J M Bishop
20/3/90

South African Computer Journal
Number 1
January 1990

Suid-Afrikaanse Rekenaar-tydskrif
Nommer 1
Januarie 1990

Computer Science and Information Systems

Rekenaarwetenskap en Inligtingstelsels

The South African Computer Journal
An official publication of the South African Computer Society and the South African Institute of Computer Scientists

Die Suid-Afrikaanse Rekenaartydskrif
'n Amptelike publikasie van die Suid-Afrikaanse Rekenaarvereniging en die Suid-Afrikaanse Instituut vir Rekenaarwetenskaplikes

Subscriptions

	Annual	Single copy
Southern Africa:	R32-00	R8-00
Elsewhere:	$32-00	$8-00

to be sent to:
Computer Society of South Africa
Box 1714 Halfway House 1685

Editorial

At last the first edition of SACJ is available. I trust that readers will find it worth the waiting. There have been a number of teething problems in getting things together, the many details of which need not be spelt out here. One significant challenge was to cope with the consequences of the resignation of Quintin Gee, QI's highly competent production editor. He assisted in the initial phases of getting this publication together but had to resign for personal reasons. It is fitting to acknowledge here not only his initial advice and assistance in getting this first issue of SACJ off the ground, but also the many hours of work that he spent in previously producing QI.

Quintin's resignation meant that a new *modus operandi* for typesetting and printing had to be established. The exercise was not only time-consuming, but also has significant cost implications. Fortunately, the Unit for Software Engineering (USE) at Pretoria University has generously agreed to sponsor this first edition. On behalf of the South African computing community, I should like to thank them for their generosity. Now that they have made a first issue of SACJ possible, it is hoped to solicit the sponsorship of one of the larger computer companies for future editions.

It might be of interest to take readers on a walk through the new journal to highlight various aspects. To begin with, the cover design follows that of several journals whose titles have the format: *The South African Journal of Subject / Die Suid-Afrikaanse tydskrif vir Vakgebied* (where *Subject* and *Vakgebied* are appropriately instantiated). While colours vary, these journals generally have *Subject* and *Vakgebied* restated on the darker portion of the cover. SACJ's title was chosen in preference to a more descriptive but also more cumbersome title such as *The South African Journal of Computer Science and Information Systems*. The appearance of the words *Computer Science and Information Systems / Rekenaarwetenskap en Inligtingstelsels* on the cover are thus out of step with the original inspiration, but seem appropriate under the circumstances.

The inside cover is of interest for several reasons. Firstly, note that Peter Lay has kindly agreed to lighten my task by acting as an assistant editor. He will deal with matters relating to Information Systems. *Contributions in this area should henceforth please be sent directly to him*. Also note that an editorial board of distinguished persons has been assembled. I should like to once again thank board members for adding status to SACJ by agreeing to serve in this capacity. They will be consulted on matters of editorial policy whenever appropriate. Finally, the subscription costs have been increased to keep pace with production costs. This increase does not affect SAICS members, who will continue to receive the journal as one of the benefits of membership.

The guest editorial by Pieter Kritzinger makes for interesting reading. Several points of concern about computer-related research in South Africa are raised. I trust that the article will focus attention on these problems and stimulate a debate which will lead to eventual solutions. It is hoped to make guest editorials a regular feature of future SACJ issues.

Of the eight research papers offered in the journal, four have been gone through the normal channel of refereeing and revisions. The remainder were submitted to the Vth SA Computer Symposium and are published here by invitation. Each paper submitted to the chairman of the symposium's program committee was sent to three referees. A ranking scheme, reflecting an aggregate measure of referee evaluation, was used as a basis for deciding on papers to be presented. After further editorial evaluation, the authors of four of the five highest ranking papers were invited to submit their papers to SACJ. While it was not possible to contact the fifth author in time for this edition, but it may be possible to publish that paper, together with a selection of others from the symposium, in future SACJ editions.

In the section marked *Communications* various items of news arriving at the editor's desk have been published. It was particularly gratifying to receive book review submissions in response to a prior general appeal. There has also been an enthusiastic response from book publishers, who have sent in a number of books for review. Titles are listed in the *Communications* section. Please contact me if you are willing to review one (or more) of these. Naturally, reviews of other books of interest in your possession will also be welcomed.

The final point to highlight in this walk through the journal is the increase in page charges indicated on the back inside cover. These reflect the increased cost of production. Since research papers in SACJ qualify for state subsidy at academic institutions, the charges should not, in general, present major problems for authors. However, it is worth pointing out that the final format of papers submitted significantly impacts on both the financial and editorial load. Submissions in camera-ready format (or nearly so) result in both a cost savings and a speed up of turn-around time by several orders of magnitude. Since many readers may not be familiar with the printing process, it may be helpful to say something about it in order to substantiate this claim.

The printing process basically involves typesetting, shooting (or photographing), and then reproduction and binding. Apart from limiting the amount of material, the printer's client has very little control over the cost of shooting, reproduction and binding. On the

other hand, anyone equipped with moderate text- or word processing facilities and a laser printer can go a long way (if not all the way) towards typesetting a paper. Even a partially typeset paper helps significantly, as I will explain below.

By typesetting I simply mean knocking the paper into the right shape and producing a laser printout. The printers regard this is a tedious, error-prone task, even if they start off with an ASCII file rather than a hardcopy of the paper. Consequently, they tend to handle large-scale typesetting by subcontracting the task. Moreover, while they may be willing to typeset uncomplicated text, they tend to balk at text containing specialized mathematical and other notation. However, they are quite skilful at cutting and pasting text, and at enlarging or reducing photographed or scanned diagrams. They are even willing to redraw sketches which are not too complicated.

