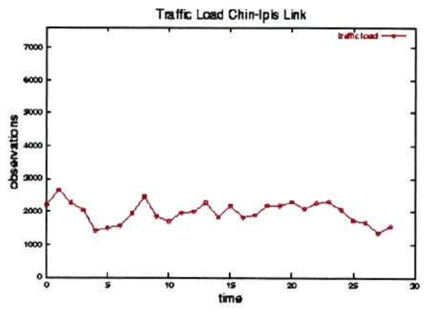
The conclusions taken from this analysis suggests to vary the traffic patterns of the demand between Figure 7.3(c) and (d) in order to generate the flow of the 50 events that characterise each session.



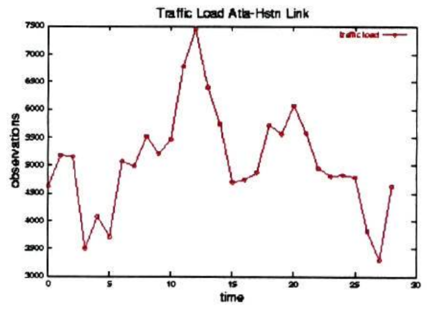
(a)



(b)



(c)



(d)

Figure 7.3: Plots of the traffic load of several links: (a) Traffic load on the Chicago-Indianapolis link; (b) Traffic load on the Atlanta-Houston link; (c) Traffic load on the Cleveland-Indianapolis link; (d) Traffic load on the Houston-Kansas link

7.7.2 Experiments results

Figure 7.4(a) represents the performance of plans generated under Knightian completeness relaxation rule and maximum expected utility. In the abscissa we have the ratio between standard deviation of the bandwidth demand, $\sigma_d(k)$, and the standard deviation of the forecast error, $\sigma(k)$. We denominate this signal to noise ratio in the graphics tag, an define it formally in equation 7.10.

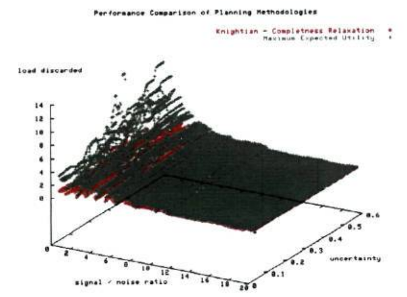
$$\text{SNR} = \frac{\sigma_d(k)}{\sigma(k)} \tag{7.10}$$

In the ordinates we have the standard error of the estimated variance. In the z axis we have the plan performance metric, which is the load discarded.

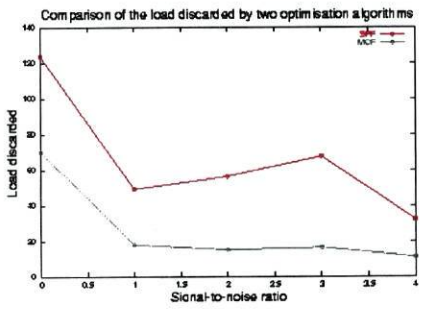
Figure 7.4(b) is the performance of plans generated under the SPF and MCF algorithm. SPF is a legacy routing algorithm, MCF is our proposal for semi-online routing. In the abscissa we have the SNR. In the z axis we have the plan performance metric, which is the load discarded. This session considered a standard error of the estimated variance of 2.31.

Figure 7.4(c) is the performance of plans generated under Knightian completeness relaxation rule and maximum expected utility. In the abscissa we have the SNR. In the z axis we have the load discarded. This session considered a standard error of the estimated variance of 0.31.

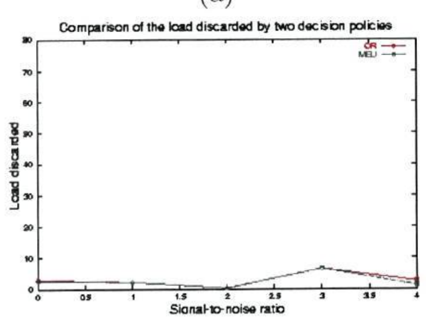
Figure 7.4(d), is the performance of plans generated under Knightian completeness relaxation rule and maximum expected utility. In the abscissa we have the SNR. In the z axis we have the load discarded. This session considered a standard error of the estimated variance of 2.31.



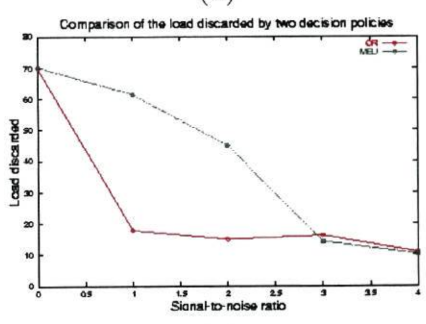
(a)



(b)



(c)



(d)

Figure 7.4: Plots of the load discarded: (a) A comparison of maximum expected utility and conservative decision policies applied to planning under uncertainty; (b) A comparison of both SPF and MCF optimisation algorithms for a planning scenario; (c) A comparison of both maximum expected utility and conservative decision policies applied to planning under uncertainty (uncertainty=0.31); (d) A comparison of both maximum expected utility and conservative decision policies applied to planning under uncertainty (uncertainty=2.37)

7.7.3 Results analysis

Figure 7.4 (a) allows us to conclude that planning maximising expected utility is much more vulnerable to uncertainty when the "signal/noise" ratio is small. This gives us an indication of which methodology should be used under such severe conditions of uncertainty and "noise".

