

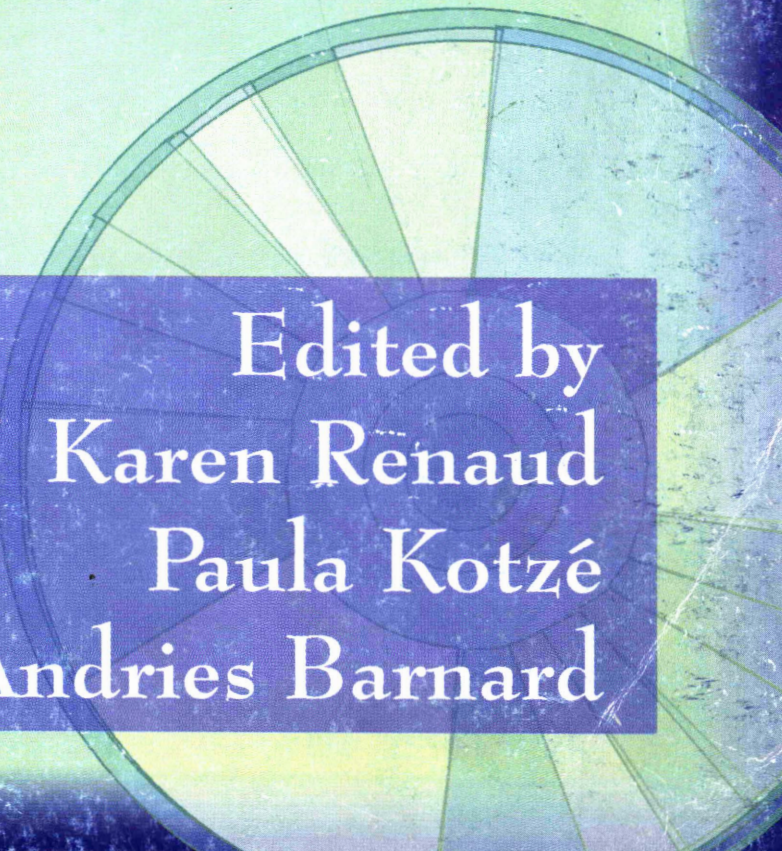
# HARDWARE, SOFTWARE AND PEOPLEWARE



UNISA



## SAICSIT 2001

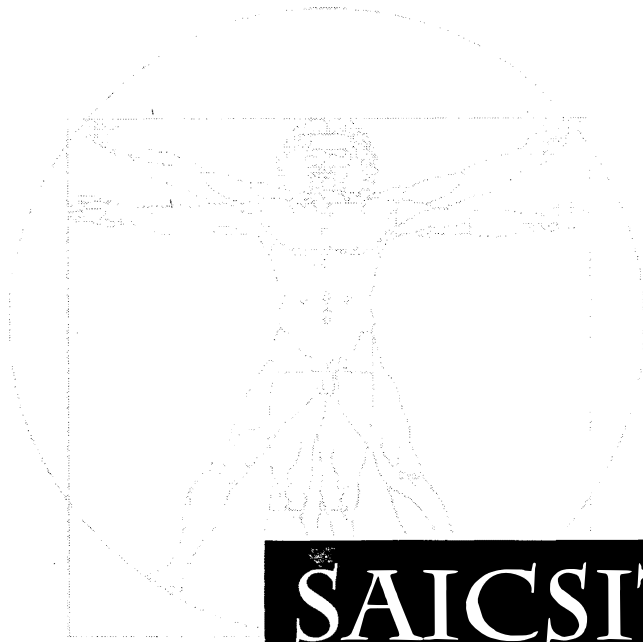


Edited by  
Karen Renaud  
Paula Kotzé  
Andries Barnard



# HARDWARE, SOFTWARE AND PEOPLEWARE

**South African Institute of Computer  
Scientists and Information Technologists**  
**Annual Conference**  
*25 – 28 September 2001*  
*Pretoria, South Africa*



**SAICSIT 2001**



*Edited by Karen Renaud, Paula Kotzé & Andries Barnard*  
*University of South Africa, Pretoria*

# Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists

First Edition, First Impression  
ISBN: 1-86888-195-4

© The South African Institute of Computer Scientists and Information Technologists (SAICSIT)

Abstracting is permitted with credit to the source. Liberties are permitted to photocopying beyond the limits of South African copyright law for private use for research purposes. For other photocopying, reprint or republication permission write to the SAICSIT President, Department of Computer Science and Information Systems, UNISA, P O Box 392, Pretoria, 0003, South Africa.

The Publisher makes no representation, expressed or implied, with regard to the accuracy of the information contained in this book and cannot accept liability for any errors or omissions that may be made. The Publisher is not responsible for the use which might be made of the contents of this book.

Published by Unisa Press  
University of South Africa  
P O Box 392, Pretoria, 0003

Cover Design by Tersia Parsons

Editors: Karen Renaud, Paula Kotzé & Andries Barnard

Electronic Publication by the Editors

Printed by Unisa Press  
2001

# Table of Contents

<b>Message from the SAICSIT President</b> .....	iv
<b>Message from the Chairs</b> .....	vi
<b>Conference Organisation</b> .....	vii
<b>Referees</b> .....	viii

## Keynote Speakers

<i>Cyber-economies and the Real World</i> .....	xi
Alan Dix	
<i>Computer-aided Instruction with Emphasis on Language Learning</i> .....	xiv
Lut Baten	
<i>Internet and Security Trends</i> .....	xv
Arthur Goldstuck	
<i>The Future of Data Compression in E-technology</i> .....	xvi
Nigel Horspool	
<i>Strategic Planning for E-Commerce Systems: Towards an Inspirational Focus</i> .....	xvii
Raymond Hackney	

## Research Papers

### Human-Computer Interaction / Virtual Reality

<i>The Development of a User Classification Model for a Multi-cultural Society</i> .....	1
M Streicher, J Wesson & A Calitz	
<i>Real-Time Facial Animation for Virtual Characters</i> .....	11
D Burford & E Blake	
<i>The Effects of Avatars on Co-presence in a Collaborative Virtual Environment</i> .....	19
J Casanueva & E Blake	

### Education

<i>Structured Mapping of Digital Learning Systems</i> .....	29
E Cloete & L Miller	

