Special Issue: SAICSIT '99
## Contents

**Preface**  
P. Machanick ............................................................... 1

### Research Articles

Active Learning: Issues and Challenges for Information Systems and Technology  
RD Quilling, GJ Erwin and O Petkova ........................................ 5

A Generic Modelling Framework for Interactive Authoring Support Environments  
Paula Kotze ......................................................................... 15

O Petkova and JD Roode .......................................................... 26

An Information-Theoretic Semantics for Belief Change  
T Meyer .................................................................................. 33

A Complexity Metrics Model for Software Correction  
A Törn, T Andersson and K Enholm ........................................... 40

A Conceptual Design for High-Volume Data Processing of Warehouse Database into Multidimensional Database  
Paisarn Trakulsuk and Vichit Avatchanakorn ................................ 49

A Pragmatic Approach to Bitemporal Databases: Conceptualization, Representation and Visualisation  
Chiyaba Njovu and WA Gray ..................................................... 58

A Building Recognition System  
SP Levitt and B Dwolatzky ......................................................... 68

Computer Programming and Learning to Write  
John Barrow ............................................................................. 77

Co-operating to Learn using JAD Technologies  
TA Thomas ............................................................................... 87

Critical Success Factors for the Implementation of DSS at a Selection of Organisations in Kwazulu/Natal  
URF Averweg and GJ Erwin ..................................................... 95

Enhancing the Predictability of Two Popular Software Reliability Growth Models  
Peter A Keiller and Thomas A Muzzuchi ..................................... 105
Generalised Unification of Finite Temporal Logic Formulas
Scott Hazelhurst .......................................................... 110

Harmonizing Global Internet Tax: A Collaborative Extranet Model
E Lawrence and B Garner .................................................. 119

Improving Object Oriented Analysis by Explicit Change Analysis
Lui Yu, Siew Chee Kong, Yi Xun and Miao Yuan .................... 128

Reconciling the Needs of New Information Systems Graduates and Their Employers in Small, Developed Countries
Rodney Turner and Glenn Lowry ......................................... 136

Shortest Delay Scheduling Algorithm for Lossless Quality Transmission of Stored VBR Video under Limited Bandwidth
Fei Li, Yan Liu, Jack Yiu-Bun Lee and Ishfaq Ahmad .................. 146

Software Croma Keying in an Immersive Virtual Environment
Frans van der Berg and Vali Lalioti ..................................... 155

Some Automata-Theoretic Properties of $\cap$-NFA
Lynette van Zijl and Andries P J van der Walt ....................... 163

Statistical Analysis of an Automated Computer Architecture Learning Environment
JT Waldron, J Horgan and G Keogh ...................................... 168

The CILT Multi-Agent Learning System
Herna L Viktor ..................................................................... 176

The Development of a Generic Framework for the Implementation of Cheap, Component-Based Virtual Video-Conferencing System
Soteri Panagou and Shaun Bangay ......................................... 185

The Role of Experience in User Perceptions of Information Technology: An Empirical Examination
Meliha Handzic and Graham Low ......................................... 194

What are Web Sites Used for: Cost Savings, Revenue Generating or Value Creating?
Man-Ying Lee ....................................................................... 201

New Ideas Papers
Approaches to Video Transmission over GSM Networks
Bing Du and Anthony Maeder ................................................ 210

From Information Security Baselines to Information Security Profiles
Rossouw von Solms and Helen van der Haar ......................... 215

Grounded Theory Methodology in IS Research: Glaser versus Strauss
J Smit ................................................................................. 219

Introducing a Continuum of Abstraction-Led Hierarchical Search Techniques
Robert Zimmer and Robert Holte ........................................... 223
Multimedia as a Positive Force to Leverage Web Marketing, with Particular Reference to the Commercial Sector
Stan Shear .......................................................... 229

Understanding HCI Methodologies
Peter Warren ......................................................... 234

Electronically Published Papers

Experience Papers
A Java Client/Server System for Accessing Arbitrary CANopen Fieldbus Devices via the Internet
Dieter Bühler, Gerd Nusser, Gerhard Gruhler and Wolfgang Küchlin ........................................ 239

An Object-Oriented Framework for Rapid Client-side Integration of Information Management Systems
Ralf-Dieter Schimkat, Wolfgang Küchlin and Rainer Krautter ...................................................... 244

Distributed Operating Systems: A Study in Applicability
Jürgen Prange and Judith Bishop ............................................. 249

Formal Verification with Natural Language Specifications: Guidelines, Experiments and Lessons so far
Alexander Holt .......................................................... 253

Introducing Research Methods to Computer Science Honours Students
Vashti Galpin, Scott Hazelhurst, Conrad Mueller and Ian Sanders .......................... 258

Visualising Eventuality Structure
ST Rock .......................................................... 264

Electronically Published Papers
SAICSIT'99
South African Institute of Computer Scientists and Information Technologists
Annual Research Conference 17-19 November 1999
Prepare for the New Millennium
Is there life after y2k?
Mount Amanzi Lodge, Hartebeespoort, South Africa

Sponsors

http://www.cs.wits.ac.za/
PHDS
research/PHDS.html

Think different.
Running SAICSIT’99, the annual research conference of the South African Institute for Computer Scientists and Information Technologists, has been quite an experience.