As a result of the above, I have pressed several authors to do their own typesetting. In cases where it was problematic to produce double column format, a single column of appropriate width was requested. While this is a second-best option, it allows for cutting and pasting to be done by the printers. Some sketches have either been directly reduced from the author's original, while others have been redrawn by the printer. By way of exception, I have personally undertaken the typesetting of a few papers using WordPerfect. However, I would like to avoid this as far as possible in future, and consequently appeal to potential authors to make every effort to do their own typesetting.

From SACJ's point of view encouraging authors to do their own typesetting involves a compromise in that there will inevitably be slight variations in the print from one article to the next (as is in fact the case in this issue). If you are pedantically inclined, you might consider this to be a disaster. Personally, I regard it as a rather neat advertisement for the typesetting skills of SACJ contributors.

As an aside, since the handling of TEX files was initially a problem for me, I was pleased to discover that Peter Wood and his colleagues at UCT have mastered the art of producing TEX printout in the format now before you. Future authors who use TEX should consult them on details.

As to the future, it is not possible at the this stage to commit to a fixed number of SACJ issues per year. The number of issues is constrained by finance, submissions of the right quality, and time available to the editorial staff (including our anonymous and unsung heroes - the referees). The ideal is to produce four issues per year, but this may not always be attainable.

In conclusion, if readers have as much fun in reading this first issue of SACJ as I have had in editing it, the hours spent on it will have been well worthwhile. Hopefully SACJ is destined not only to be a permanent feature of the Southern African computing scene, but also to significantly contribute to research in the region.

Derrick Kourie
Editor

This SACJ issue is sponsored by
The Unit for Software Engineering
(USE)
Department of Computer Science
Pretoria University

Guest Editorial

Funding Computer Science Research in South Africa

P S Kritzinger
Department of Computer Science, University of Cape Town, Rondebosch 7700

The word *research* has many connotations and is often abused. In everyday language a person does not simply *search for information in a library*, for example, but rather does *research*, thus pretentiously conferring an aura of intellectual activity on an effort which requires very little original thought.

Here I will interpret the term to mean work which generates results that gain international recognition. This implies that the work is published in good international journals or presented at international conferences. I believe this is the only valid index of the quality of research.

With very few exceptions, the computer industry in South Africa is a consumer of computer technology, rather than a developer. In contrast with, say, the chemical industry, there is therefore no tradition of research in computer science in the South African computer industry and computer science researchers therefore have, as virtually their only source of funding, the Foundation for Research Development (FRD) which has its origins in the CSIR.

The FRD was formed in April 1984 with the development and use of research expertise in the natural and applied sciences and engineering as its mission. This mission is primarily directed at the universities, museums and technicons with the ultimate aim of improving the life of all South Africans.

Although the FRD has several programmes, the two which are of main concern to computer scientists are the Core Programmes and the Special Programmes.

FRD Core Programmes foster the optimum development of a scientific and technological knowledge base by supporting individual self-initiated research. These programmes, started only about 4 years ago, have met with considerable acclaim, particularly in regard to the way in which research funding for a particular individual is decided. To qualify for support within a Core Programme, researchers must obtain a certain evaluation status within the FRD and funding is then linked directly and exponentially to the merit of the individual concerned, rather than being linked to the specific project proposed.

In the evaluation process, peer review is strongly emphasised. The researcher himself is expected to nominate referees, whose status and reports play a decisive role in the evaluation. As a result of this evaluation, an applicant is assigned a specific evaluation status category. There are currently 9 categories in all, but the ones of main interest are:

- **A** researchers who are without any doubt accepted by the international community as being amongst the leaders in their field (52);
- **B** researchers not in category **A** but who nevertheless enjoy considerable international recognition as independent researchers of high quality (182);
- **C** proven researchers who have maintained a constant high level of research productivity and whose work is regularly made known internationally, or proven researchers whose current research output is less but who are actively engaged in scholastic activity (433);
- **P** researchers younger than 35 years of age who have already obtained a doctoral degree and who have shown exceptional potential as researchers (10); and
- **Y** young researchers usually under 35 years of age, who are highly likely to achieve C status by the end of their support period (108).

The number of researchers in the various categories as of August 1989 has been indicated in parentheses above. Of these, only 7 persons are computer scientists: 1 in category B; 3 in category C; and 3 in category Y. Only 4 departments of computer science are involved.

The other main programmes of concern to computer persons are the Special Programmes which aim at developing research manpower in priority areas. After identification of an area that merits particular research development, given local expertise, a Special Programme is launched to address the problem in the national interest.

Although a manager of a Special Programme has to be an FRD evaluated researcher, the same need not be true for the other team members. Regular peer evaluation of researchers as well as evaluation of the progress and results of Special Programmes are considered essential. Special Programme awards will be made for the first time towards the end of 1989. It is therefore not yet known whether proposals already submitted for programmes in computer science have been successful.

It is clear that, in the context explained above, there is virtually no computer science research being done in South Africa - a scary thought which has considerable implications for this country! Why is this so? There are several reasons, but I would like to single out two in particular.

Qualified faculty and students is an abiding problem at the heart of computer science departments. Acquisition of new faculty members is an issue intimately linked to the number of graduate students successfully completing PhD degrees. This problem is by no means unique to South Africa. For instance, data gathered in North America indicates that in 1983 there were over 200 vacancies in the 91 departments that have doctoral programmes in computer science. At the same time, only approximately 250 PhD's were granted in North America - a figure that has remained relatively unchanged for the past several years. A large number of those graduates were attracted to industry and industrial research laboratories. Although I do not have solid data at my disposal, I would think that South Africa produces at most one PhD graduate in computer science per year. There are currently 20 departments of computer science at universities is South Africa. It will therefore take us 20 years to locally produce one new faculty member with a PhD in computer science for every university.