### Formal Methods

<i>The specification of a multi-level marketing business</i> .....	35
A van der Poll & P Kotzé	
<i>Finite state computational morphology - the case of the Zulu noun</i> .....	45
L Pretorius & S Bosch	
<i>Combining context provisions with graph grammar rewriting rules: the three-dimensional case</i> .....	54
A Barnard & E Ehlers	

### Human-Computer Interaction / Web Usability

<i>Web Site Readability and Navigation Techniques: An Empirical Study</i> .....	64
P Licker, R Anderson, C Macintosh & A van Kets	
<i>Jiminy: Helping Users to Remember Their Passwords</i> .....	73
K Renaud & E Smith	

### Information Security

<i>Computer Security: Hacking Tendencies, Criteria and Solutions</i> .....	81
M Botha & R von Solms	
<i>An access control architecture for XML documents in workflow environments</i> .....	88
R Botha & J Eloff	

## **Graphics and Ethics**

<i>Model-based Segmentation of CT Images</i> .....	96
O Marte & P Marais	
<i>Towards Teaching Computer Ethics</i> .....	102
C de Ridder, L Pretorius & A Barnard	

## **Human-Computer Interaction / Mobile Devices**

<i>Ubiquitous Computing and Cellular Handset Interfaces – are menus the best way forward?</i> .....	111
G Marsden & M Jones	
<i>A Comparison of the Interface Effect on the Use of Mobile Devices</i> .....	120
J Franken, A Stander, Z Booley, Z Isaacs & R Rose	
<i>The Effect of Colour, Luminance, Contrast, Icons, Forgiveness and Closure on ATM Interface Efficiency</i> .....	129
A Stander, P van der Zee, & Y Wang	

## **Object Orientation**

<i>JavaCloak - Considering the Limitations of Proxies for Facilitating Java Runtime Specialisation</i> .....	139
K Renaud	

## **Hardware**

<i>Hierarchical Level of Detail Optimization for Constant Frame Rate Rendering</i> .....	147
S Nirenstein, E Blake, S Windberg & A Mason	
<i>A Proposal for Dynamic Access Lists for TCP/IP Packet Filtering</i> .....	156
S Hazelhurst	

## **Information Systems**

<i>The Use of Technology to Support Group Decision-Making in South Africa</i> .....	165
J Nash, D Gwilt, A Ludwig & K Shaw	
<i>Creating high Performance I.S. Teams</i> .....	172
D C Smith, M Becker, J Burns-Howell & J Kyriakides	
<i>Issues Affecting the Adoption of Data Mining in South Africa</i> .....	182
M Hart, E Barker-Goldie, K Davies & A Theron	

## **Information Systems / Management**

<i>Knowledge management: do we do what we preach?</i> .....	191
M Handzic, C Van Toorn, & P Parkin	
<i>Information Systems Strategic Planning and IS Function Performance: An Empirical Study</i> .....	197
J Cohen	

## **Formal Methods**

<i>Implication in three-valued logics of partial information</i> .....	207
A Britz	
<i>Optimal Multi-splitting of Numeric value ranges for Decision Tree Induction</i> .....	212
P Lutu	

## Abstracts of Electronic Papers

<i>Lessons learnt from an action research project running groupwork activities on the Internet: Lecturers' experiences</i> .....	221
T Thomas & S Brown	
<i>A conceptual model for tracking a learners' progress in an outcomes-based environment</i> .....	221
R Harmse & T Thomas	
<i>Introductory IT at a Tertiary Level – Is ICDL the Answer?</i> .....	222
C Dixie & J Wesson	
<i>Formal usability testing – Informing design</i> .....	222
D van Greunen & J Wesson	
<i>Effectively Exploiting Server Log Information for Large Scale Web Sites</i> .....	223
B Wong & G Marsden	
<i>Best Practices: An Information Security Development Trend</i> .....	223
E von Solms & J Eloff	
<i>A Pattern Architecture, Using patterns to define an overall systems architecture</i> .....	224
J van Zyl & A Walker	
<i>Real-time performance of OPC</i> .....	224
S Kew, & B Dwolatzky	
<i>The Case for a Multiprocessor on a Die: MoaD</i> .....	225
P Machanick	
<i>Further Cache and TLB Investigation of the RAMpage Memory Hierarchy</i> .....	225
P Machanick & Z Patel	
<i>The Influence of Facilitation in a Group Decision Support Systems Environment</i> .....	226
T Nepal & D Petkov	
<i>Managing the operational implications of Information Systems</i> .....	226
B Potgieter	
<i>Finding Adjacencies in Non-Overlapping Polygons</i> .....	226
J Adler, GD Christelis, JA Deneys, GD Konidaris, G Lewis, AG Lipson, RL Phillips, DK Scott-Dawkins, DA Shell, BV Strydom, WM Trakman & LD Van Gool	

# Message from the SAICSIT President

The South African Institute of Computer Scientists and Information Technologists (SAICSIT) was formed in 1982 and focuses on research and development in all fields of computing and information technology in South Africa. Now in the 20th year of its existence, SAICSIT has come of age, and through its flagship series of annual conferences provides a showcase of not only the best research from the Southern-African region, but also of international research, attracting contributions from far afield. SAICSIT does, however, not exist or operate in isolation.

More than 50 years have passed since the first electronic computer appeared in our society. In the intervening years technological development has been exponential. Over the last 20 years there has been a vast growth and pervasiveness of computing and information technology throughout the world. This has led into the expansion and consolidation of research into a diversity of new technologies and applications in diverse cultural environments. During this period huge strides have also been made in the development of computing devices. The processing speed of computers has increased thousand-fold and memory capacity from megabytes to gigabytes in the last decade alone. The Southern African region did not miss out on these developments.

It is hardly possible for such quantitative expansion not to bring a change in quality. Initially computers had been developed mainly for purposes such as automation for the improvement of processing, labour-reduction in production and automation control of machinery, with artificial intelligence, which made great strides in the 1980s, seen as the ultimate field to which computers could be applied. As we moved into the 1990s it was recognized that such an automation route was not the only direction in the improvement of computers. The expansion of processing power has enabled image data to be incorporated into computer systems, mainly for the purpose of improving human utilisation. For most computer technologies of the 1990s, including the Internet and virtual reality, automation was not the ultimate purpose. Humans were increasingly actively involved in the information-processing loop. This involvement has gradually increased as we move into the 21<sup>st</sup> century. Development of computer technology based not on automation, but on interaction, is now fully established.