Figure 7.4 (b) allows us to illustrate the improvement on the load discarded using the MCF optimisation algorithm.

Figure 7.4 (c) allows us to illustrate the differences between the plan generated using maximum expected utility and conservative decision policies. In this scenario of relatively low uncertainty, both plans present relatively similar performance.

Figure 7.4 (d) allows us to illustrate the differences between the plan generated using maximum expected utility and conservative decision policies. In this scenario of relatively high uncertainty, the conservative decision policy outperforms the maximum expected utility for medium values of "signal/noise" ratio.

In this uncertainty model, the upper and lower probability are symmetric. The risk and uncertainties associated to each bandwidth demand are equal to all flows, thus the plans generated maximising utility, expected utility and considering the integrity relaxation decision rule have all similar performance. Therefore, the combination scheme of planning algorithms was not useful because the majority of planning algorithms outputs identical plans, with the exception of the conservative decision policy.

7.8 Conclusions

We have experimented the usage of probabilistic decision rules in the problem of planning changes in a network configuration.

We studied the performance of Knightian decision methodologies applied to planning in artificial intelligence under diverse conditions of uncertainty and risk. The experiments' results allowed us to identify the conditions in which those methodologies should be applied. Namely, we achieved good results in conditions of severe risk and uncertainty applying completeness relaxation decision rule under Knightian uncertainty. The remaining regions are amenable to planning algorithms that maximise expected utility.

We have proposed a scheme of voting to compare plans made using different decision-making methodologies. The results obtained were similar to maximising expected utility.

Chapter 8

Conclusions

This chapter summarises the work described in this dissertation, proposes areas in which further work is required and draws the final conclusions.

8.1 Summary

This dissertation has addressed issues of Internet traffic engineering that are suitable for an Artificial Intelligence approach.

Chapter 1 motivates the need for traffic engineering on the Internet and presents arguments for an Artificial Intelligence approach.

Chapter 2 discusses Internet Traffic Engineering and decomposes the problem into several dimensions. It gives an overview of the currently available solutions, providing a comparison of traffic engineering automation systems from several perspectives. It discusses trade-off solutions for routing, network dimensioning, path restoration and traffic forecasting issues. It analyses other approaches and establishes a comparison between them and our work.

Chapter 3 presents an architecture based on a multi-agent system for dealing with the network management issues of Internet traffic engineering. The architecture supports the deployment of reactive agents inside network elements to reduce communications latency. It is proposed to enhance the adaptability of reactive agents, with local knowledge of the world, by exchanging information with deliberative agents that have network-wide knowledge. Allowing reactive agents' actions to approximate those performed by agents with network-wide knowledge and thus, network-wide optimality. It concludes arguing that such hybrid architecture responds to the scalability, reliability, fault-tolerance, openness, security and real-time requirements of this problem.

Chapter 4 presents an overview of two theories for time series analysis and forecasting. It concludes that time series analysis and forecasting is a very broad research topic that requires a considerable amount of human intervention analysing graphics and crossing the collected information with domain knowledge, being therefore a challenge to Artificial Intelligence methodologies.

Chapter 5 discusses an approach to time series analysis and forecasting automation based on ILP. It presents improvements in the numerical reasoning capabilities of ILP systems to cope with this issue, some of the proposals are generalisable to other Machine Learning methodologies. It concludes that ILP provides a framework for combining the expert knowledge, human judgement, statistical methodologies for time series forecasting and empirical results. The knowledge combination process has the potential to extract information about the time series features, while allowing the discovery of new knowledge.

Chapter 6 discusses a linear model for network dimensioning. The proposed model is capable of mapping common QoS requirements like packet loss, delay and bandwidth into the problem formulation. It is capable of distributing traffic across the network and of performing admission control by discarding traffic that minimises economic loss. It concludes that suitable adjustments on model parameters can approximate the performance of a non-linear model formulation in polynomial time.

Chapter 7 discusses the issue of connection re-routing. The proposed planning algorithm implements three decision policies: (i) Maximum expected utility; (ii) Risk aversion and (iii) conservative decision policies. It argues that under severe conditions of uncertainty and risk a conservative approach has better results. It concludes that the number of path reconfigurations may be reduced and the network efficiency may be improved by reducing the load discarded.

8.2 Future Work

This dissertation addresses the problem of time-dependent traffic engineering. We propose, as future work, the research on artificial intelligence methods to improve state-dependent traffic engineering process models.

The proposals for future work will be addressed in a per-chapter basis.

Chapter 3. Security is a major issue in telecommunications applications. The agents platform should be extended with such functionality to deal with security issues. The development of an on-line demonstration software would be a plus. Specially, with features to simulate network failures and different traffic patterns. The simulator shall provide a comparison with a legacy network routing protocol, using traffic performance statistics. This would enable us to assess the proposed system suitability for on-line routing.

Chapter 5. Empirical study of other Model Selection Criteria. Namely, SIC [SM03] because it copes with root mean square error minimisation. Improve the F-Test implementation to be full compliant with the proposal in [WM93]. Include more statistical forecasting models in the background knowledge. Synthesise an algorithm for Model Class Selection based on the proposals stated in [ECR90, Lo94, RJV96] and integrate the resultant algorithm in the ILP framework.