Contributing to the above problem is our current academic image. The graduate student usually sees concerned computer science faculty members as rather harried individuals, having large undergraduate classes, much committee and professional work, and labouring under an ill-fitting model (applicable to more established disciplines) for decisions on tenure, salary and promotion. Further, as undergraduates, many prospective graduate students were not engaged in research projects involving computer science faculty, and for that reason were not exposed to graduate students doing research, and rarely developed a camaraderie with any computer science professionals. At last count there were only 5 individuals in South Africa who completed their computer science doctorate at a university outside South Africa where they had the good fortune to work in an environment in which sufficient faculty and funds were available to create an ethos of research. It is difficult to convince students that their interests and goals can be served by a PhD in computer science or by an academic career.

The second problem, which is of greater concern to me since there is no immediate solution to it, has to do with the fact that senior persons who decide the fate and fortune of academic computer science departments are, in general, individuals whose professional careers started well before computing machines came into every day use - that is to say, in the years B.C. (Before Computers). These persons of influence do not always understand what "computers" are, and what their potential influence upon the workplace in particular and society in general are. As far as research (as opposed to teaching) is concerned, most of them understand that a medical school needs special and expensive equipment (not to mention, expensive faculty) and that engineers must have a workshop and special machinery to teach their students and conduct research. They understand that if one needs to build up a defense industry, it will cost billions of rands; but they are not so sure about computer science, even though many other countries have recognised it as of national strategic importance.

I believe that only time and dedication will lead to a solution of these seemingly insurmountable problems and allow computer scientists to take their rightful place in the research community in South Africa.

Bibliography

P J Denning, [1981], Eating our seed corn, *Communications of the A.C.M.*, **24**(6), pp.341 - 343.

J Tartar (Ed.), [1985], The 1984 Snowbird Report: Future issues in computer science, *Communications of the ACM*, **28**(5), pp490 - 493.

D Gries, R Miller, R Ritchie and P Young, [1986], Imbalance between growth and funding in academic computing science: two trends colliding, *Communications of the A.C.M.*, **29**(9), pp.870 - 878.

J E Hopcroft and D B Krafft, [1987], Towards better computer science, *IEEE Spectrum*, pp.58 - 60.

A Multi-criteria Partitioning Technique for Information System Design

J. Mende
Department of Accounting, University of the Witwatersrand, WITS 2050

Abstract

In order to design and operate a complex computer based information system, one needs to partition its transformation process into modules of manageable size. The same process can be partitioned in many different ways, each alternative partition having a different effect on the system's multiple success criteria. The optimal partition of a given process can be found by a four-step technique:

1. *Identify significant interfunctional relationships.*
2. *Rank system success criteria in sequence of user preference.*
3. *Apply known partitioning rules in preference sequence to find a set of near-optimal partitions.*
4. *Select the final partition using judgement, prototyping or other techniques.*

Keywords : *System design, partitioning, success criteria, effectiveness, efficiency, theorem-proving.*

CR Categories: *D.2.2, H.2.2*

Presented at V th S.A. Computer Symposium.

Introduction

The typical information system today contains a complicated transformation mechanism which performs a wide variety of different computer operations on many different data types. Developing such a mechanism still poses a difficult problem. Solving that problem calls for the application of Descartes' Principle, namely to divide it into simpler parts, and then into even simpler parts, until one reaches problems of manageable size [15].

For instance, one can divide the entire problem into separate design, programming and implementation problems. One can then divide the design problem into separate data-design, procedure-design and process-design problems. Then the *process-design* problem can be *partitioned* into still simpler problems of designing individual processing modules.

Existing Partitioning Techniques

More than fifty alternative partitioning techniques have been proposed by previous authors (table 1). Each technique was intended to help designers find an optimal partition. However previous designers' notions of optimality were imprecise. A recent study [47] showed that an optimal information system should satisfy at least *four major* criteria, each major criterion comprising several minor criteria. They are the following:

— technical efficiency T (the extent to which waste has been eliminated from the system) involving minor criteria such as resource utilisation and output realisation

— economic efficiency E (the extent to which the cost of the input resource mix has been optimised) involving minor criteria such as development cost, maintainability and reusability

— user effectiveness N (the extent to which the value of the output information mix has been optimised) involving minor criteria such as timeliness, reliability and flexibility

— marginal effectiveness R (the extent to which the incremental output value has been optimised in relation to incremental input cost) involving minor criteria such as expansion potential and sufficiency.

The first three major criteria are all affected by the way the system's transformation process is partitioned. Therefore an optimal partition should satisfy the three major criteria T, E and N. However an analysis of the criteria actually addressed by existing techniques (in the right-hand column of table 1) shows that none of these techniques addresses *all* three criteria.