SAICSIT represents Computer Science and Information Systems academics and professionals, mainly those with an interest in research. When I took over as SAICSIT president at the end of 1998, the conference had not previously been run as an international event. I decided that South African academics had enough international contacts to put together an international programme committee, and a South African conference would be of interest to the rest of the world.

I felt that we could make this transition at relatively low cost, given that we could advertise via mailing lists, and encourage electronic submission of papers (to reduce costs of redistributing papers for review).

The first prediction turned out to be correct, and we were able to put together a strong programme committee.

As a result, we had an unprecedented flood of papers: 100 submitted from 21 countries. As papers started to come in, it became apparent that we needed more reviewers. It was then that the value of the combination of old-fashioned networking (people who know people) and new-fashioned networking (the Internet) became apparent. While the Internet made it possible to convert SAICSIT into an international event at relatively low cost, the unexpected number of papers made it essential to find many additional reviewers on short notice. Without the speed of e-mail to track people down and to distribute papers for review, the review process would have taken weeks longer, and it would have been much more difficult to track down as many new reviewers in so little time.

Even so, the number of referees who were willing to help on short notice was a pleasant surprise.

The accepted papers cover an interesting range of subjects, from management-interest Information Systems, to theoretical Computer Science, with subjects including database, Java, temporal logic and implications of e-commerce for tax.

In addition, we were very fortunate in being able to invite the president of the ACM, Barbara Simons as a keynote speaker. Consequently, the programme for SAICSIT’99 should be very interesting to a wide range of participants.

We were only able to find place in the proceedings for 36 papers out of the 100 submitted, of which only 24 are full research papers. While this number of papers is in line with our expectation of how many papers would be accepted in each category, we did not have a hard cut-off on the number of papers, but accepted all papers which were good enough, based on the reviews. Final selection was made by myself as Programme Chair, and Derrick Kourie, as editor of the South African Computer Journal. Additional papers are published via the conference web site.

We believe that we have put together a quality programme, and hope you will agree.

Acknowledgments

I would like to thank the South African Computer Journal production team, Andries Engelbrecht and Herna Viktor, respectively from the Department of Computer Science and Informatics, University of Pretoria, for their work on producing the proceedings.

The reviewers listed overleaf did an excellent job: many wrote very detailed reports, sometimes after being called in on very short notice. Inevitably, there were some glitches resulting from the unexpected workload, but the buck stops with the programme chair: I promise to do better next time.

I would also like to thank my own department for putting up with the extra work and expense that running a conference entails. I tried not to burden them with too much extra work, but our secretaries, Zalm Gowar and Leanne Reddy, inevitably had to take on some extra work. John Ostrowick provided valuable assistance with design of our web pages and call for papers poster. Carol Kernick, who handles our finances and membership records, did a fine job of keeping up with the demands of the conference.

Finally, I would like to thank our sponsors, whose contribution made this conference been possible:

- PricewaterhouseCoopers – sponsored generous prizes and the conference banquet
- National Research Foundation (NRF) – provided financial support
- University of the Witwatersrand – provided financial support
- Programme for Highly Dependable Systems, University of the Witwatersrand – provided financial support
- Standard Bank – provided financial support
Editorial

- Apple Computer – provided equipment for the conference
- Qualica – provided technical support including helping with the conference web site

Web Site

For more information about SAICSIT, including a pointer to the conference site, see <http://www.saicsit.org.za>.

Referees

- Department of Computer Science, University of Pretoria
  - Derrick Kourie
  - Bruce Watson
  - Vali Lalioti
  - Andries Engelbrecht
  - Ivan Mphahlele
- German National Research Center for Information Technology - GMD
  - Gernot Goebbels
- School of Math Stat and IT, Natal University Pietermaritzburg
  - Peter Warren
- Graduate School of Business, University of Cape Town
  - Kurt April
- CSIR
  - James Jardine
- Department of Computer Science, Rhodes University
  - Shaun Bangay
  - Peter Clayton
  - John Ebden
  - Richard Foss
  - George Wells
  - Peter Wentworth
- Information Technology Division, Rhodes University
  - Caro Watkins
- Department of Information Systems, Rhodes University
  - Brenda Mallinson
  - Dave Sewry
- Department of Informatics, University of Pretoria
  - Carina de Villiers
  - Herna Viktor
  - Niek du Plooy
  - Elsie van Rooyen
  - Machdel Matthee
  - Alan Abrahams
  - Jackie Phahlamohlaka
- School of Information Technology & Engineering, University of Ottawa
  - Stan Szpakowicz
  - Dwight Makaroff
- Department of Information Systems, Victoria University
  - Glenn Lowry
  - Peter Shackleton
  - Tas Adam
  - Alastair Wallace
  - Stephen Burgess
  - Dave Burgess
  - Julie Fisher
  - Jerzy Lepa
  - Geoff Sandy
  - Rod Turner
  - Alastair Wallace
  - Andrew Wenn
- Peninsula School of Computing and Information Technology, Monash University
  - Ainslie Ellis
- School of Information Management and Systems, Monash University
  - Angela Carbone
- School of Computer Science & Software Engineering, Monash University
  - Dianne Hagan
  - Judy Sheard
- School of Management Information Systems, Deakin University
  - John Lamp
  - David Mackay
  - Philip Joyce
Editorial