The method of interaction has significantly changed as well. The expansion of computer ability means that the same function can be performed far more cheaply and on smaller computers than ever before. The advent of portable and mobile computers and pervasive computing devices is ample evidence of this. The need for users to be at the same location as a computer in order to reap the benefits of software installed on that computer is becoming an obsolete notion. Time and space are no longer constraints. One of the most discussed impacts of computing and information technology is *communication* and the easy accessibility of information. This changes the emphasis for research and development – issues such as cultural, political, and economic differences must, for example, be accommodated in ways that researchers have not previously considered. Our goal should be to enable users to benefit from technological advances, hence matching the skills, needs, and expectations of users of available technologies to their immense possibilities.



The conference theme for the SAICSIT 2001 Conference – *Hardware, Software and Peopleware: The Reality in the Real Millennium* – aims to reflect technological developments in all aspects related to computerised systems or computing devices, and especially reflect the fact that each influences the others.

Not only has SAICSIT come of age in the 21<sup>st</sup> century, but so has the research and development community in Southern Africa. The outstanding quality of papers submitted to SAICSIT 2001, of which only a small selection is published in this collection, illustrates both the exciting and developing nature of the field in our region. I hope that you will enjoy SAICSIT 2001 and that it will provide opportunities to cultivate and grow the seeds of discussion on innovative and new developments in computing and information technology.

Paula Kotzé  
SAICSIT President

# Message from the Chairs

Running this conference has been rewarding, exciting and exhausting. The response to the call for papers we sent out in March was overwhelming. We received 64 paper submissions for our main conference and twelve for the postgraduate symposium. We had a panel of internationally recognized reviewers, both local and international. The response from the reviewers was impressive – accepting a variety of papers and *mostly* returning the reviews long before the due date. We were struck, once again, by the sheer magnanimity of academia – as busy as we all are, we still manage to contribute fully to a conference such as SAICSIT.

After an exhaustive review process, where each paper was reviewed by at least three reviewers, the program committee accepted 26 full research papers and 14 electronic papers. Five papers were referred to the postgraduate symposium, since they represented work in progress – not yet ready for presentation to a full conference but which nevertheless represented sound and relevant research. The papers published in this volume therefore represent research of an internationally high standard and we are proud to publish it. Full electronic papers will be available on the conference web site (<http://www.cs.unisa.ac.za/saicsit2001/>).

Computer Science and Information Systems academics in South Africa labour under difficult circumstances. *The popularity of IT courses stems from the fact that IT qualifications are in high demand in industry, which leads in turn to a shortage of IT academic staff to teach the courses, even when posts are available. The net result is that fewer people teach more courses to more students. IT departments thus rake in ever-increasing amounts of state subsidy for their universities. These profits, euphemistically labelled “contribution to overhead costs”, are deployed in various ways: cross-subsidization of non-profitable departments; maintenance of general facilities; salaries for administrative personnel, etc. Sweeteners of generous physical resources for the IT departments may be provided. We have yet to hear of a University in South Africa where significant concessions have been made in terms of industry-related remuneration. At best, small subventions are provided. As a result, shortages of quality staff remain acute in most IT departments – especially at senior teaching levels. What is even worse is that academics in these departments have to motivate the value of their conference contributions and other IT outputs to selection committees, often dominated by sceptical academic power-brokers from the more traditional departments whose continued survival is underwritten by IT’s contribution to overhead costs.*<sup>1</sup>

The papers published in this volume are conclusive evidence of the indefatigability and pertinacity of Computer Science and Information Systems academics and technologists in South Africa. We are proud to be part of such a prestigious and innovative group of people.

In conclusion, we would like to thank the conference chair, Prof Paula Kotzé, for her support. We also specially thank Prof Derrick Kourie for his substantial contribution. Finally, to all of you, contributors, presenters, reviewers and organisers – a big thank you – without you this conference could not be successful.

Enjoy the Conference!  
Karen Renaud & Andries Barnard

---

<sup>1</sup> This taken almost verbatim from Professor Derrick Kourie’s SACLA 2001 paper titled: “*The Benefits of Bad Teaching*”.

# Conference Organisation

## **General Chair**

Paula Kotzé

## **Programme Chairs**

Karen Renaud

Andries Barnard

## **Organising Committee Chairs**

Lucas Venter, Alta van der Merwe

## **Art and Design**

Tersia Parsons

## **Sponsor Liaison**

Paula Kotzé, Chris Bornman

## **Secretarial & Finances**

Christa Prinsloo, Elmarie Havenga

## **Marketing & Public Relations**

Klarissa Engelbrecht, Elmarie van Solms, Adriaan Pottas, Mac van der Merwe

## **Audio Visual**

Tobie van Dyk, Andre van der Poll, Mac van der Merwe

## **Program Committee**

Bob Baber – McMaster University, Canada  
Andries Barnard – University of South Africa  
Judy Bishop – University of Pretoria  
Andy Bytheway – University of the Western Cape  
Andre Calitz – University of Port Elizabeth  
Elsabe Cloete – University of South Africa  
Carina de Villiers – University of Pretoria  
Alan Dix – Lancaster University, United Kingdom  
Jan Eloff – Rand Afrikaans University  
Andries Engelbrecht – University of Pretoria  
Chris Johnson – University of Glasgow, United Kingdom  
Paul Licker – University of Cape Town  
Paula Kotzé – University of South Africa  
Derrick Kourie – University of Pretoria  
Philip Machanick – University of the Witwatersrand  
Gary Marsden – University of Cape Town  
Don Petkov – University of Natal in Pietermaritzburg  
Karen Renaud – University of South Africa  
Ian Sanders – University of the Witwatersrand  
Derrick Smith – University of Cape Town  
Harold Thimbleby – Middlesex University, United Kingdom  
Theda Thomas – Port Elizabeth Technikon  
Herna Viktor – University of Pretoria, South Africa  
Bruce Watson – Universities of Pretoria and Eindhoven  
Janet Wesson – University of Port Elizabeth