Authors	Sources	Acronym	Technique	Criteria
Alford	2,3,71	SREM	Software Requirements Engineering	N
Archibald et al	4	ADAPT	Specification & Design Language	E
Barros	6		Partitioning Model	E
Berzins et al	7,8	MSG	Architectural Design	N,E
Booch, Verity	9,79		Object Oriented Development	E,T
Botting	10		Entity Life Cycle Modeling	N
Chand, Warnier	12,82	LCS	Hierarchy of data transforms	E
Constantine et al	50,61,76, 78,79,90		Structured Design	E
Davis & Vick	13	PDM	Process Design Methodology	E
De Marco	14,81		Functional Decomposition	N
Emery	17		Modular Design	E
Ewusi-Mensah	18	SC	Strong Components	E
Freeman	19	Draco	Specification refinement	N,E
Gane & Sarson	21,85		Data Flow Explosion	N,E
Gilb & Krzanik	22	DBO	Design by Objectives	N,E
Gomaa	23	Darts	Design Approach for Real-Time	E
Huff	24	SDS	Systematic Design Methodology	E
Hamilton	25	HOS	Higher Order Software	N
Hansen et al	26		Structural Modeling	N,E
Hawryszkiewycz	27		Semantic Design	N
Heitmeyer	28		Implementation-free	N
Heninger	29		Externally-oriented	N
Hutchens	30		Cluster Analysis	E
IBM	31	HIPO	Top Down Design	E
Jackson, Cameron	32,11	JSD	Event Modeling	E
Klein	34		Mixed Integer Programming	E,T
Lundeberg et al	35,36	ISAC	D-Graphs	N,E
Martin	37		Logical-to-physical mapping	E
Mastro	38		Functional Hierarchy	N,E
Matsumura et al	39		Structural Modeling	E
McFarland	40		Bottom-up-design	N
Mekly & Yau	41		Abstract Process Networks	E
Mills	48		Box structure	E
Miyamoto & Yeh	49		Requirements Analysis	E
Neumann	51		Hierarchical Abstraction	N
Nielsen	52		Process Graphs	E,T
Orr	53		Structured System Development	E
Orr	54		Structured Requirements Def.	N
Parnas	55,56, 58,60		Information Hiding	E
Parnas	57,59		Transparency and Uses Relation	E
Randell	62		Functional Components	E
Robinson	63,64		Entity-event Modeling	E
Ross, Rudkin	65-7,70	SADT	Structured Analysis and Design	E
Rotenstreich	68		Vertical and Horizontal	E
Ritter	69		Environmental Model	E
Simpson	74,75	MASCOT	Functional & Structural Decomp.	E
Sievert & Mizell	73	TAGS	Automated Generation of Systems	E
Stevens	77		Data flow network	E
Tse & Pong	79		Petri Nets	E
Wasserman	83	USE	External Design	N,E
Weber & Ehrig	84		Evolutionary Design	E
Wirth	86		Stepwise Refinement	E
Witt	87,88		Communicating Modules	E
Zave, Schatz	91,92,72		Operational Approach	E

Table 1. Analysis of Existing Process Partitioning Techniques

Research Objective and Method

This paper presents a comprehensive partitioning technique which helps the designer to find a process partition that best meets *all* relevant success criteria.

Emulating Ahituv [1], the new technique was established using the research method known as "theorem proving" [20]. Basic axioms were extracted from previous Information Systems publications. The procedure was then derived from those axioms by logical reasoning. Finally its feasibility was verified by applying the technique to a simple case study.

Axioms

The new partitioning technique is based on the following three propositions. The available evidence suggests that they are true for most information systems, but not necessarily *all* information systems.

Functional Partitions — *An information system's transformation process consists of many distinct processing functions. The set of processing functions can be partitioned into a set of modules by repeatedly considering pairs of functions X and Y and deciding whether they should be combined (X+Y), or separated (X/Y).*

Standard Information Systems texts such as Gane & Sarson [21], Jeffrey & Lawrence [33], Yourdon & Constantine [90], Weinberg [85] and Awad [5] show that an information system's transformation process can be decomposed into successively simpler modules until many elementary functions are isolated. Table 1 indicates that existing decomposition techniques do not ensure an optimal set of modules. However the elementary functions can be combined into a different, improved set of modules.

Design Objectives — *An information system's designer wants to find optimal process partitions with least effort. An optimal partition satisfies several independent success criteria which can be ranked in sequence of user preference.*

The authors of most of the fifty-five partitioning techniques have suggested that the information system designer should use those techniques in order (a) to find good partitions, and (b) to find those partitions relatively easily. This implies that many designers' motives actually conform to the first part of the axiom. The second part of the axiom is supported by the following examples. They involve a transformation process containing two functions X and Y. The process can be partitioned in two alternative ways, partition P(X+Y) combining X and Y while partition P(X/Y) separates them.

1. Suppose X and Y are to be executed at inherently different times. When P(X+Y) and P(X/Y) are implemented, they produce the same output mix from the same input mix, and so would have equal N and E. However P(X+Y) would waste more load time and memory space than P(X/Y), so the two partitions would have unequal T [44].

2. Suppose X and Y were weakly coupled. When P(X+Y) and P(X/Y) are implemented, they might produce the same output mix with the same degree of resource wastage. However P(X+Y) would be more costly to program and maintain than P(X/Y), and so the two partitions would have unequal E [90].

The possibility of ranking the criteria is confirmed by Klein et al [34]: "certain aspects take on more significance in different situations and different organisations."

Partitioning Rules — *An optimal process partition satisfies several partitioning rules. Each rule predicts whether two functions related in a particular way will be more successful or less successful in terms of a specific criterion if they are combined or separated.*

Previous authors have identified several types of interfunctional relationships across a broad spectrum of information systems. They have also discovered rules that distinguish between more successful and less successful partitions when functions involved in those relationships are either combined or separated. Each of those rules is associated with a specific success criterion [46]. The rules have been listed in table 2.