– Alan Chalmers

• Department of Systems and Computer Science, Howard University, Washington, D.C
  – Todd Shurn
  – John Trimble

• Department of Computer Science, University of Cape Town
  – Sonia Berman

• King’s College, London
  – Peter Wood

• Department of Information Systems, University of Cape Town
  – Paul Licker
  – Steve Erlank
  – Alemayehu Molla

• Department of Department of Mathematics and Applied Mathematics, University of Cape Town
  – Ingrid Rewitzky
  – Peter Jipsen
  – Renkuan Guo

• Department of Information Systems, University of the Western Cape
  – Andy Bytheway

• Department of Electrical Engineering and Computer Science, The George Washington University, Washington, DC
  – Rachelle Heller

• School of Biophysical Sciences and Electrical Engineering, Swinburne University of Technology
  – Ian Macdonald

• Electrical and Electronic Systems Engineering, Queensland University of Technology
  – Anthony Maeder

• Computer Science Department, Åbo Akademi
  – Aimo Törn
  – Kaisa Sere
  – Mats Aspnas

• Division of Informatics, University of Edinburgh
  – Graeme Ritchie
  – Arturo Espinosa-Romero
  – Jon Oberlander

SART/SACJ, No 24, 1999
Editorial

- Chris Brew
- Alexander Holt
- Department of Computer Science and Engineering, Helsinki University of Technology
  - Tommi Junnila
  - Nisse Husberg
- Technical University of Budapest, Department of Measurement and Information Systems
  - Andras Pataricza
- NASA-Goddard Space Flight Center
  - Nigel Ziyad
- Department of Computer Science & Information Systems, University of Port Elizabeth
  - Janet Wesson
  - Leon Nicholls
- Computer Technology Department, Indiana University Purdue University Indianapolis
  - Tim Price
- Department of Electrical Engineering and Computer Science, The University of Michigan
  - Trevor Mudge
- Department of Electrical Engineering and Computer Sciences, University of California, Berkeley
  - David Forsyth
- School of Computer Science, University of Birmingham
  - Mark Ryan
- Faculty of Mathematics, Computer Science, Physics & Astronomy, University of Amsterdam
  - Carlos Areces
- Department of Computer Science, Universidade do Vale do Rio dos Sinos - UNISINOS Rio Grande do Sul
  - Marcelo Walter
- School of Information Technology & Mathematical Sciences, University of Ballarat
  - Binh Pham
A Conceptual Design for High-Volume Data Processing of Warehouse Database into Multidimensional Database

Paisarn Trakulsuka\textsuperscript{a} and Vichit Avatchanakorn\textsuperscript{b}

Assumption University, Bangkok, Thailand, \texttt{paisarn.t@thaiairways.co.th, vichit@au.ac.th}

Abstract

This paper presents a conceptual model of analyzing huge amount of information from a data warehouse into Multidimensional Database (MDDB) using multidimensional data (MDD) model. The major advantages of this model are: 1) it shows the input-output design phase of MDDB and 2) it reduces workload in high-volume data processing. This conceptual model shows how data catalogs can be used to generate the MDD forms (Dimensions, Variables or data values, and Relative dimensions) of a multidimensional cube for On-line Analytical Processing (OLAP). The model may also be expressed by fact schemes that integrate relative information and external sources, as well as extends the algorithm to build MDD patterns in standard SQL form. This schema is based on the basic elements of a MDD model. This model uses three symbols: center nodes, branch nodes, and rectangles. The center node represents the fastest varying dimension, branch nodes represent associate dimension attributes, and rectangles represent data value indexing and measurements.

Keywords: Databases, Data Model, Graph Model, Data Warehouse, OLAP

Computing Review Categories: H2

1 Introduction

Data warehouses collect, integrate, and clean a wide variety of external data sources and base data into a single database. A data warehouse is used primarily for decision support system (DSS) and management information system (MIS). Thus, it may contain many data entities to speed up query processing for different resulting data arrays [2].

Metadata, an important aspect of a data warehouse, collects various definitions of data elements in warehouse databases. Metadata collates conceptual or logical data structures of both objects and entities, together with all the relationships of the data elements within the database [6]. In this paper, metadata will support a Multidimensional Data (MDD) model to generate its own multidimensional cube.

A MDD model is a pattern of configuring large masses of data into patterns meaningful for management analysis and decision-making. It is composed of dimension, data values or variables, and relative dimension. A dimension is a group of attributes that is chosen by management to arrange the data (How many suppliers of shrimp is needed during the high flying season compared to during the low flying season?). Relative dimensions are the attributes in each dimension (Who are the suppliers? What size, color, of shrimp is needed at the galley?) Data values or variables refer to units of quantity or value measurements of the data. (How many kinds of dishes use shrimp? How many percent of passengers do not choose shrimp? How much less passengers fly during the off season?)