Chapter 6. There are several algorithms that solve the multi-commodity problem, although recent proposals [Ste92, KPP95] for optimisation with polynomial time may be a good improvement for upgrading the system for on-line routing as proposed in [AKK$^+$00]. The parallelisation of the optimisation algorithm will be an essential step in the convergence to an on-line routing approach, namely in very large networks. Bertsekas [Ber98] proposes an auction algorithm that allows the distribution of computation. Another interesting approach is stochastic optimisation algorithms [UP01]. These algorithms include mechanisms to cope with multi-service availabilities, sharing guard capacity among flows that are assumed to be statistically independent, increasing resource efficiency.

Chapter 7. There are several planning algorithms that cope with uncertainty. Markov Decision Processes (MDP) and Bayesian Inference approaches [Bly99] seem two promising research directions. An interesting challenge is to implement conservative and risk aversion policies in both algorithms. Another direction to further improvement is to develop a planning tool with the three approaches above. This would speed-up the research in several areas not directly concerned with planning. An example is robocup.

8.3 Conclusion

It is the thesis of this dissertation that Artificial Intelligence methodologies may be applied to the traffic engineering process model. The presented methodologies improved on legacy architectures and mechanisms in the following issues. (i) Provide an architecture for network management that copes with telecommunications requirements such as: resilience, fault-tolerance and real-time. (ii) Optimise network routing, using periodical and alarm-driven mechanisms, improving resource usage efficiency. (iii) Optimise pro-actively network routing, using autonomously forecasting techniques that capture traffic trends, improving the efficiency of resource utilisation in a semi-online fashion. (iv) Reduce the number of re-routed connections, using a planning algorithm that improves efficiency of resource usage, allowing effectively a semi-online approach.

This dissertation applied and evaluated the above methodologies. Chapters 2 and 3 presented arguments for supporting claim (i) Chapter 6 provided empirical evidence that supports claim (ii). Chapter 7 provided empirical evidence for supporting claim (iii) and (iv).

In summary, this dissertation has argued that deployment of the proposed mechanisms for Internet traffic engineering would help the operator to satisfy service level requirements, in a resource-efficient way, without changing current network routing protocols or traffic control mechanisms.

Appendix A

Acronyms

ACF Autocorrelation Coefficients Function

AIC Akaike Information Criterium

AR AutoRegressive

AUT Aic U-Theil

BDS Brock Dechert Scheinkman test

BIC Bayesian Information Criterium

DiffServ Differentiated Services

CAIC consistent AIC

CAICF Corrected AIC with Fisher information

CBR Constraint Based Routing

ECG ElectroCardioGram

FOL First Order Logic.

FOIL First Order Inductive Logic.

FreeBSD Free Berkeley Software Design

FRMSE Forecasts Root Mean Square Error

ILP Inductive Logic Programming

ILP* IndLog Program with prunning and noise handling improvements

IOS Internal Operating System

IP Internet Protocol

KLI Kullback-Leibler Information

LSP Label Switch Path

MAE Mean Absolute Error

MAS Multi-Agent Systems

MCF Maximum Concurrent Flow

MDP Markov Decision Processes

MEU Maximum Expecte Utility

MDL Minimum Description Length

MIS Model Inference System

MLE Maximum Likelihood Estimatior

MML Minimum Message Length.

MPEG Motion Picture Experts Group

MSA Markov Switching Autoregressive

MSC Autoregressive model with multiple structural Changes

MSE Mean Square Error

MU Maximum Utility

NAIC Normalized AIC (by sample size)

NBIC Normalized BIC (by sample size)

OLS Ordiary Least Squares

OSPF Open Shortest Path First

OSS Operating Support System

PAC Probably Approximatelly Correct

PACF Partial Auto Correlation Function

PDF Probability Density Function

PHB Per Hop Behaviors

QoS Qualtity of Service

RIP Routing Internet Protocol

RMSE Root Mean Square Error

SIC Subspace Information Criterium

SLA Service-Level Agreement

SLS Service Level Specification

SPF Shortest Path First

SSE Sum of the Squared Errors

TAR Threshold AutoRegressive

TCP Transport Control Protocol

TE Traffic Engineer

TLS Theory Level Search

URBA U-theil Root mean square error Bic Aic

VBR Variable Bit Rate

References

[AAO03] R. Camacho A. Alves and E. Oliveira. Learning time series models using inductive logic programming. In *In Proceedings of the third European Symposium on Intelligent Technologies, Hybrid Systems and their implementation on Smart Adaptive Systems*, Oulu, Finland, july 2003.

[ACE$^+$01] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. A framework for internet traffic engineering, November 2001.

[AKK$^+$00] P. Aukia, M. Kodialam, P. Koppol, T. Lakshman, H. Sarin, and B. Suter. Rates: A server for mpls traffic engineering. *IEEE Network Magazine*, 14(2), March 2000.

[Alv02] A. Alves. Analysis and forecasting of a traffic time series of a data communication network. Technical report, Faculdade de Engenharia da Universidade do Porto, January 2002.

[And90] J. Anderson. *The adaptive character of thought.* Lawrence Erlbaum, 1990.

[BA97] M. Anandarajan B. Arinze, S.-L. Kim. Combining and selecting forecasting models using rule based induction. *Computers and operations research*, 5(24):423–433, May 1997.

[BA02] K. Burnham and D. Anderson. *Model Selection and Multimodel Inference.* Springer, New York, 2002.

[BD91] P. Brockwell and R. Davis. *Time series: theory and methods.* Springer-Verlag, New York, 1991.

[Ber98] Dimitri P. Bertsekas. *Network Optimization: Continuous and Discrete Models.* Athena Scientific, 1998.