Rule	Relationship	Prediction
N1	Real time X and Y are time critical	Combine for N
N2	An environmental change affects X but not Y	Separate for N
N3	X and Y extract temporally independent information	Separate for N
N4	X and Y collect temporally independent source data	Separate for N
N5	X collects data which predates info extracted by Y	Separate for N
E1	X and Y depend on the same design decision	Combine for E
E2	X and Y access common data	Combine for E
E3	X and Y process data in a sequence subject to change	Combine for E
E4	X and Y are functionally or sequentially cohesive	Combine for E
E5	X and Y do not interact	Separate for E
E6	An environmental change affects X but not Y	Separate for E
E7	X + Y is incomprehensibly complex	Separate for E
E8	X + Y has a complex external interface	Separate for E
E9	X + Y is subject to change	Separate for E
E10	X iterates Y	Separate for E
E11	Several X perform Y	Separate for E
T1	X and Y process the same data	Combine for T
T2	X transmits a high volume of data to Y	Combine for T
T3	X iterates Y	Combine for T
T4	X accesses Y with high volume	Combine for T
T5	X accesses Y with high frequency	Combine for T
T6	X and Y are computationally intensive	Combine for T
T7	X is executed shortly before Y	Combine for T
T8	Real time X and Y are constrained by I/O speed	Combine for T
T9	X is optional	Separate for T
T10	X is only used once	Separate for T
T11	X transmits data to/from a sort Y	Separate for T
T12	X is computationally intensive but Y is not	Separate for T

Table 2. Known partitioning rules

Theorems

The following theorems hold for information systems that conform to the three axioms.

Multiple Partitions — *A information system's transformation process can be partitioned in a very large number of different ways.*

Donnellan [16] has proved mathematically that the number of partitions of a set of n + 1 elements is:

$$p(n+1) = \sum_{r=0}^{n} \binom{n}{r} p(n-r)$$

Numerical computation of this formula demonstrates that "values of p(n) rise steeply with n", as illustrated in table 3

Functions	Partitions	Number
Y,Z	Y+Z, Y/Z	2
X,Y,Z	X+Y+Z, X/Y+Z, Y/X+Z, X+Y/Z, X/Y/Z	5
W,X,Y,Z	W+X+Y+Z, W/X+Y+Z, X/W+Y+Z, Y/W+X+Z, Z/W+X+Y, W+X/Y+Z, W+Y/X+Z, W+Z/X+Y, W+X/Y/Z, W+Y/X/Z, W+Z/X/Y, W/X/Y+Z W+Y/X+Z, W+Z/X/Y, W/X/Y/Z	15
Q,R,....Z	Q+R+S+T+U+W+X+Y+Z, etc	115 975

Table 3. Partitions of n functions

The Functional Partitions axiom postulates that the number of functions in an information system is large. Consequently the number of alternative partitions is very large indeed.

Optimal Partitions — *A pair of functions in a transformation process may be subject to several rules associated with different success criteria. An optimal partition satisfies the rule(s) aimed at the dominant success criterion*

Suppose a transformation process contains two functions X and Y. By the Partitioning Rules axiom these functions may be involved in one or more relationships R1, R2 etc, each relationship being associated with one or more partitioning rules aimed at different success criteria C1, C2 etc. Therefore X and Y may be subject to one or more rules of the following form:

— If R1 then P(X+Y) > or < P(X/Y) for C1
— If R1 then P(X+Y) > or < P(X/Y) for C2
— If R2 then P(X+Y) > or < P(X/Y) for C1
— If R2 then P(X+Y) > or < P(X/Y) for C2.

By The Design Objectives axiom, the criteria are independent. So a partition which is optimal with respect to C1 is not necessarily optimal with respect to C2. Consequently X and Y may be subject to:

a. If R1 then P(X+Y) > P(X/Y) for C1
b. If R1 then P(X+Y) < P(X/Y) for C2
c. If R2 then P(X+Y) < P(X/Y) for C1
d. If R2 then P(X+Y) > P(X/Y) for C2.

By the Design Objectives axiom, the criteria can be ranked by user preference, say C1 C2. Therefore the partition that conforms to rule **a** will be more successful than the partition that conforms to rule **b.** Similarly rule **c** gives a better partition than rule **d.**

Alternative Reduction — *A comparatively small set of optimality candidates can be extracted from a large set of all possible partitions by:*
— identifying all significant relationships between pairs of functions
— ranking the system success criteria by user preference
— applying the dominant rules associated with the relationships.

By the Partitioning Rules axiom, the functions in a given transformation process satisfy several partitioning rules. These rules can be found from table 2 by identifying all significant relationships between pairs of functions.

The Optimal Partitions theorem shows that the dominant rules can be selected by ranking the success criteria. The dominant rules can then be used to determine whether related functions will be combined or separated in an optimal partition.

By the Multiple Partitions theorem, a transformation process containing n+1 uncombined functions can be partitioned in:

$$p(n+1) = p(n) + np(n-1) + \ldots + p(0)$$

If two functions are known to be combined in the optimal partition, then the two functions can be treated as a single function. So the number of alternatives reduces to p(n), i.e.

$p(n+1) - p(n) = np(n-1) + \ldots + p(0)$ fewer alternatives.

If two functions are known to be separated in the optimal partition, then all alternatives containing the combined functions can be dismissed. This also yields a substantial reduction in the number of alternatives.

Similarly knowledge of further combinations or separations reduces the number of alternatives still further.

Decision Technique — *In the absence of a better method, the system designer should find optimal partitions of information systems transformation processes by applying the alternative reduction procedure.*

Suppose an information system is being designed to perform 10 or more functions. According to the Design Objectives axiom the system designer wants to find the optimal partition of those functions. However the Multiple Partitions theorem predicts that more than 100 000 alternative partitions are possible. So in the absence of a direct search method, the designer needs to compare 100 000 alternatives.