Previous research shows that a warehouse database leans on a multidimensional data model to generate data analysis. In this paper, the function is extended to initiating the data warehouse routine of transforming the data into a MDD model according to user specifications.

The accuracy of conceptual design is vital in building an information system so that it responds fully to user requirements. This paper presents a graph model that shows a routine for processing voluminous data from various sources into groups of related dimensions and associated data values. The focus of this paper is 1) to create a MDD model to meet user requirements more accurately, 2) to design a routine that automatically creates a database format, 3) to speed up the MDDB's input-output design phase, and 4) to minimize human error in the process. Using the graph theory this paper creates a MDD model based on relational format concepts of data processing.

Currently, the relational database is the standard operating format all over the world for high volume data processing. This format requires intensive man-hours to shape the mass of data into forms meaningful to management. At Thai Airways, an average of 2,016 man-hours per year is consumed for this routine, too slow to meet the demands of high-speed decision-making. This paper proposes MDDB, an automated process of MDD model creation that identifies, corrects, and screens out errors in a large mass of data. With MDDB, the procedure takes only an hour, and never again. When factored against the number of man-hours spent in all companies all over the world, this represents incredible savings of time, effort, and money.

The graph model describes how data entities are collected from a database through three symbols: center nodes, branch nodes, and rectangle nodes. A primary attribute is presented in a center-node. Foreign keys or char-
acter format attributes, and associated dimensions are presented in branch-nodes. Numeric format attributes are data values or variables presented in rectangle-node.

Curved lines show MDD relationships. Dashes show links between dimensions that support the OLAP drill-down concept. In complex data, the lines overlap to rotate and compare information. The finished B-tree structure fact scheme is reorganized into MDD models of a hypercube in terms of a standard SQL form supported by algorithms.

2 Background and Literature Review

Data warehousing has three phases that start from raw data and ends in knowledge extracted from the database. First, data is segmented according to some criteria. Second, data is cleaned, then transformed. Data is then stored in data warehouse with distinct structures.

When data is stored in relational forms at the data access layer, it is voluminous because of storage at the lowest level of granularity. It is necessary to produce summarized data in MDDB in order to access the data volumes quickly. The two types of data summaries in MDDB are lightly summarized data, which is distilled from low levels, and highly-summarized data, which is compact and easily accessible.

To rebuild the lowest detailed data into summarized data, data accuracy must be declared to prevent error contamination. There are several techniques to acquire and integrate information from heterogeneous sources and to query very large database efficiently. The E/R approach is presented as a graph model for data warehousing called Dimension Fact Model. This proposes a semi-automated routine based on pre-existing Entity/Relationship schemes. The E/R model is used widespread as a conceptual formalism to provide standard documentation for relational information systems. Many efforts have been done to use the E/R scheme as input for designing non-relational database.

A decision tree (name, date) is a method of decision-making that is applicable when complex branching occurs in a structured decision process. A decision tree is particularly useful when it is essential to keep a string of decisions in a particular sequence. A decision tree allows description of conditions and actions on branch nodes. Thus, it was applied in this research to negotiate the complex routes of data warehouse structuring.

Most research about data warehousing focuses on control of multi-dimensional views within data warehouse environments such as maintenance techniques, view technique designs, and multiple views consistency. To reduce maintenance costs, Nam Huyn proposed a process of preserving the views and keeping them consistent without updating the warehouse. The process introduces an algorithm to generate SQL queries, the answers of which determine if a view can be maintained in a given situation.

The process presents a framework of view design that achieves the best combination of good query performance and low view maintenance. A framework is presented to highlight issues of materialized view design in a distributed data warehouse environment. For multiple views consistency (MVC), Yue Zhuge, et al., presents three layers of consistency for materialized views in a distributed environment. This consistency develops simple and scalable algorithms for achieving MVC in a warehouse database.

Recent research in the area of conceptual design of data warehouse introduces the E/R (Entity/Relationship) diagrams conceptual model, on which the graph model of this research is based. Matteo Golfarelli, et al., used the E/R model to build a Dimensional Fact (DF) model. He also represents query patterns on a fact scheme made up of a set of markers placed on the dimension attributes. The DF model was used as a basis to change the graph model into MDD model.

Aric Shoshani introduces a statistical database (SDB) schema using a graph representation of a bi-dimensional table that represents attributes as nodes, and represents levels of hierarchical classification as branching lines. This formed the basis of creating the symbolic representation of the multidimensional cube of this research.

3 Conceptual Design of MDD model

The MDD model was created in two steps. First, the structure of the relational format was changed into the graph model structure. Then, the data elements of the relational format were changed into dimensional fact elements of nodes and rectangles to represent various data dimensions. As a result, the bidimensional relational format became a multidimensional model, referred to in this paper as a Cube Model. The Graph Model is a two-dimensional representation of the multidimensional theory of the Cube Model.