[Bew88] Truman F. Bewley. Knightian decision theory, part ii. intertemporal problems. *Cowles Foundation Discussion Papers, paper number 835*, 1988.

[BG93] I. Bratko and M. Grobelnik. Inductive learning applied to program construction and verification. In *Third International Workshop on Inductive Logic Programming*, pages 279–292, 1993. Available as Technical Report IJS-DP-6707, J. Stefan Inst., Ljubljana, Slovenia.

[Bie02] Daniel Bienstock. *Potential Function Methods for Approximately Solving Linear Programming Problems: Theory and Practice*, volume 53 of *INTERNATIONAL SERIES IN OPERATIONS RESEARCH AND MANAGEMENT SCIENCE*. Kluwer Academic Publishers, Boston, 2002.

[BL88] D. Broomhead and D. Lowe. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2:321, 1988.

[BL98] M.D. Boehlje and D.A. Lins. Risks and risk management in an industrialized agriculture. *Agricultural Finance Review*, 58:1–16, 1998.

[Bly99] Jim Blythe. An overview of planning under uncertainty. *Lecture Notes in Computer Science*, 1600:85–??, 1999.

[Boy01] T. Boyle. Towards a theoretical base for educational multimedia design. *Journal of Interactive Media in Education*, 2001.

[BR94] Jenkins Box and Reinsel. *Time Series Analysis, Forecasting and Control.* Prentice Hall, Englewood Cliffs, N.J., USA, 3rd edition edition, 1994.

[Bre96] L. Breiman. Heuristics of instability and stabilization in model selection. *Annals of Statistics*, 24:2350–2383, 1996.

[BRR98] Hendrik Blockeel, Luc De Raedt, and Jan Ramon. Top-down induction of clustering trees. In J. Shavlik, editor, *Proceedings of the 15th International Conference on Machine Learning*, pages 55–63. Morgan Kaufmann, 1998.

[BW91] D. Bunn and G. Wright. Interaction of judgmental and statistical forecasting methods: Issues & analysis. *Management Science*, 32:501–518, 1991.

[Cam00] Rui Camacho. *Inducing Models of Human Control Skills using machine Learning Algorithms.* PhD thesis, Faculdade de Engenharia da Universidade do Porto, 2000.

[Cas89] M. Casdagli. Nonlinear prediction of chaotic time series. *Physica*, 35(D):335, 1989.

[CCL98] Morsy M. Cheikhrouhou, Pierre Conti, and Jacques Labetoulle. Intelligent agents in network management, a state-of-the-art. *Networking and Information Systems*, 1(1):9–38, 1998.

[CD98] Gianni Di Caro and Marco Dorigo. Antnet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, 9:317–365, 1998.

[CF90] J. Case and M. Fedor. Rfc 1157 - simple network management protocol (snmp), 1990.

[CH98] M. Clements and D. Hendry. *Forecasting economic time series*, volume 1, chapter xxi, page 368. Cambridge University Press, Cambridge, 1998.

[CHL98] D.N. Batanov C.Q. Hung and T. Lefevre. Kbs and macro-level systems: Support of energy demand forecasting. *Computers in Industry*, 2(37):87–95, 1998.

[CM59] H. Chernoff and L.E. Moses. *Elementary Decision Theory.* John Wiley & Sons, New York, 1959.

[CM03] S. Colton and S. Muggleton. ILP for mathematical discovery. In T. Horváth and A. Yamamoto, editors, *ILP03*, volume 2835 of *LNAI*, pages 93–111. Springer, 2003.

[CR67] Laha Chakravarti and Roy. *Handbook of Methods of Applied Statistics*, volume 1. John Wiley and Sons, USA, 1967.

[CS94] C.T. Chiu and R. Scott. Intelligent forecasting support system in auditing: Expert systemand neural network approach. In *Proceedings of the Hawaii International Conference on System Sciences*, volume 3, pages 272–280. IEEE, 1994.

[Cus97] James Cussens. Part-of-speech tagging using progol. In S. Džeroski and N. Lavrač, editors, *Proceedings of the Seventh International Workshop on ILP, volume 1297 of LNAI*, pages 93–108, 1997.

[Dan56] H. E. Daniels. The approximate distribution of serial correlation coefficients. *Biometrika*, 43:1–13, 1956.

[DDRW94] S. Džeroski, L. Dehaspe, B. Ruck, and W. Walley. Classification of river water quality data using machine learning. In *Proceedings of the Fifth International Conference on the Development and Application of Computer Techniques Environmental Studies*, 1994.

[DGP+87a] J. Deneubourg, S. Goss, J. Pasteels, D. Fresneau, and J. Lachaud. Self-organization mechanisms in ant societies (i): Trail recruitment to newly discovered food sources. In J. Pasteels and J. Deneubourg, editors, *From Individual to Collective Behavior in Social Insects*, volume 54, pages 155–176. Birkhauser Verlag, Basel, 1987.

[DGP+87b] J. Deneubourg, S. Goss, J. Pasteels, D. Fresneau, and J. Lachaud. Self-organization mechanisms in ant societies (ii): Learning in foraging and division of labor. In J. Pasteels and J. Deneubourg, editors, *From Individual to Collective Behavior in Social Insects*, volume 54, pages 177–196. Birkhauser Verlag, Basel, 1987.

[DL91] Edmund H. Durfee and Victor R. Lesser. Partial global planning: A coordination framework for distributed hypothesis formation. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(5):1167–1183, - 1991.