# Referees

Molla Alemayehu	Klarissa Engelbrecht	Pekka Pihlajasaari
Trish Alexander	David Forsyth	Nelisha Pillay
Adi Attar	John Galletly	Laurette Pretorius
Bob Baber	Vashti Galpin	Karen Renaud
Andries Barnard	Wayne Goddard	Ingrid Rewitzky
John Barrow	Alexandr� Hardy	Sheila Rock
Judy Bishop	Scott Hazelhurst	Markus Roggenbach
Gordon Blair	Johannes Heidema	Ian Sanders
Arina Britz	Tersia H�rne	Justin Schoeman
Andy Bytheway	Chris Johnson	Martie Schoeman
Andr� Calitz	Bob Jolliffe	Elsje Scott
Charmain Cilliers	Paula Kotz�	Derek Smith
Elsabe Cloete	Derrick Kourie	Elm� Smith
Gordon Cooper	Les Labuschagne	Adrie Stander
Richard Cooper	Paul Licker	Harold Thimbleby
Annemieke Craig	Philip Machanick	Theda Thomas
Thad Crews	Anthony Maeder	Judy Van Biljon
Quintin Cutts	David Manlove	Alta Van der Merwe
Michael Dales	Gary Marsden	Andr� van der Poll
Carina de Villiers	Thomas Meyer	Tobias Van Dyk
Alan Dix	Elsa Naud�	Lynette van Zijl
Dunlop Mark	Martin Olivier	Lucas Venter
Elize Ehlers	Don Petkov	Herna Viktor
Jan Eloff		Bruce Watson
Andries Engelbrecht		Janet Wesson

## Conference

### Sponsors



## **Keynote Abstracts**



# Optimal Multi-splitting of Numeric Ranges for Decision Tree Induction

P.E.N. Lutu

Department of Computer Science & Information Systems  
Vista University, Mamelodi Campus  
lutu-p@marlin.vista.ac.za

**Abstract:** Data mining is the process of extracting informative patterns from data stored in a database or data warehouse. Decision tree induction algorithms, from the area of machine learning are well suited for building classification models in data mining. The handling of continuous-valued attributes in decision tree induction has received a lot of research attention in recent years. Typically, an evaluation function is used to dynamically select the best multi-split for the range of values of a continuous-valued attribute. This paper discusses useful and well behaved evaluation functions and proposes an algorithm for optimal multi-splitting

**Key words:** Knowledge discovery in databases, machine learning, data mining, decision tree induction, classification.

**Computing Review Categories:** H.2.8, I.2.6

## 1. Introduction: Machine Learning and Data Mining

Knowledge Discovery in Databases (KDD), is the non-trivial process of identifying valid, novel, potentially useful and ultimately understandable patterns in data [Fayyad 1996]. Data mining, which is one step of the overall KDD process, is the act of extracting useful and informative patterns from data. It may be done for purposes of seeking knowledge about concepts (classification), taxonomies (clustering) or regularities in data. Data mining tasks typically require the efficient handling of large sets of data consisting of both nominal (categorical) and continuous (integer or real) valued attributes.

Inductive learning algorithms, from the area of machine learning, are well suited for adaptation for data mining, for classification purposes. Inductive learning is learning by generalisation. The learner is presented with many training examples about a concept in a specific domain. From these examples, the learner creates and stores a representation of the acquired knowledge, which it can later apply to generalise about unseen instances in the domain. Classification problems may be stated as follows (Quinlan 1988)

**Given:** a collection of entities (and their attributes), whose class membership is known

**Find:** a (set of) classification rule(s) that will assign any entity to its class.

Typically, the learning algorithm is supplied with a set of training examples, specifying the attribute values and class membership for each example, and a representation of the concept to be learned. The learner then constructs a model

of the knowledge it acquires from the training set. Decision trees [Breiman et al 1984], [Quinlan 1986] are commonly used to model this knowledge. For decision tree induction, the learner constructs a decision tree in which the non-leaf nodes are tests on the attributes values, and the leaf nodes are the classes. Determining the class of an instance is then a matter of starting at the root node of the tree, performing the test at the node, and branching to the root node of the selected sub-tree until a leaf node is reached.

The issue of scalability comes into play when attempting to adapt machine learning algorithms to data mining. In machine learning, researchers mostly deal with flat files and algorithms that run in minutes or seconds on a desktop computer. A training set of say 100,000 instances with a few dozen attributes is the beginning of the range "very large" data sets. The database community deals with gigabyte databases. "Very large" to a database practitioner usually means databases (warehouses) of 100GB or larger. However, it is very unlikely that all the data in a database (warehouse) would be mined simultaneously. In practice, pre-processing techniques reduce by orders of magnitude, the size of the data set presented to the data mining algorithm.

For scaling machine learning algorithms, the issue is not one of speeding up a slow algorithm, but rather, one of turning an impracticable algorithm into a practicable one. The crucial issue is not "how fast" you can run on a certain problem, but "how large" a problem you can feasibly deal with [Provost and Kolluri 1999]. From the point of view of complexity analysis, the limiting factor of the data set is the number of examples. A large number of examples introduces problems with

both time and space complexity. Time complexity deals with growth rate of the execution time as the number of examples, number of attributes and number of attribute values increase. Space complexity deals with the amount of storage space required while the algorithm is executing.

This paper discusses the optimal generation of multiple intervals for continuous valued attributes. Most commercially available decision tree learners handle continuous valued attributes by employing binary splitting, which results in two intervals. Even though we do not expect multi-splitting to increase the prediction accuracy of a decision tree, intuitively we expect that decision trees that employ multi-splitting will be more comprehensible to the user. Dynamic programming [Elomaa & Rousu 1999], [Fulton, Kasif & Salzberg 1995] has been proposed as an efficient method for the optimal generation of multiple intervals. This is an attempt to improve on the time and space complexity of machine learning and data mining algorithms when dealing with continuous-valued attributes. This paper proposes the use of heuristic search as an alternative to dynamic programming. Section 2 discusses currently available methods for handling continuous-valued attributes. Section 3 discusses attribute evaluation functions commonly employed in decision tree induction and multi-splitting. Section 4 discusses currently available algorithms for generating optimal multi-splits of numeric value ranges. Section 5 proposes the use of heuristic search to obtain optimal multi-splits of numeric value ranges.

## 2. Handling Continuous-valued Attributes

### 2.1 Attribute Selection in Decision Tree Induction

For decision tree induction, the learner constructs a decision tree in which the non-leaf nodes are tests on the attributes values, and the leaf nodes are the classes. The training set normally consists of both nominal (categorical) and continuous-valued attributes. The most crucial step in decision tree induction is the selection of the attribute on which to branch. An attribute evaluation function is used for this purpose. The evaluation function used, typically measures the reduction in *class impurity* if the attribute were selected for branching, or partitioning the

training set. The attribute which produces the lowest impurity partition is selected for branching [Breiman 1984]. The evaluation function has to be supplied with discrete values for all attributes in the training set. This is not a problem for nominal attributes but, for continuous-valued attributes, this poses a challenge.