Comparing those partitions involves 100000!/2 comparisons. This evidently involves an enormous amount of effort. However the Alternative Reduction theorem enables the designer to reduce the number of alternatives substantially. To minimise the search effort (Design Objectives theorem) the designer should therefore use the Alternative Reduction technique.

Case Study

To prove its feasibility and illustrate its applicability, the technique was applied to partition the transformation process of a simple invoicing system. The system performs the following functions:
— P.ACC accepts product data from a keyboard
— P.INS inserts new product records into an indexed-sequential sales file
— P.CHA changes existing product records via a product code index
— P.SOR sorts the product file in product name sequence
— P.PRT prints the sorted product file in the form of a price list
— O.ACC accepts orders data from a keyboard
— O.INS stores valid orders with product data on a serial sales file
— O.CAL retrieves orders from the sales file and calculates amounts
— O.PRT prints order data and amounts in the form of invoices.

The data flows between functions are shown in figure 1.

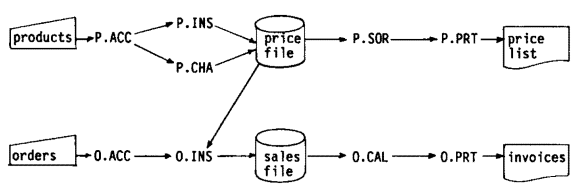

Fig. 1. Elementary processing functions

Interfunctional relationships were represented by their rule numbers in corresponding cells of figure 2. The symbols + and / indicate whether the rule predicts combination or separation respectively.

	P.INS	P.CHA	P.SOR	P.PRT	O.ACC	O.INS	O.CAL	O.PRT
P.ACC	E4+ T2+ T7+	E4+ T2+ T7+			N4/ E1+ E5/			
P.INS		E2+	N5/ E2+			N5/ E2+		
P.CHA			N5/ E2+			N5/ E2+		
P.SOR				E4+ T2+ T11/		N3/ E2+ T1+		
P.PRT								N3/ E1+ E5/
O.ACC						E4+ T2+ T7+		
O.INS							N5/ E2+	
O.CAL								E4+ T2+ T7+

Fig. 2. Relationships between functions

Conflicts between rules in the same cell of figure 2 were all resolved by ranking success criteria in the sequence N then E then T. The prioritised rules therefore call for the following functions to be *combined:*

— P.ACC and P.INS and P.CHA
— P.SOR and P.PRT
— O.ACC and O.INS
— O.CAL and O.PRT

The resulting modules are shown as a flowchart in figure 3, and as a structure chart in figure 4. A referee found this partition self-evidently optimal.

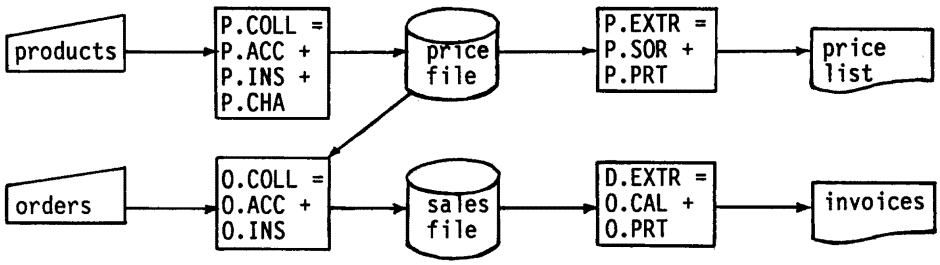

Fig. 3. Major Processing Modules

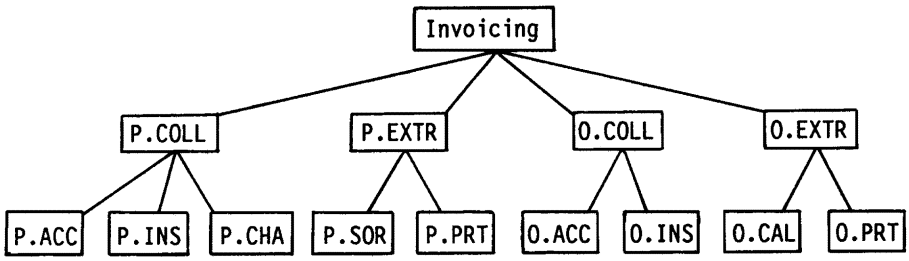

Fig. 4. Invoicing System Partition

Although extremely simple, the case study demonstrates that the technique is a) simple to use and b) can actually reduce the number of alternatives very substantially indeed, namely to a single optimal partition. With larger systems, the technique obviously requires more effort and may yield several optimal or near-optimal alternatives. The designer will then have to select the final partition using judgement, prototyping or one of the earlier techniques listed in table 1.

Conclusion

This paper derives a partitioning technique from some fairly common properties of information systems by means of the research method "theorem proving". The new technique enables system designers to apply known partitioning rules to find the process partition which satisfies all known system success criteria. Furthermore, the derivation confirms the author's previous predictions [42, 43, 45] that the subject Information Systems can and should contain techniques which are logically connected to underlying laws. Finally, a referee has suggested that this technique could serve as "the decision making front-end to an automated tool used in the design process".

References

[1] N Ahituv, [1987], A Metamodel of Information Flow: A Tool to Support Information Systems Theory, *Comm. ACM,* **30,** 781-791.

[2] M W Alford, [1977], A Requirements Engineering Methodology for Real-Time Processing Requirements, *IEEE Trans. Soft. Eng.* **3,** 60-69.