The connecting links between the various dimensions and associated data values comprise the multidimensionality of the Cube Model. On paper, the Graph Model represents this multidimensionality with basic MDD components and relationships between the various groupings. A MDD form represents dimension as logical grouping of attributes with a common atomic key relationship.
In a great mass of data elements, one cannot identify whether an attribute is a dimension or a variable; the selection is automatic, not human. The system does the selection using what is called a propositional function. The result needs to be verified if it is really a dimension, so it is in the meantime called a candidate dimension. To verify, an algorithm is used to compare attribute by attribute, until the candidate dimension is verified as a real dimension. A real dimension is called actual dimension.

Each group of data elements includes so many dimensions that it is necessary to identify the actual dimensions, those that are relevant to the needs of the management. A dimension is the logical grouping of attributes with a common atomic key relationship [6]. This is a subject-oriented grouping that includes product, location and time.

How is a dimension identified exactly? First, when an attribute in a data entity is established as part of a MDD model, the attribute is classified as ‘candidate dimension’. The propositional function is a formal way for representing knowledge in terms of declarative sentences [8]. The propositional function specifies that any attribute \( x \) of a data format is defined as a candidate dimension if the attribute is an index key field or its data is expressed either in character or date format, thus: \((\forall x)(I_x \cup (d_{var} \cup d_{num})) \Rightarrow D_i\)

where, \( I_x \) is the index key format, normally stored as key field, \( x \) is the type of data property which is equivalent to character format (denoted by “c”), or date/time format (denoted by “t”), and \( D_i \) is the attribute \( x \) within dimension of data entity \( i \), when \( i > 0 \).

Second, any candidate dimension is reclassified as ‘actual dimension’ using a comparative algorithm as described in the next section (Section 3.2: Building the B-tree Model).

### 3.1 Defining Candidate Dimension and Variable

#### 3.1.1 Selection of Candidate Dimensions

In a great mass of data elements, one cannot identify whether an attribute is a dimension or a variable; the selection is automatic, not human. The system does the selection using what is called a propositional function. The result needs to be verified if it is really a dimension, so it is in the meantime called a candidate dimension. To verify, an algorithm is used to compare attribute by attribute, until the candidate dimension is verified as a real dimension. A real dimension is called actual dimension.

Each group of data elements includes so many dimensions that it is necessary to identify the actual dimensions, those that are relevant to the needs of the management. A dimension is the logical grouping of attributes with a common atomic key relationship [6]. This is a subject-oriented grouping that includes product, location and time.

How is a dimension identified exactly? First, when an attribute in a data entity is established as part of a MDD model, the attribute is classified as ‘candidate dimension’. The propositional function is a formal way for representing knowledge in terms of declarative sentences [8]. The propositional function specifies that any attribute \( x \) of a data format is defined as a candidate dimension if the attribute is an index key field or its data is expressed either in character or date format, thus: \((\forall x)(I_x \cup (d_{var} \cup d_{num})) \Rightarrow D_i\)

where, \( I_x \) is the index key format, normally stored as key field, \( x \) is the type of data property which is equivalent to character format (denoted by “c”), or date/time format (denoted by “t”), and \( D_i \) is the attribute \( x \) within dimension of data entity \( i \), when \( i > 0 \).

Second, any candidate dimension is reclassified as ‘actual dimension’ using a comparative algorithm as described in the next section (Section 3.2: Building the B-tree Model).

#### 3.1.2 Selection of Candidate Variables

Each group of data includes so many variables, that it is necessary to identify the variables that are relevant to the needs of the management. A variable is a fact or measure that is generally stored as a numeric field [6]. According to the attributes remaining after the candidate dimensions are indicated, the propositional function of defining a candidate variable is that any attribute \( x \) is defined as a candidate variable if its data type is in a numeric (binary, integer, or decimal) format: \((\forall x)(d_{num}) \Rightarrow V_i\)

where, \( d_{num} \) is data type in numeric format (denoted by “n”), and \( V_i \) is attribute \( x \) within the variable of data entity \( i \), where \( i > 0 \).

The term relative dimension refers to the description of the associated data of a dimension. It defines the relationships of an ordinary attribute to a grouping relation. The first step of defining the relative dimension is to convert it into a candidate dimension. The next step is to repeatedly predicate the characteristic of actual dimension and relative dimension. (See Section 3.2)
### Example 1: Define candidate dimension and variable of data entity PRODUCT.

Table 1 shows that the attribute PRODUCT is the primary key field of the data entity. This is thus assigned as 'candidate dimension.' Other attributes (SALESMAN, MANUFACTURER, CUSTOMER, and SHIPMENT) present the foreign key fields of the data entity. These attributes are thus assigned as candidate dimensions corresponding to the PRODUCT attribute. The attributes of CATEGORY, SIZE, and ORDER DATE are assigned as candidate dimension because they are in character or date formats. The attributes of ORDER QUANTITY and PRICE are classified as candidate variables because they are in numeric format. The result is:

\[
\begin{align*}
D_{\text{PRODUCT}} & \quad D_{\text{PRODUCT}} \\
D_{\text{PRODUCT}} & \quad D_{\text{PRODUCT}} \\
D_{\text{PRODUCT}} & \quad D_{\text{PRODUCT}} \\
D_{\text{PRODUCT}} & \quad D_{\text{PRODUCT}} \\
D_{\text{PRODUCT}} & \quad D_{\text{PRODUCT}} \\
D_{\text{PRODUCT}} & \quad D_{\text{PRODUCT}} \\
D_{\text{PRODUCT}} & \quad D_{\text{PRODUCT}}
\end{align*}
\]

3.2 Building the B-tree of MDD model

The previous chapter discusses the automated process of identifying candidate dimensions and variables within a mass of high-volume data. This chapter presents the process of changing candidate dimensions and variables into actual dimensions and variables.