A continuous-valued attribute takes on numeric values (integer or real). In general, it is an attribute that has a linearly ordered range of values. Such an attribute is typically handled by partitioning its range into sub-ranges, in order to discretise its values. During decision tree generation, all continuous-valued attributes are discretised prior to attribute evaluation and selection. At each attribute selection step, the discretisation process needs to be performed.

A function commonly used as a measure of impurity is the entropy function

$$H(S) = \sum_{i=1}^m -P(C_i, S) \log_2 P(C_i, S)$$

where  $P(C_i, S)$  is the proportion of examples in  $S$  with class  $C_i$ , and  $m$  is the total number of classes.  $H(S)$  measures the class coherence (or class impurity) in the set  $S$  of examples. The goodness of a split on a discrete-valued attribute  $A$ , is then measured using the average class entropy function ACE, as:

$$ACE(A; \cup_{j=1}^k S_j) = \sum_{j=1}^k \frac{|S_j|}{|S|} H(S_j)$$

where:

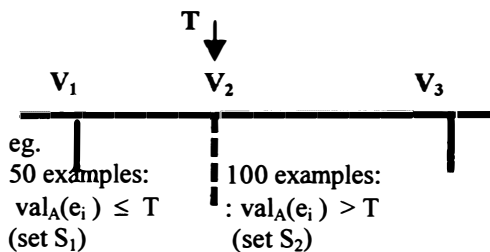
$H(S_j)$  is the impurity in the subset of examples  $S_j$ , having the same value for attribute  $A$ , and  $\cup_{j=1}^k S_j = S$  denotes the  $k$  disjoint subsets of the partition induced by the discrete values for attribute  $A$ , and  $|S_j|$  denotes the size of subset  $S_j$ .

### 2.2 Binary Discretisation of Continuous-valued Attributes

In practice, the discretisation of a continuous-valued attribute may be binary or  $k$ -ary, also known as multi-splitting. For binary discretisation, a *threshold* value,  $T$ , for the continuous valued attribute,  $A$ , is determined, and the test  $A \leq T$  is assigned to the left branch, while  $A > T$  is assigned to the right branch of the tree. The threshold value,  $T$ , is called a cut point. Suppose we are to select an attribute for branching at a node, having a set  $S$



of  $N$  examples. For each continuous-valued attribute,  $A$ , we select the 'best' cut point  $T_A$  from its range of values by evaluating each candidate cut point in the range of values. The examples are first sorted into ascending order by their value on attribute  $A$ , and the mid point between each successive pair of examples, in the sorted sequence, is evaluated as a potential cut point. Figure 1 illustrates the idea.



**Figure 1: Illustration of a Cut point**

$V_1, V_2, V_3, V_4$  are the values of attribute  $A$ .  $val_A(e_i)$  denotes the value of attribute  $A$  for training example  $e_i$ .  $T$  is the cut point currently being evaluated.

For each continuous valued attribute, at most  $N-1$  evaluations must take place. For each evaluation of a candidate cut point,  $T$ , the data are partitioned into two subsets  $S_1$  and  $S_2$  and the partition is evaluated. Using the ACE evaluation function, the partition is evaluated as:

$$ACE(A; S_1 \cup S_2) = \frac{|S_1|}{|S|} H(S_1) + \frac{|S_2|}{|S|} H(S_2)$$

The cut point for which  $ACE(A; S_1 \cup S_2)$ , is minimum, is selected as the best cut point for binary discretisation. Fayyad and Irani (1992) have used this evaluation function for binary discretisation and for multi-splitting.

One of the main problems with this selection criterion is that it is relatively expensive. Although it is polynomial in complexity, it must be evaluated at least  $N-1$  times for each continuous valued attribute, at each attribute selection step. For practical machine learning and especially data mining applications,  $N$  is typically very large. An improvement to this has been proposed by Fayyad and Irani (1992) by observing that only boundary points need be considered as candidate cut points. The notion of a boundary point is captured in the definition below [Fayyad & Irani 1992], [Elomaa & Rousu 1999].

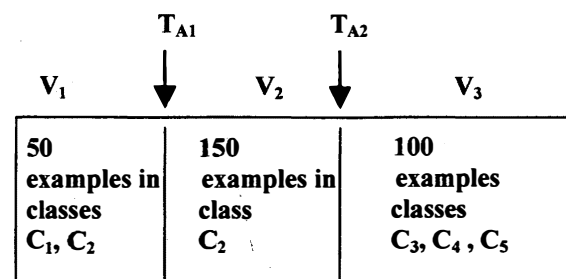
**Definition 1: Boundary Point**

Let  $val_A(e_i)$  denote the value of attribute  $A$  for the training example  $e_i$ . Given a sequence  $S$  of examples, sorted in ascending order, by their values of attribute  $A$ , then:

1. The maximum value in  $S$ , for attribute  $A$ , is a boundary point.
2. A value  $T$  in the range of attribute  $A$  is a boundary point iff in the sequence  $S$ , there exist two examples  $e_1, e_2$ , having different classes, such that:  
 $val_A(e_1) = T < val_A(e_2)$  and there exists no other example  $e'$  in  $S$ , such that :  
 $val_A(e_1) < val_A(e') < val_A(e_2)$ .

In other words,  $T$  is a boundary point if it falls between two consecutive examples that do not fall in the same class. Figure 2 illustrates the idea.

Fayyad and Irani (1992) have proved that, if  $T$  minimises  $ACE(A; S_1 \cup S_2)$  then  $T$  is a boundary cut point. Based on this observation, they have proposed an algorithm that incrementally computes the class entropy  $ACE(A; S_1 \cup S_2)$  for all boundary cut-points. With this scheme, great computational savings are achieved by eliminating the need to consider  $N-1$  candidate cut points. The binarisation algorithm only needs to evaluate  $B$  ( $B \ll N$ ) boundary points in order to select the best cut-point.



**Figure 2: Illustration of a boundary point.**

$T_{A2}$  is a boundary point as it does not split any examples in the same class.  $T_{A1}$  is not a boundary point.

**2.3 Generating Multiple Numeric Intervals**

In multi-splitting, the range of a continuous-valued attribute is discretised into  $k$  intervals,  $k \geq 2$ . Algorithms for multi-splitting attempt to obtain the best  $k$ -ary split of the numeric range, using some evaluation function.