[DM92] B. Dolšak and S.H. Muggleton. The application of inductive logic programming to finite element mesh design. In S.H. Muggleton, editor, *Inductive Logic Programming*, pages 453–472, London, 1992. Academic Press.

[DT93] S. Dzeroski and L. Todorovski. Discovering dynamics. In *10th ICML*, pages 27–29, Massachusetts, USA, 1993.

[ECR90] F. Golshaniv E. Cortes-Rello. Uncertain reasoning using the dempster-shafer method -an application in forecasting and marketing management. *Expert Systems*, 1(7):9, 1990.

[Efr86] B. Efron. How biased is the apparent error rate of a prediction rule? *JASA*, 81:461–470, 1986.

[FC88] J. Armstrong F. Collopy. Research needs in forecasting. *International Journal of Forecasting*, 4:449–465, 1988.

[FC92] J. Armstrong F. Collopy. Rule-based forecasting: Development and validation of an expert systems approach to combining time series extrapolations. *Management Science*, 38(10):1394–1414, 1992.

[fDANM96] HYBRID: Intelligent Agents for Distributed ATM Network Management. Analysis of the delay and jitter of voice traffic over the internet. In *IATA 96 Workshop at ECAI 96*, 1996.

[Fen91] C. Feng. Inducing temporal fault diagnostic rules from a qualitative model. In *Proc. Eighth International Workshop on Machine Learning*, pages 403–406. Morgan Kaufmann, San Mateo, CA, 1991.

[Fis70] P. Fishburn. *Utility theory for decision making*. John Wiley & Sons, New York, 1970.

[FL02] M. Fox and D. Long. Single trajectory opportunistic planning under uncertainty. In *Proceedings of the 3rd International NASA Workshop on Planning and Scheduling for Space*, Delft, Netherlands, 2002.

[Fle02] P. Flegkas. A policy-based quality of service management system for ip diffserv networks. *IEEE Network*, pages 50–56, March/April 2002.

[FS77] J. A. Feldman and Sproull. Decision theory and artificial intelligence ii: The hungry monkey. *Cognitive Science*, 1:158–192, 1977.

[FS86] A. M. Fraser and H. L. Swinney. Independent coordinates for strange attractors from mutual information. *Physical Review*, 33(A):1134, 1986.

[FS87] J. D. Farmer and J. Sidorowich. Predicting chaotic time series. *Physical Review Letters*, 59:845, 1987.

[GDMO92] V. Genovese, P. Dario, R. Magni, and L. Odetti. Self-organizing behavior and swarm intelligence in a pack of mobile miniature robots in search of pollutants. In *Proc. 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1575–1582, Raleigh, NC, 1992.

[Gru69] Frank Grubbs. Procedures for detecting outlying observations in samples. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(1):1–21, - 1969.

[GS89] I. Gilboa and D. Schmeidler. Maxmin expected utility with non-unique prior. *Mathematical Economics*, 18:141–153, 1989.

[GS97] R. Guerraoui and A. Schiper. Software-based replication for fault tolerance. *IEEE Computer*, 30(4):68–74, April 1997.

[GVEDZ95] AA Girgis, S. Varadan, AK El-Din, and J. Zhu. Comparison of different approaches to short-term load forecasting. *International Journal of Engineering Intelligent Systems for Electrical Engineering and Communications*, 5(3):205–210, 1995.

[HB98a] A. L. G. Hayzelden and J. Bigham. Heterogeneous multi-agent architecture for ATM virtual path network resource configuration. In S. Albayrak and F. J. Garijo, editors, *Intelligent Agents for Telecommunication Applications — Proceedings of the Second International Workshop on Intelligent Agents for Telecommunication (IATA'98)*, volume 1437, pages 45–59. Springer-Verlag: Heidelberg, Germany, 1998.

[HB98b] Alex L. G. Hayzelden and John Bigham. Software Agents in Communications Network Management: An Overview. Technical report, Queen Mary and Westfield College University of London, 1998.

[HHC+90] K. Ho, Y. Hsu, C. Chen, T. Lee, C. Liang, T. Lai, and K. Chen. Short term load forecasting of taiwan power system using a knowledge-based expert system. *IEEE Transactions on Power Systems*, 2:1214–1221, 1990.

[Hie94] Y. Hiemstra. Stock market forecasting support system based on fuzzy logic. In *Proceedings of the Hawaii International Conference on System Sciences*, volume 3, pages 281–287. IEEE, 1994.

[HL94] A. Dussauchoy H. Lacaze. C2h2f2: case comprehensive hybrid-hybrid forecasting framework. In *Proceedings of IEEE Symposium on Emerging Technologies & Factory Automation*, volume 1, pages 701–709, Piscataway, Oct. 1994. IEEE.

[Hol57] C.C. Holt. Forecasting seasonals and trends by exponentially weighted moving averages. *ONR Research Memorandum*, 52, 1957.

[HQ93] H. C. Harrison and Gong Qizhong. An intelligent business forecasting system. In *Proceedings of the 1993 ACM conference on Computer science*, pages 229–236. ACM Press, 1993.

[Hub77] P. J. Huber. *Robust statistical procedures*, volume 1, chapter x, page 56. Society for Industrial and AppliedMathematics, Philadelphia, 1977.

[Int] Internet2. Abilene network operations centre. `http://www.internet2.org`.

[JEC86] D. Ruelle J.P. Eckmann, S. Oliffson Kamphorst and S. Ciliberto. Lyapunov exponents from a time series. *Physical Review*, 34:4971, 1986.