The MDD is a finite set of grouping relations, a combination of the attributes of dimension, variable, and/or relative dimension. Grouping relation generally contains dimension as base attributes carrying useful information. The variables attribute and relative dimension are related to any dimension attribute called 'associated attributes' and are derived from the base attributes. The candidate dimension, which is the primary index of a data entity, is assigned as 'main dimension,' the starting point of the Graph Model.

The starting point of a tree structure is the main dimension, which is represented by a center node. From this, other candidate dimension attributes (sub-dimensions) follow in the form of smaller circles. For instance, elements of Table 1 are shown in Figure 1 with lines that delineate the relationships between the main dimension PROD-
able' in Section 3.1.2, they must be compared together within the same data. This repeated predication is necessary in order to verify whether the candidate dimensions and variables are actual dimensions, variables, and relative dimensions.

The comparative process follows:

1. Choose one data entity from the warehouse database.

2. From the data entity, choose one candidate dimension as primary index key.

3. Use this primary index key to start a data map. Map the primary index key with the other attributes in the data entity.

4. After mapping, use a comparative algorithm to screen and remove all incomplete, missing, unrelated, and duplicate data.

5. Program the algorithm to identify each attribute in each data set as either ‘actual dimension,’ ‘variable,’ or ‘relative dimension.’ (Or use commercial software such as Personal Express Tool<sup>TM</sup>, Metacube<sup>TM</sup>, etc.)

6. Execute the program.

Data is now ready for analysis for relationships among the attributes of one data entity. The Graph Model now incorporates the element of relationship between the primary and other attributes: the small circles are white to indicate sub-dimensions while black circles indicate relative dimensions (See Section 5.2).

Comparative Algorithm:

At each data mapping:

- Initially, set \( V_L_1, V_L_2 \) to empty.
- Count data values of \textit{first attribute} \((a_1)\) of data mapping into \( V_L_1 \).
- Do while data mapping is not empty
  - If \textit{second attribute} \((a_2)\) of data mapping is candidate dimension.
    - Performed SQL process of second attribute
      1. Count data values into \( V_L_2 \)
      2. If \( V_L_1 \) equal to \( V_L_2 \) Set \( a_2 \) to be relative dimension.
         Else Set \( a_2 \) to be actual dimension.
    Else
    - Performed SQL process of second attribute
      1. If \( V_L_1 \) equal to \( V_L_2 \) Set \( a_2 \) to be relative dimension.
         Else Set \( a_2 \) to be actual variable
      - \( V_L_2 \) is set to empty.

End Do

End Algorithm

Example 2: Identifying actual dimension, variable, and relative dimension of \textit{PRODUCT} data entity

1. Extract the file labeled \textit{PRODUCT} data entity from the warehouse database.

2. From the file’s many attributes, select \( D_{product} \) as the primary index key.

3. Compare \( D_{product} \) with the other attributes.

4. Data mapping is accomplished as: \{\( D_{product} \cup D_{salesman}\), \( D_{product} \cup D_{manufacturer}\), \( D_{product} \cup D_{customer}\), \( D_{product} \cup D_{shipment}\), \( D_{product} \cup D_{category}\), \( D_{product} \cup D_{size}\), \( D_{product} \cup V_{orderquantity}\), \( D_{product} \cup V_{price}\)\}

5. Execute comparative algorithm.

The program executes a routine that counts the item number of \( D_{product} \) attribute by aggregate function (e.g. \textit{sum}, \textit{avg}, \textit{min}, \textit{count}) into variable \( V_L_1 \). The routine is executed until the data mapping set is empty.

At the first data mapping set, the comparative algorithm represents the difference between \( D_{product} \) and \( D_{salesman} \) as candidate dimension. The \( D_{salesman} \) attribute is counted by aggregate function of the SQL process and recorded into variable \( V_L_2 \).

6. Identify actual and relative elements.

The program executes a routine that compares the variables \( V_L_1 \) and \( V_L_2 \). If the results show absolute dissimilarity, then the \( D_{salesman} \) attribute is assigned as actual dimension. If the results show similarity, then, the attribute is assigned as relative dimension.

The program executes the routine to process the candidate dimensions of \textit{MANUFACTURER}, \textit{CUSTOMER}, and \textit{SHIPEMENT} with \textit{PRODUCT} attribute and assigned as actual dimensions. If the Step 6 routine shows a similarity of relationships, the candidate dimensions are labeled as relative dimensions. If dissimilar, they are labeled as actual dimension. When candidate variables of \textit{PRICE} are similar to \textit{PRODUCT}, \textit{PRICE} is then assigned as a relative dimension. The candidate variable of \textit{ORDER QUANTITY} is automatically assigned as actual variable.