Several evaluation functions for selecting the best  $k$ -partition for a continuous valued

attribute have been reported in the literature. Fayyad and Irani (1992) have shown that, for binary discretisation, the average class entropy function  $ACE(A; S_1 \cup S_2)$ , is convex downward between boundary points and will therefore have a minimum value at a boundary point. Elomaa and Rousu (1999) have proved that, in the general case,  $ACE(A; \cup S_i)$  will minimise at boundary points. They have analysed a class of cumulative, useful and well-behaved evaluation functions and proved that these functions minimize at boundary points. These functions are discussed in the next section.

The use of boundary points as cut points in multi-splitting has been investigated by Elomaa and Rousu (1999). They propose that prior to the discretisation of a numeric attribute, the training examples should be divided into  $B$  blocks after sorting. The block borders are the boundary points,  $\{T_1, \dots, T_B\}$  for the numeric value range of the attribute,  $A$ . If, in isolation, attribute  $A$  has predictive power, ie. if its values directly correlate with the example set's classification, then the number of blocks,  $B$ , satisfies  $B \ll N$  [Fayyad & Irani, 1992]. In practical machine learning and data mining a multi-split of arity  $B$  could still be too large and impractical, in the sense that it would result in incomprehensible trees. It then becomes necessary to determine the best  $k$ -split where  $k < B$ .

### 3. Attribute Evaluation Functions

#### 3.1 Well-behaved and Useful Functions

##### Definition 2: Convex Downward Functions

Breiman et al (1984) have established that an attribute evaluation function  $F$  should be strictly concave (convex downward) ie. should be twice differentiable and the second derivative should be negative. They have further established that for any tree node  $t$ , and any split  $s$ , if the evaluation function  $F$  is used, then the decrease in impurity,  $\Delta I(s, t) = I(t) - F(s, t) \geq 0$ , where  $I(s)$  measures the impurity before splitting. In other words, the impurity as measured by  $\Delta I(s, t)$ , can never increase due to splitting.

##### Definition 3: Cumulative Functions

Let  $\cup S_i$  be a partition of an arbitrary example set  $S$ . An evaluation function  $F$  is cumulative if there exists a function  $f$  such that

$$F(A; \cup S_i) = c \cdot \sum f(S_i)$$

where  $c$  is an arbitrary coefficient whose value may depend on the whole data set  $S$  but not on its partitions.

##### Definition 4: Useful and Well-behaved Functions

Let

1.  $\tau = \{T_1, \dots, T_B\}$  be the boundary points in an example set  $S$ .
2.  $F(A; T)$  denote the value of the evaluation function  $F$ , for the binary partition that is defined by cut point  $T$ .
3.  $F(A; T_1, \dots, T_{k-1})$  denote the value of the evaluation function  $F$ , for the  $k$ -partition defined by the cut points  $\{T_1, \dots, T_{k-1}\}$

An evaluation function  $F$ , is useful, in binary partitioning, if there exists a value  $T$  in  $\tau$  such that  $T$  minimises  $F(A, T)$ . An evaluation function is well-behaved if for any  $k$ ,  $1 \leq k \leq B$  there exists at most  $k$  values  $\tau' \subseteq \tau$ , such that  $\tau'$  minimises  $F(A; T_1, \dots, T_{k-1})$ .

If a useful function is also cumulative, then it is also well-behaved with respect to multi-way partition evaluation. Elomaa & Rousu (1999) have shown that, for any cumulative and useful function  $F$ , there exists a partition of an arbitrary example set, such that it minimises the value of  $F$  with the corresponding cut points being situated at boundary points.

#### 3.2 Entropy-based Evaluation functions

The most commonly used attribute evaluation functions build upon impurity measures [Breiman et al 1984], which are functions that try to estimate the class coherence in a given set of examples. The average entropy function  $ACE(A; \cup S_i)$  may be used to evaluate partitions in multi-splitting as it is well-behaved. Fayyad and Irani (1992), have used this evaluation function in a scheme that employs recursive binarisation.

The information gain function,  $IG$ , of ID3 [Quinlan 1993] and Gain Ratio function,  $GR$ , of C4.5 [Quinlan 1996] are also useful in partition evaluation. The information gain is defined as:

$$IG(A; \cup S_i) = H(S) - ACE(A; \cup S_i)$$

The intent here is to maximise the value of the function. Note that  $IG$  measures the decrease in impurity as discussed in section 3.1. The function  $IG$ , tends to favour excessively multi-

valued nominal attributes and large arity multi-splits. To correct this shortcoming, Quinlan suggested dividing the IG measure of a partition by the term:

$$g(A; \cup S_i) = - \sum \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

The resulting evaluation function is known as the gain ratio GR:

$$GR(A; \cup S_i) = IG(A; \cup S_i) / g(A; \cup S_i)$$

Even though GR is not convex, its optimal partitions are defined on boundary points. However, since GR is not cumulative, it cannot be employed in incremental evaluation schemes [Elomaa & Rousu 1999].

Lopez de Mantras (1991) has proposed the *Normalised Distance Measure*, ND, as an alternative to the IG and GR evaluation functions. The measure can be expressed in terms of IG as:

$$ND(A; \cup S_i) = 1 - IG(A; \cup S_i) / g(A; \cup S_i)$$

where

$$g(A; \cup S_i) = \sum_{i=1}^k \sum_{j=1}^m \frac{M(j, S_i)}{|S|} \log_2 \frac{M(j, S_i)}{|S|}$$

k is the partition arity, m is the number of classes and M(j, S<sub>i</sub>) is the number of examples of class j in S. The intent is to minimise the value of ND(A;  $\cup S_i$ )  $\in [0,1]$  or equally maximise 1 - ND(A;  $\cup S_i$ ). As with GR, ND is not convex and it is not cumulative, but its optimal partitions are on boundary points. [Elomaa & Rousu 1999].

The Gini index (of diversity) or quadratic entropy is defined as:

$$GI(A; \cup S_i) = \sum \frac{|S_i|}{|S|} \text{gini}(S_i)$$

where gini is the impurity measure

$$\text{gini}(S) = \sum_{j=1}^m P(C_j, S) (1 - P(C_j, S))$$

The GI function is known to be convex and therefore useful. Since the function is also cumulative, it is well-behaved.

### 3.3 Other Evaluation functions

Stemming from the work of Wallace and Freeman (1987) on the Minimum Description Length Principle (MDL) and Rissanen (1989) on the Message Length Principle (MML), several researchers have explored MDL/MML decision tree learning. Here, the intent is to minimise the coding length of examples. A cost function is used to evaluate attributes and the attribute which gives the least rise to the total coding length of examples is chosen for the evolving tree. The cost function used in MDL is useful and well-behaved [Elomaa & Rousu 1999].