[3] M Alford, [1985], SREM at the Age of Eight; The Distributed Computing Design System, *IEEE Computer,* **18**(4), 36-46.

[4] J L Archibald, B M Leavenworth and L R Power, [1983], Abstract design and program translator: New tools for software design, *IBM Syst J,* **22,** 170-187.

[5] E M Awad, [1985], *Systems Analysis and Design,* R D Irwin Inc, Homewood, Illinois, p 266-269.

[6] O. Barros, [1981], Management Information Systems Structure, Types and Integration, *Inform. Systems,* **6,** 243-254.

[7] V Berzins and M Gray, [1985], Analysis and Design in MSG.84: Formalising Functional Specifications, *IEEE Trans Soft Eng,* **11,** 657-670.

[8] V Berzins, M Gray and D Naumann, [1986], Abstraction-Based Software Development, *Comm. ACM,* **29,** 402-415.

[9] G Booch, [1986], Object-Oriented Development, *IEEE Trans Soft Eng,* **12,** 211-221.

10] R J Botting, [1986], Into the Fourth Dimension -An Introduction to Dynamic Analysis and Design, *ACM SIGSOFT SE Notes,* **11.**2, 36-48.

[11] J R Cameron, [1986], An Overview of JSD, *IEEE Trans Soft Eng,* **12,** 222-240.

[12] D R Chand and S B Yadav, [1980], Logical Construction of Software, *Comm. ACM,* **23,** 546-555.

[13] C G Davis and C R Vick, [1977], The Software Development System, *IEEE Trans Soft Eng,* **3,** 69-84

[14] T De Marco, [1979], *Structured Analysis and System Specification,* Prentice-Hall, Englewood Cliffs, New Jersey.

[15] R Descartes, [1637], *A Discourse on Method,* Dent, London.

[16] T Donnellan, [1968], *Lattice Theory,* Pergamon, Oxford, p16.

[17] J C Emery, [1962], Modular Data Processing Systems Written in COBOL, *Comm. ACM,* **5,** 263-268.

[18] K Ewusi-Mensah, [1984], Identifying Subsystems in Information Systems Analysis, *Inform. Systems,* **9,** 181-190.

[19] P Freeman, [1987], A Conceptual Analysis of the Draco Approach to Constructing Software Systems, *IEEE Trans Soft Eng,* **13,** 830-844.

[20] R D Galliers and F F Land, [1987], Choosing Appropriate Information Systems Research Methodologies, *Comm. ACM,* **30,** 900-902.

[21] C Gane and T Sarson, [1979], *Structured Systems Analysis,* Prentice-Hall, Englewood Cliffs, New Jersey.

[22] T Gilb and L Krzanik, [1987], Design by Objectives [DBO]: A Basis for Automated Software Engineering Design, *ACM SIGSOFT SE Notes,* **12**(2), 42-49.

[23] H Gomaa, [1984], A Software Design Method for Real-Time Systems, *Comm. ACM,* **27,** 938-949.

[24] S L Huff, [1982], A Methodology for Supporting System Architects During Preliminary Design, *Info & Mgmt,* **5,** 259-268.

[25] M Hamilton and S Zeldin, [1976], Higher Order Software -A Methodology for Defining Software, *IEEE Trans Soft Eng,* **2,** 9-32.

[26] J Hansen, L E Heitger and L McKell, [1978], Computer-aided Modeling of Decision-support Systems, *J. Opl. Res. Soc.* **29,** 789-802.

[27] I T Hawryszkiewycz, [1983], A Sementic Design Method, *IEEE Trans Soft Eng,* **9,** 373-384.

[28] C L Heitmeyer and J D McLean, [1983], Abstract Requirements Specification: A New Approach and its Application, *IEEE Trans Soft Eng,* **9,** 580-589.

[29] K L Heninger, [1980], Specifying Software Requirements for Complex Systems: New Tech-