[JKL90] T. Kohonen J. Kangas and J. Laaksonen. Variants of self-organizing maps. *IEEE Trans. on Neural Networks*, 1(1):193–99, 1990.

[JSW98] N. R. Jennings, K. Sycara, and M. Wooldridge. A roadmap of agent research and development. *Journal of Autonomous Agents and Multi-Agent Systems*, 1(1):7–38, 1998.

[JT98] Bongseog Jang and Charles Thomson. Threshold autoregressive models for vbr mpeg video traces. In *Proceedings IEEE INFOCOM '98*, pages 209–216, San Francisco, CA, March 1998.

[KA99] K.Oida and A.Kataoka. Lock-free antnets and thier adaptability evaluations. *IEICE Transactions on Communications*, J82-B(7), 1999.

[Kar00] Ilkka Karanta. Expert systems in forecast model building. In *Conference (STep 2000), "AI of Tomorrow": Symposium on Theory*, pages 77–85, Espoo, Finland, August 2000.

[Kle64] L. Kleinrock. *Communication Nets: Stochastic Message Flow and Delay.* McGraw-Hill, New York, 1964.

[KMLS92] R. King, S. Muggleton, R. Lewis, and M. Sternberg. Drug design by machine learning: The use of inductive logic programming to model the structure-activity relationships of trimethoprim analogues binding to dihydrofolate reductase. *Proceedings of the National Academy of Sciences*, 89(23):11322–11326, 1992.

[KMS$^+$01] G. Kim, P. Mouchtaris, S. Samtani, R. Talpade, and L. Wong. Qos provisioning for voip in bandwidth broker architecture: A simulation approach. In *Proceedings of the Communication networks and distributed systems modeling and simulation conference*, Phoenix, Arizona, USA, January 2001.

[KMSS96] R. King, S. Muggleton, A. Srinivasan, and M. Sternberg. Structure-activity relationships derived by machine learning: The use of atoms and their bond connectivities to predict mutagenicity by inductive logic programming. *Proceedings of the National Academy of Sciences*, 93:438–442, 1996.

[Koh84] T. Kohonen. *Self-Organization and Associative Memory.* Springer-Verlag, New York, 1984.

[KPP95] Kamath, Palmon, and Plotkin. Fast approximation algorithm for minimum cost multicommodity flow. In *SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms)*, 1995.

[KRS+99] M. Kreutz, A. Reimetz, B. Sendhoff, C. Weihs, and W. Seelen. Structure optimization of density estimation models applied to regression problems with dynamic noise. In *Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics*, 1999.

[KS97] H. Kantz and T. Schreiber. *Nonlinear Time Series Analysis.* Cambridge Univ. Press, Cambridge, UK, 1997.

[KT01] Mansour Karam and Fouad A. Tobagi. Analysis of the delay and jitter of voice traffic over the internet. In *INFOCOM*, pages 824–833, 2001.

[Lan96] P. Langley. *Elements of machine learning.* Morgan Kauffman, San Fransisco, 1996.

[LC86] R. H. Edmundson Lawrence, M. J. and M. J. O Connor. The accuracy of combining judgmental and statistical forecasts. *Management Science*, 32:1521–1532, 1986.

[LFK73] M. Gerla L. Fratta and L. Kleinrock. The flow deviation method: an approach to store-and-forward communication network design. *Networks*, 3:97 – 133, 1973.

[Lit96] M. Littman. *Algorithms for sequential decision making.* PhD thesis, Department of Computer Science, Brown University, 1996.

[Lju87] L. Ljung. *System identification - theory for the user*, chapter xxi, pages +519. Prentice-Hall, Englewood Cliffs, Amesterdam, 1987.

[lLMT98] Ruey S. Tsay lan L. Montgomery, Victor Zarnowitz and George C. Tiao. Forecasting the u.s. unemployment rate. *JASA*, 93:478–493, 1998.

[Lo94] T. Lo. Expert system for choosing demand forecasting techniques. *International Journal of Production Economics*, 1-3(33):5–15, 1994.

[LR57] R. D. Luce and Raiffa. *Games and Decisions: Introduction and Critical Survey*. John Wiley & Sons, Inc., New York, USA, 1957.

[LR98] T. Li and Y. Rekhter. Provider architecture for differentiated services and traffic engineering (paste). Rfc-2430, IETF Informational RFC-2430, October 1998.

[Luc03] Chris Lucas. Self-organizing systems (sos) faq, 2003.

[MB99] S. P. Simonovic M. Bender. Decision-support system for long-range stream flow forecasting. *Journal of Computing in Civil Engineering*, 1(8):20–34, 1999.

[MBKA92] R. Brown M. B. Kennel and H. D. I. Abarbanel. Determining embedding dimension for phase-space reconstruction using a geometrical construction. *Physical Review*, 45:3403, 1992.

[Mei00] J.D. Meiss. Frequently asked questions about nonlinear science, 2000.

[MH79] S. Makridakis and M. Hibon. Accuracy of forecasting: An empirical investigation. *Journal of Royal Statististics Society*, 38(142):97–145, 1979.

[MJ93] M.Littman and J.Boyan. A distributed reinforcement learning scheme for network routing. In *Proc. the 1993 International Workshop on Applications of Neural Networks to Telecommunications*, Hillsdale, 1993.

[MKS92] S. Muggleton, R. King, , and M. Sternberg. Predicting protein secondary structure using inductive logic programming. *Protein Engineering*, 5:647–657, 1992.