In learning decision trees of limited depth eg. one and two-level decision trees (Iba & Langley 1992; Holte 1993; Elomaa 1994; Auer, Holte & Maas 1995), the process entails optimal multi-splitting of a numeric domain. The evaluation function to be optimised is the Training Set Error, TSE: the number of training instances falsely classified by the decision tree. The evaluation function TSE is useful and well-behaved [Elomaa & Rousu 1999].

## 4. Optimal multi-splitting of numeric value ranges

### 4.1 Recursive Binarisation

Fayyad and Irani (1992) have proposed a multi-splitting algorithm based on binary discretisation. The algorithm is applied recursively, during tree construction, always selecting the best cut point from all boundary points. As in binary discretisation, the measure ACE(A; S<sub>1</sub> ∪ S<sub>2</sub>) for average class entropy is used to select the best cut point. A criterion is applied to decide when to refrain from further binary partitioning of a given interval. Incremental evaluation of ACE(A; S<sub>1</sub> ∪ S<sub>2</sub>), further enhances the algorithm's efficiency.

### 4.2 Incremental Evaluation

Fulton, Kasif & Salzberg (1995) have proposed an algorithm for efficiently finding optimal multi-splits for a class of evaluation functions. This algorithm examines all potential cut points within the example sequence in order to obtain the optimal multi-

split by a cumulative evaluation function. They give the recurrence by which the impurities of the candidate cut point can be calculated from those for lower arity partitions, during the process. Hence, given a value  $k$ , the algorithm can choose in time  $O(\ln^2)$ , the partition with the lowest impurity, from among all those that have at most  $k$  intervals. Elomaa and Rousu (1999) have extended this scheme to a faster algorithm. Instead of evaluating the cut points between every pair of examples, only boundary points are evaluated. This is accomplished by pre-processing the data into blocks, where a block is defined by boundary points. Pre-processing of the data into blocks can be done in linear time, since it requires a single pass through the example set. Any of the evaluation functions discussed in the last section may be used to evaluate partitions.

Elomaa and Rousu have shown that, given blocks  $i$  through  $j$  in the sorted sequence of examples, we may denote the impurity that results when the blocks are partitioned into  $k$  intervals by  $impurity(k, i, j)$ . The recurrence for impurity calculation is then given as:

$$\begin{aligned}
 impurity(k, i, j) = & \\
 \min \{ & impurity(k-1, i, i) + impurity(1, i+1, j) \\
 k-1 \leq i < j & \quad \quad \quad \text{if } k \leq j \\
 infinity & \quad \quad \quad \text{otherwise}
 \end{aligned}$$

Using dynamic programming, they have proposed an  $O(kB^2)$  algorithm which can find an optimal multi-partition of at most arity  $k$ , where  $B$  is the number of blocks in the range. In comparison, with binary splitting using blocks, the time to find the optimal binary split is  $O(B)$ .

## 5. Heuristic Search for Optimal Multi-splits

### 5.1 Heuristic Evaluation functions

As an alternative to dynamic programming, this paper proposes heuristic search. Heuristic search requires the use of a state evaluation function  $f(n) = g(n) + h(n)$  where,  $n$  is any state encountered in the search,  $g(n)$  is the cost of  $n$  from the start state, and  $h(n)$  is the heuristic estimate of going from the  $n$  to the goal state. A search algorithm is *admissible* if, for any graph, it always terminates in the

optimal solution path whenever a path from the start state to the goal state exists. Heuristic search also employs a function  $f^*(n) = g^*(n) + h^*(n)$ , where  $g^*(n)$  is the cost of the shortest path from the start state to node  $n$ , and  $h^*(n)$  is the actual cost of the shortest path from  $n$  to the goal state.  $f^*(n)$  is then the actual cost of the optimal path from the start to the goal state. If an algorithm uses best-first-search with an evaluation function  $f(n) = g(n) + h(n)$  where  $h(n) \leq h^*(n)$ , then the algorithm is *admissible*. A heuristic is *monotonic*, or locally admissible, if it consistently finds the minimal path to each state it encounters in the search. A heuristic function  $h$  is *monotone* if

1. for all states  $n_i$  and  $n_j$ , where  $n_j$  is a descendant of  $n_i$ ,  $h(n_i) - h(n_j) \leq cost(n_i, n_j)$ , where  $cost(n_i, n_j)$  is the actual cost of going from state  $n_i$  to  $n_j$ .
2. the heuristic evaluation of the goal state is zero, ie.  $h(goal) = 0$ .

It is argued in the next section that the class of evaluation functions  $F$  can be used in heuristic search for optimal multi-splits.

### 5.2 Heuristic Search for Optimal Multi-splits

For heuristic search, the problem of finding the optimal multi-split may be stated as follows:

‘ Find that combination of at most  $k$  cut points  $\tau' \subseteq \tau$ , where  $\tau = \{T_0, \dots, T_B\}$ , that minimises the impurity measure  $F(A; \cup S_i)$ , for the partition  $\cup S_i$  induced by  $\tau'$  ‘

A state is represented by the cut points that define the partition induced by the cut points. The initial state is one where no splitting has taken place, therefore it is represented by the cut points  $\{T_0, T_B\}$ . The descendant states are successively generated by including one of the remaining boundary points  $\{T_0, T_{B-1}\}$ . Each of these states defines a unique partition. Given the cumulativity property for the evaluation function  $F$ , and the fact that splitting can never increase the impurity, we can conclude that  $F(A; \cup S_j) \leq F(A; \cup S_i)$ , where  $j$  is a descendant state of state  $i$ .

The state evaluation function to be used should measure the amount of impurity to be removed in order to reach the goal state. The split that moves us closest to the goal is then chosen. We can therefore define  $h(n)$  as:  $h(n) = F(A; p:n) - F(A; p:goal)$  where  $F(A; p:n)$  denotes the evaluation of the

function  $F$  for the partition at state  $n$ . Here, 'goal' denotes the state for which a split of arity  $B$  has been reached.  $g(n)$  can be used to measure the arity of the split. When employed with the best-first search algorithm [Pearl 1984] the proposed heuristic is *monotonic* and *admissible* since it will terminate on the optimal path to the split which has arity  $B$ , with  $h(\text{goal}) = 0$ .