3.2.2 Defining Main and Sub-dimensions

Figure 1 shows the relationship between main and sub dimensions, both indicated as actual dimensions. An actual dimension, a primary index key, is assigned as main dimension. The other actual dimensions are assigned as sub-dimensions.

For instance, a \( D_{product} \) attribute is a primary index key of the data entity labeled as \textit{PRODUCT}. The process assigns \textit{PRODUCT} attribute as main dimension or finest aggregation of the tree structure, represented by a large
3.2.3 Drawing the Graph Model

MDD models are drawn as graphs using symbols such as circle nodes, boxes that are connected with lines, dashes, and arrows. The main dimension is shown as a center-node in the Graph Model by a large circle. Sub-dimensions are shown as branch nodes, represented by small black circles of two types: Relative dimensions are solid black and connected solely to the center node, actual dimensions are white and may be connected to either the center node or to branch-nodes. Actual variables are boxes labeled with attribute names in numeric format and linked solely with the center node. Measurements are in boxes linked to the actual variables solely. They are separated from the actual variable box because they are only formulas while actual variable boxes store physical data values. An example is in the following Graph Model 2:

Assume the measurement of the attribute AMOUNT is included in this graph model, as a result of the multiplication of the attributes ORDER QUANTITY and PRICE. The model shows a box with the attribute name linked by a related actual variable. This is an example of a graph model utilizing only bidimensional (2D) relationships. For example, the SALES MAN branch node and the PRODUCT center node represents only a row-and-column relationship associated with a tabulated form of quantity and price value.

However, a Graph Model can be fully utilized to conduct a multidimensional view simply by connecting lines into the desired nodes. The model below illustrates the adaptivity of a 2D graph model into 3D graph model.

Figure 2 uses the Graph Model on the bidimensional (2D) level, similar to tabular representation. The result of order quantity is summarized by \( \{D_{\text{product}} \cup D_{\text{salesman}}\}, \{D_{\text{product}} \cup D_{\text{manufacturer}}\}, \{D_{\text{product}} \cup D_{\text{customer}}\}, \{D_{\text{product}} \cup D_{\text{shipment}}\}, \) and \( \{D_{\text{product}} \cup D_{\text{orderdate}}\}. \) (See Fig. 3a and 3b.)

Figures 3a and 3b show similar results by simply shifting the focus of analysis from SALES MAN to MANUFACTURER. The multidimensionality of the data analysis is in the flexibility of shifting focus of analysis as desired. Figure 3c shows the summary of ORDER QUANTITY as processed by \( \{D_{\text{product}} \cup D_{\text{salesman}} \cup D_{\text{manufacturer}}\}, \{D_{\text{product}} \cup D_{\text{salesman}} \cup D_{\text{customer}}\}, \{D_{\text{product}} \cup D_{\text{salesman}} \cup D_{\text{shipment}}\}, \) and \( \{D_{\text{product}} \cup D_{\text{salesman}} \cup D_{\text{orderdate}}\}. \) The results of processing AMOUNT and the ORDER QUANTITY are the same.

To transform a graph model into a MDD model using Personal Express Tool™ the routine is:

**Variable Process:**

- DEFINE orderqty_01 VARIABLE <product salesman manufacturer> LD Orderqty of product-salesman-manufacturer dimension
- DEFINE orderqty_02 VARIABLE <product salesman customer> LD Orderqty of product-salesman-customer dimension
- DEFINE orderqty_03 VARIABLE <product salesman shipment> LD Orderqty of product-salesman-shipment dimension
- DEFINE orderqty_04 VARIABLE <product salesman orderdate> LD Orderqty of product-salesman-orderdate dimension

**Measurement Process:**

- DEFINE amount_01 FORMULA orderqty_01*Price LD amount of product-salesman-manufacturer dimension
- DEFINE amount_02 FORMULA orderqty_02*Price LD amount of product-salesman-customer dimension
The creation of a MDD model based on the Graph Model as shown in Figure 3 shows that a three-dimensional graph model is insufficient to analyze the problem in all its aspects. In such instances, the model is easily reorganized into as many dimensions as desired.

3.3 Linking Related Graph Models Together

When dimension attributes of various data entities are related, Graph Models may be linked using a dotted line to connect the related dimension attributes. An example follows.

There are three Graph Models in Figure 4: a PRODUCT graph model, a SALESMAN graph model, and a DATE graph model. Table 2's graph model with the base attribute SALESMAN is associated to the same attribute in the PRODUCT graph model. The connection of the two graph models will result in a wider dimensional analysis.

For example, management may want to know how much commission came out of the purchases of customer X. The analysis is completed when two graph models are combined as: \( D_{product} \cup D_{saleman} \cup D_{customer} \cup D_{ordertime} \Rightarrow V_{commission} \).

3.3.1 Additional Aids

Illustrating multidimensional views in other ways, (e.g. Roll-Ups, Drill-Downs) can be useful techniques to present levels of hierarchical data. These techniques are helpful because they create attributes for each level on the hierarchy. For instance, an ORDER DATE attribute may be defined into levels of WEEK, MONTH, QUARTER, and YEAR. The attribute AREA may be divided into geographical levels such as: CONTINENT, COUNTRY, REGION, CITY, and so on. Users can roll-up and drill down through multiple dimensions concurrently.
The MOD model can easily incorporate such additional procedures. Existing databases can easily be processed using these add-ons. The MOD model can be reprogrammed to automatically process these new dimensions. This not only widens the range of analysis, it also allows local programmers to utilize routines developed elsewhere to enrich their processing functions.