[ML01] M.R. Manfredo and R.M Leuthold. Market risk and the cattle feeding margin: An application of value-at-risk. *Agribusiness-An International Journal*, 17:333–353, 2001.

[MLM93] W. F. Fraissler M. L. Markovic. Short-term load forecast by plausibility checking of announced demand: an expert-system approach. In *European Transactions on Electrical Power Engineering*, volume 3, pages 353–358, May 1993.

[Moy89] J. Moy. Rfc 1131: Ospf, 1989.

[MR99a] D. Mitra and K.G. Ramakrishnan. A case study of multiservice, multiprioirity traffic engineering design for data communications. In *Proceedings of IEEE GLOBECOM 99*, pages 1087–1093, Brazil, December 1999.

[MR99b] D. Mitra and K.G. Ramakrishnan. A case study of multiservice, multiprioirity traffic engineering design for data communications. Technical report, Bell labs report, September 1999.

[MSB92] S. Muggleton, A. Srinivasan, and M. Bain. Compression, significance and accuracy. In D. et al. Sleeman, editor, *ML92*, pages 338–347. Morgan Kauffman, 1992.

[MSBG01] Olivier Marin, Pierre Sens, Jean-Pierre Briot, and Zahia Guessoum. Towards adaptive fault tolerance for distributed multi-agent systems. In *Proceedings of ERSADS*, 2001.

[Mug96] S. Muggleton. Learning from positive data. In S. Muggleton, editor, *ILP96*, volume 1314 of *LNAI*, pages 358–376. Springer, 1996.

[MW89] S. Makridakis and S. Wheelwright. *Forecasting Methods for Management.* John Wiley & Sons, New York, fifth edition edition, 1989.

[Nan97] B. Nangle. A survey on intelligent agents in telecommunications. Iag technical report, Trinity College, 1997.

[New01] A. Newell. The knowledge level. *Journal of Artificial Intelligence*, 18:87–127, 2001.

[NLT99] R. Mason N. Lertpalangsunti, C. Chan and P. Tontiwachwuthikul. Toolset forconstruction of hybrid intelligent forecasting systems: Application for water demand prediction. *Artificial Intelligence in Engineering*, 13(1):21–42, 1999.

[NP77] G. Nicolis and I. Progogine. *Self-Organization in Nonequilibrium Systems.* John Wiley & Sons, New York, 1977.

[Pea82] J. Pearl. Reverend bayes on inference engines: A distributed hierarchical approach. In *Proceedings of the second national conference on artificial intelligence (AAAI-82)*, pages 133–136, Menlo Park, CA, 1982. AAAI Press.

[Pri81] M.B. Priestley. *Spectral Analysis and Time Series*, volume 1 and 2. Academic Press, New York, 1981.

[PT02] P. Flegkas & G. Pavlou P. Trimintzios. Policy-driven traffic engineering for intra-domain quality service provisioning. In *Proceedings of Quality of Service for Future Internet Services (QofIS'02)*, pages 272–280, Zuerich, Switzerland, 2002.

[PW] Peter Pirolli and Mark Wilson. A theory of the measurement of knowledge content, access, and learning.

[Que49] M.H Quenoille. The joint distribution of serial correlation coefficients. *Annals of mathematical statistics*, 20:561–571, 1949.

[RAO93] T. Magnanti R. Ahuja and J. Orlin. *Network Flows: Theory, Algorithms and Applications*. Prentice-Hall, 1993.

[Rei03] David Reilly. Autobox - automatic forecasting systems. ="http://www.autobox.com/", 2003.

[Rek88a] J. Rekhter. Rfc 1074: The nsfnet backbone spf based interior gateway protocol, 1988.

[Rek88b] J. Rekhter. Rfc 1074: The nsfnet backbone spf based interior gateway protocol, 1988.

[RJV96] S. L. Pearce R. J. Vokurka, B. E. Flores. Automatic feature identification and graphical support in rule-based forecasting: a comparison. In *Proceedings of the Annual Meeting of the Decision Sciences Institute*, volume 2, pages 16–18. Decision Sciences Institute, November 1996.

[RK96a] A. Rueda and W. Kinsner. [incorrect:] self-similarity of ethernet traffic traces. In *[INCORRECT:]IEEE Canadian conference on electrical and computer engineering*, volume II, pages 830–833. Morgan Kaufmann, May 1996.

[RK96b] A. Rueda and W. Kinsner. A survey of traffic characterization techniques in telecommunication networks. In *IEEE Canadian conference on electrical and computer engineering*, volume II, pages 830–833. Morgan Kaufmann, May 1996.

[RN03] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach (second edition)*. Prentice-Hall, Englewood Cliffs, NJ, 2003.

[Ros91] M. Rose. Rfc 1215: Convention for defining traps for use with the snmp, 1991.

[RP] R-Project. The r project for statistical computing. `http://www.r-project.org`.

[RWL01] Marcus O'Connor Richard Webby and Michael Lawrence. Judgmental time series forecasting using domain knowledge. In *In J. Scott Armstrong editor, Principles of Forecasting: A Handbook for Researchers and Practitioners*. Kluwer, 2001.

[SC89] George W. Snedecor and William G. Cochran. *Statistical Methods*. Iowa State University Press, USA, eighth edition edition, 1989.

[SC99] Ashwin Srinivasan and Rui Camacho. Numerical reasoning with an ILP system capable of lazy evaluation and customised search. *Journal of Logic Programming*, 40(2-3):185–213, 1999.