### 5.3 Time and Space Complexity for Heuristic Search

For the search scheme discussed in the last section, we can see that all the states at level  $k$  in the search tree define partitions of arity  $k$  or less. The best partition of arity  $k$  or less will therefore be found at level  $k$  of the search tree. For small  $k$ , the optimal multi-split will be found high up in the tree, while for large  $k$ , the optimal multi-split will be much lower down. At each level  $k$  of the tree,  $B - k$  partitions are generated and evaluated. The asymptotic time complexity is therefore  $O(kB)$ . In the worst case, when all possible arities must be generated, this becomes  $O(B^2)$ . The pre-processing of data into blocks as proposed by Elomaa & Rousu (1999) is illustrated in Figure 3. This eliminates the need to examine each individual example in  $S$ .

The space requirements for the algorithm are very modest. The open list can never be larger than  $B$ , the maximum number of descendant states. The closed list will hold only the path to the goal, at most  $k$  states. A state is simply represented by the cut points that define its partition. Global data structures are used to store the data indicated in Figure 3, for each continuous-valued attribute.

Block $B_1$ ( $v_1..v_i$ )	Block $B_2$ ( $v_j..v_p$ )	Block $B_b$ ( $v_r..v_z$ )
<b>class freq.</b> $C_1$ freq( $C_1$ ) $C_2$ freq( $C_2$ ) $C_3$ freq( $C_3$ )  $C_m$ freq( $C_m$ )	<b>class freq.</b> $C_1$ freq( $C_1$ ) $C_2$ freq( $C_2$ ) $C_3$ freq( $C_3$ )  $C_m$ freq( $C_m$ )	<b>class freq.</b> $C_1$ freq( $C_1$ ) $C_2$ freq( $C_2$ ) $C_3$ freq( $C_3$ )  $C_m$ freq( $C_m$ )

**Figure 3: Pre-processing data into blocks**  
For each block, the values for the attribute and the class frequencies are recorded.

## 6. Conclusions

Well-behaved functions are clearly the only reasonable ones to use in multi-splitting [Fayyad & Irani 1992], [Breiman 1996]. The discussion in section 5 of this paper has shown that, when well-behaved functions are used in multi-splitting, well understood and efficient methods for obtaining optimal solutions can be used. Multi-splitting is definitely slower than binarisation, and it does not result in improved prediction accuracy. However, intuitively, it would appear that multi-splitting produces more compact and comprehensible trees than binary splitting. Theoretically, it would appear that heuristic search performs better than dynamic programming, for generation of optimal multi-splits. An experimental system is currently being implemented at Vista University, in order to perform empirical studies of both methods.

## 7. Acknowledgements

Acknowledgements are due to Dr. R. Angelov and Dr. R. Kasonga, Department of Mathematics & Statistics, Vista University, for many useful insights given on the mathematical and statistical aspects of this research.

## 8. References

1. Auer P., Holte R.C., & Maas W., (1995), 'Theory and Application of Agnostic PAC-learning with small decision trees'. In A. Prieditis & S. Russel (eds), Machine Learning: Proceedings of the Twelfth International Conference (pp.244-251), San Francisco CA: Morgan Kauffman.
2. Breiman L, Friedman J, Olshen R, & Stone C., (1984), 'Classification and Regression Trees', Wadsworth.
3. Breiman L., (1996), 'Technical Note: Some properties of Splitting Criteria', Machine Learning, vol. 24, no. 1, pp41-47, July 1996
4. Cartlett, J., (1991), 'Megainduction: A test flight', Proc. of the Eighth Workshop on Machine Learning', Morgan Kaufmann, 1991, pp 596-599.
5. Elomaa T. and Rousu J., (1999), 'General and Efficient Multisplitting of Numerical Attributes', Machine Learning, vol 36, no.

- 3, sept 1999.
6. Fayyad U. M., (1996), 'Data Mining and Knowledge Discovery: Making Sense Out of Data', IEEE Expert, oct. 1996, pp 20 - 25.
  7. Fayyad U.M and Irani K.B., (1992), 'On the Handling of Continuous-valued Attributes in Decision Tree Generation', Machine Learning, 8, pp 87-102
  8. Fulton T., Kasif S., & Salzberg S., (1995), 'Efficient Algorithms for Finding Multi-way Splits for Decision Trees', In A. Prieditis & S. Russel (eds), Machine Learning: Proceedings of the Twelfth International Conference (pp.244-251), San Francisco CA: Morgan Kauffman.
  9. Holte R.C. (1993), 'Very Simple Classification Rules perform well on most commonly used data sets', Machine Learning, 11, pp63-90.
  10. Iba W. F., & Langley P. (1992), Induction of one-level decision trees', In. D. Sleeman & P. Edwards (eds), Machine Learning: Proc. Ninth International Workshop (pp.233-240), San Francisco CA, Morgan Kaufman.
  11. Lopez de Mantras R. (1995), 'A distance-based attribute selection measure for decision tree induction', Machine Learning, 6, pp81-92.
  12. Pearl, J. (1984), 'Heuristics: Intelligent Search Strategies for Computer Problem Solving', Addison-Wesley.
  13. Provost, F. and Kolluri, V., (1999), ' A Survey of Methods for Scaling Up Inductive Algorithms', Data Mining and Knowledge Discovery, vol3, no.2, june 1999, pp 131-169.
  14. Quinlan J.R., (1996), 'Improved use of Continuous Valued Attributes in C4.5', Journal of Artificial Intelligence Research 4, 1996, pp 77-90.
  15. Quinlan J.R., (1993), 'C4.5: Programs for Machine Learning', Morgan Kaufman, San Francisco, 1993.
  16. Quinlan J.R., (1988), 'Decision Trees and Multi-Valued Attributes', Machine Intelligence 11, 1988, pp 305 - 318.
  17. Quinlan J.R. (1986), 'Induction of Decision Trees', Machine Learning 1, 1986, pp 81- 106.
  18. Rissanen J., (1989), 'Statistical Complexity in Statistical Enquiry', River Edge, NJ: World Scientific.
  19. Shannon C.E. & Weaver W. , (1962), 'The Mathematical Theory of Communication', University of Illinois Press, Urbana, 1962.
  20. Uitgoff, P. (1989), 'Incremental Induction of Decision Trees', Machine Learning,4, 1989, pp 161-186.
  21. Wallace C.S., & Freeman P.R, (1987), 'Estimation and Inference by Compact Coding', Journal of the Royal Statistical Society(B), 49, pp240-265.