For example, the year 1998 of ORDER DATE attribute is connected to year 1998 in DATE data entity. The QUARTER sublevel of year 1998 is divided into Q198, Q298, Q398, and Q498. The MONTH sublevel of Q198 is divided into JAN98, FEB98, and MAR98, all embedded as aggregation of detailed levels and summed up to the highest level as follows:

The routine may be programmed thus:

```
DEFINE parent.orderdate RELATION orderdate <orderdate>
CALL execution_1
ROLLUP orderqty_04 OVER orderdate USING parent.orderdate
ROLLUP amount_04 OVER orderdate USING parent.orderdate
```

The system will automatically aggregate values of actual data to the higher level, as shown in Table 4.

### 4 Conclusion

This paper introduces the MDD model, a more efficient alternative to the relational database, the standard operating format all over the world for high volume data processing. The model was created with this four-point focus: 1) to create a MDD that meets user requirements more accurately, 2) to design a routine that automatically creates an MDDB format, 3) to speed up the input-output design phase, and 4) to minimize human error.

The MDD structure identifies data attributes as candidate dimension or/and candidate variables using the propositional function. The attributes are then compared by comparative algorithm for relationship between attributes. The outcome is automatically generated as MDDB by linking similar dimension attributes to build the multidimensional views. Data loading is expressed as query patterns. Validation and rebuilding is by user interaction. Roll-up and drill down features of OLAP can be incorporated as self-activating routines to increase efficiency.

### References

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>B1</td>
<td>$2940.00</td>
<td>$2940.00</td>
<td>$740.00</td>
<td>$2200.00</td>
<td>0.00</td>
</tr>
<tr>
<td>B2</td>
<td>$1865.00</td>
<td>$1865.00</td>
<td>0.00</td>
<td>$1865.00</td>
<td>0.00</td>
</tr>
<tr>
<td>C1</td>
<td>$1006.00</td>
<td>$1006.00</td>
<td>$796.00</td>
<td>$210.00</td>
<td>0.00</td>
</tr>
<tr>
<td>D1</td>
<td>$3744.00</td>
<td>$3744.00</td>
<td>$900.00</td>
<td>$1650.00</td>
<td>$1194.00</td>
</tr>
</tbody>
</table>

Table 4:
Notes for Contributors

The prime purpose of the journal is to publish original research papers in the fields of Computer Science and Information Systems, as well as shorter technical research notes. However, non-refereed review and exploratory articles of interest to the journal’s readers will be considered for publication under sections marked as Communications of Viewpoints. While English is the preferred language of the journal, papers in Afrikaans will also be accepted. Typed manuscripts for review should be submitted in triplicate to the editor.

Form of Manuscript
Manuscripts for review should be prepared according to the following guidelines:

- Use wide margins and 1 1/2 or double spacing.
- The first page should include:
  - the title (as brief as possible)
  - the author's initials and surname
  - the author's affiliation and address
  - an abstract of less than 200 words
  - an appropriate keyword list
  - a list of relevant Computing Review Categories
  - Tables and figures should be numbered and titled.
- References should be listed at the end of the text in alphabetic order of the (first) author's surname, and should be cited in the text according to the Harvard method.

Manuscripts accepted for publication should comply with guidelines as set out on the SACJ web page, http://www.cs.up.ac.za/sacj, which gives a number of examples.

SACJ is produced using the \LaTeX\ document preparation system, in particular \LaTeX\ 2e. Previous versions were produced using a style file for a much older version of \LaTeX, which is no longer supported.

Please see the web site for further information on how to produce manuscripts which have been accepted for publication.

Authors of accepted publications will be required to sign a copyright transfer form.

Charges
Charges per final page will be levied on papers accepted for publication. They will be scaled to reflect typesetting, reproduction and other costs. Currently, the minimum rate is R30.00 per final page for contributions which require no further attention. The maximum is R120.00, prices inclusive of VAT.

These charges may be waived upon request of the author and the discretion of the editor.

Proofs
Proofs of accepted papers may be sent to the author to ensure that typesetting is correct, and not for addition of new material or major amendments to the text. Corrected proofs should be returned to the production editor within three days.

Letters and Communications
Letters to the editor are welcomed. They should be signed, and should be limited to about 500 words. Announcements and communications of interest to the readership will be considered for publication in a separate section of the journal. Communications may also reflect minor research contributions. However, such communications will not be refereed and will not be deemed as fully-fledged publications for state subsidy purposes.

Book Reviews
Contributions in this regard will be welcomed. Views and opinions expressed in such reviews should, however, be regarded as those of the reviewer alone.

Advertisement
Placement of advertisements at R1000.00 per full page per issue and R500.00 per half page per issue will be considered. These charges exclude specialised production costs, which will be borne by the advertiser. Enquiries should be directed to the editor.