[Sch78] Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6:461–464, 1978.

[Sha97] J Shao. An asymptotic theory for linear model selection. *Statistica Sinica*, 7:221–264, 1997.

[SM90] Farhad Shahrokhi and D. W. Matula. The maximum concurrent flow problem. *Journal of the ACM (JACM)*, 37(2):318–334, 1990.

[SM03] Masashi Sugiyama and Klaus-Robert M&#252;ller. The subspace information criterion for infinite dimensional hypothesis spaces. *J. Mach. Learn. Res.*, 3:323–359, 2003.

[Smi92] L. A. Smith. Identification and prediction of low dimensional dynamics. *Physica*, 58(D):500, 1992.

[SR97] S.Kumar and R.Miikkulainen. Dual reinforcement q-routing: An on-line adaptive routing algorithm. In *Proc. the 1993 International Workshop on Applications of Neural Networks to Telecommunications*, St. Louis, 1997.

[SR0b] N. R. Sanders and L. P. Ritzman. Improving short-term forecasts. *Omega*, 18:365–373, 1990b.

[Sri04] A. Srinivasan. The aleph manual. `http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/`, , 2004.

[SSY$^+$] John Sum, Hong Shen, G. Young, Jie Wu, and Chi-Sing Leung. Analysis on extended ant routing algorithms for network management.

[Ste92] Clifford Stein. *Approximation Algorithms for Multicommodity Flow and Shop Scheduling Problems.* PhD thesis, MIT, 1992.

[SvD03] Boriss Silvertovs and Dick van Dijk. Forecasting industrial production with linear, nonlinear and structural change models. Technical report, Econometric Institute report from Erasmus University Rotterdam, May 2003.

[TA02] C. Scoglio T. Anjali. A new path selection algorithm for mpls networks based on available bandwidth estimation. In B. Stiller, editor, *Proc. QofIS/ICQT 2002, LNCS 2511*, Heidelberg, 2002. Springer-Verlag.

[Tak81] F. Takens. Detecting strange attractors in turbulence. In *Lecture Notes in Math*, volume 898. Springer Verlag, New York, 1981.

[TAPF01] P. Trimintzios, I. Andrikopoulos, G. Pavlou, and P. Flegkas. A management and control architecture for providing ip differentiated services in mpls-based networks. *IEEE Communications Magazine*, 39(5), May 2001. Special Issue on IP-oriented Operations and Management.

[TGGD90] G. Theraulaz, S. Goss, J. Gervet, and J. Deneubourg. *Task Differentiation in Polistes Waps Colonies: a Model for Self-Organizing Group of Robots.* John Wiley & Sons, New York, 1990.

[Ton90] H. Tong. *Nonlinear time series, a dynamical system approach.* Claredon press, Oxford, UK, 1st edition edition, 1990.

[TSC91] J. Yorke T. Sauer and M. Casdagli. Embedology. *Journal of Statistical Physics*, 65:579, 1991.

[UP01] S. Uryasev and P. Pardalos, editors. *Stochastic Optimization: Algorithms and Applications.* Kluwer Academic Publishers, 2001.

[Val84] L. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984.

[Vil02] Pere Vila. Automated network management using a hybrid multiagent, 2002.

[WABL96] J. A. Scheinkman W. A. Brock, W. D. Dechert and B. LeBaron. A test for independence based on the correlation dimensionr. *Econometric Review*, 15(3):197–235, 1996.

[Wei90] W. W. S. Wei. *Time series analysis: univariate and multivariate methods*, chapter xv, pages +496. PAddison-Wesley, Amesterdam, 1990.

[Wei04] E. Weisstein. *MathWorld.* A Wolfram Web Resource, 2004.

[Whi63] P. Whittle. *Prediction and regulation by linear least squares methods.* The English Universities Press, Ltd, London, 1963.

[Whi97] T. White. Routing with swarm intelligence. Technical Report SCE-97-15, nasa, 1997.

[Win60] P.R. Winters. Forecasting sales by exponentially weighted moving averages. *Management Science*, 6:324–342, 1960.

[WM93] R. Walpole and R. Myers. *Probability and Statistics for Engineers and Scientists.* Macmillan Publishing Company, New York, USA, 5th edition edition, 1993.

[WR90] Brandau R. Weihmayer R. Cooperative distributed problem solving for communication network management. *Computer Communications*, 13(9):547–556, 1990.

[WWE96] M. Taqqu W. Willinger and A. Erramilli. A bibliographical guide to self-similar traffic and performance modeling for modern high-speed networks. In *In F. P. Kelly, S. Zachary and I. Ziedins editors, Stochastic Networks: Theory and Applications*, pages 339–366. Clarendon Press, Oxford University Press, Oxford, 1996.

[XHBN00] X. Xiao, A. Hannan, B. Bailey, and L. Ni. Traffic engineering with mpls in the internet. *IEEE Network Magazine*, March 2000.

[Ye98] J. Ye. On measuring and correcting the effects of data mining and model selection. *JASA*, pages 120–131, 1998.

[YG62] G.T. Jacobi Yovits, M.C. and G.D. Goldstein. *Self-Organizing Systems.* McGregor & Werner, Washington, 1962.

[Zel01] Filip Zelezny. Learning functions from imperfect positive data. In *International Workshop on Inductive Logic Programming*, pages 248–260, 2001.

[ZM93] J. Zelle and R. Mooney. Learning semantic grammars with constructive inductive logic programming. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 817–822, California, 1993. Morgan Kaufmann.