niques and Their Application, *IEEE Trans Soft Eng,* **6,** 2-13.
[30] D H Hutchens and V R Basili, [1985], System Structure Analysis, Clustering with Data Bindings, *IEEE Trans Soft Eng,* **11,** 749-757.
[31] IBM, [1973], HIPO: *Design Aid and Documentation Tool,* IBM Corporation, SR20-9413-0.
[32] M A Jackson, [1983], *System Development,* Prentice/Hall International, London.
[33] D R Jeffrey and M J Lawrence, [1984], *Systems Analysis and Design,* Prentice-Hall of Australia, Sydney.
[34] G Klein, P.O. Beck and B Konsynski, [1988], Computer-Aided Process Structuring Via Mixed Integer Programming, *Decision Sciences,* **19,** 750-761.
[35] M Lundeberg, G Goldkuhl and A Nilsson, [1979], A Systematic Approach to Information Systems Development, *Inform. Systems,* **4,** 1-12 and 93-118.
[36] M Lundeberg, G Goldkuhl and A Nilsson, [1981], *Information Systems Development,* Prentice-Hall, Englewood Cliffs, New Jersey.
[37] J Martin, [1985], From Analysis to Design, *Datamation,* **31**(18), 129-135.
[38] V A Mastro, [1985], Three Dimensional System Development, *ACM SIGSOFT SE Notes,* **10**(5), 47-59.
[39] K Matsumura, H Mizutani and M Arai, [1987], An Application of Structural Modeling to Software Requirements Analysis and Design, *IEEE Trans Soft Eng,* **13,** 461-471.
[40] G McFarland, [1986], The Benefits of Bottom-UP Design, *ACM SIGSOFT SE Notes,* **11**(5), 43-51.
[41] L J Mekly and S S Yau, [1980], Software Design Representation Using Abstract Process Networks, *IEEE Trans Soft Eng,* **6,** 420-435.
[42] J Mende, [1986], Research Directions in Information Systems, *Quaestiones Informaticae,* **4**(1), 1-4.
[43] J Mende, [1986], Laws and Techniques of Information Systems, *Quaestiones Informaticae,* **4**(3), 1-6
[44] J Mende, [1987], Three Packaging Rules for Information System Design, *Quaestiones Informaticae,* **5**(3), 32-35.
[45] J Mende, [1988], A Structural Model ofnfman ystems Theory, *Quaestiones Informaticae,* **6**(1), 28-32.
[46] J Mende, [1988], A Classification of Partitioning Rules for Information System Design, *Quaestiones Informaticae,* **6**(2), 63-66.
[47] J Mende, [1989], Four Major Success Criteria for Information System Design, *S.A. Comp. J,* **1** [to appear].
[48] H D Mills, R C Linger and A R Hevner, [1987], Box structured information systems, *IBM Syst J,* **26,** 395-413.
[49] I Miyamoto and R T Yeh, [1981], A software requirements analysis and definition methodology for business data processing, *Proc. AFIPS NCC,* 571-581.
[50] G J Myers, [1978], *Composite/Structured Design,* Van Nostrand Reinhold, New York.
[51] P G Neumann, [1986], On Hierarchical Design of Computer Systems for Critical Applications, *IEEE Trans Soft Eng,* **12,** 905-920.
[52] K W Nielsen and K Shumate, [1987], Designing Large Real-time Systems with ADA, *Comm. ACM,* **30,** 695-715.
[53] K T Orr, [1977], *Structured Systems Development,* Yourdon Press, New York.
[54] K T Orr, [1981], *Structured Requirements Definition,* Ken Orr & Associates, Topeka, Kansas.
[55] D L Parnas, [1972], A Technique for Software Module Specification with Examples, *Comm. ACM,* **15,** 330-336.
[56] D L Parnas, [1972], On the Criteria to be Used in Decomposing Systems into Modules, *Comm. ACM,* **15,** 1053-1058.
[57] D L Parnas and D P Siewiorek, [1975], Use of the Concept of Transparency in the Design of Hierarchically Structured Systems, *Comm. ACM,* **18,** 401-408.
[58] D L Parnas, [1976], On the Design and Development of Program Families, *IEEE Trans Soft Eng,* **2,** 1-9.
[59] D L Parnas, [1979], Designing Software for Ease of Extension and Contraction, *IEEE Trans Soft Eng,* **5,** 128-138.
[60] D L Parnas, P C Clements and D M Weiss, [1985], The Modular Structure of Complex Systems, *IEEE Trans Soft Eng,* **11,** 259-266.
[61] J Post, [1986], Application of a structured methodology to real-time industrial software development, *Soft Eng J,* **1,** 222-235.
[62] B Randell, [1986], System Design and Structuring, *The Computer Journal,* **29,** 300-306.
[63] K A Robinson, [1979], An entity/event data modeling method, *The Computer Journal,* **22,** 270-281.
[64] C J Rosenquist, [1982], Entity Life Cycle Models and their Applicability to Information Systems Development Life Cycles, *The Computer Journal,* **25,** 307-315.
[65] D T Ross, [1977], Structured Analysis [SA]: A Language for Communicating Ideas, *IEEE Trans Soft Eng,* **3,** 16-34.
[66] D T Ross, [1985], Applications and Extension of SADT, *IEEE Computer,* **18**(4), 25-34.
[67] D T Ross and K E Schoman, [1977], Structured Analysis for Requirements Definition, *IEEE Trans Soft Eng,* **3,** 6-15.

[68] S Rotenstreich and W E Howden, [1986], Two-Dimensional Program Design, *IEEE Trans Soft Eng,* **12,** 377-384.
[69] A Ritter, [1987]; Highlights of the New Approach for Structuring Complex *Systems, ACM SIGSOFT SE Notes,* **12**(2) 38-42.
[70] R I Rudkin and K D Shere, [1979], Structured Decomposition Diagram: A New Technique for System Analysis, *Datamation,* **25**(10), 130-146.
[71] P A Scheffer, A H Stone and W E Rzepka, [1985], A Case Study of SREM, *IEEE Computer,* **18**(4), 47-54.
[72] S M Shatz and J-P Wang, [1987], Introduction to Distributed-Software Engineering, *IEEE Computer,* **20**(10), 23-31.
[73] G E Sievert and T A Mizell, [1985], Specification-Based Software Engineering with TAGS, *IEEE Computer,* **18**(4), 56-65.
[74] H Simpson, [1986], The Mascot method, *Soft Eng J,* **1,** 103-120.
[75] H R Simpson and K Jackson, [1979], Process Synchronisation in MASCOT, *The Computer Journal,* **22,** 332-345.
[76] W P Stevens, [1981], *Using Structured Design,* John Wiley, New York.
[77] W P Stevens, [1982], How data flow can improve application development productivity, *IBM Syst J,* **21,** 162-178.
[78] W P Stevens, G J Myers and L L Constantine, [1974], Structured design, *IBM Syst J,* **13,** 115-139.
[79] T H Tse and L Pong, [1986], An Application of Petri Nets in Structured Analysis, *ACM SIGSOFT SE Notes,* **11**(5), 53-56.
[80] J W Verity, [1987], The OOPS Revolution, *Datamation,* **33**(9), 73-78.
[81] P T Ward, [1986], The Transformation Schema: An Extension of the Data Flow Diagram to Represent Control and Timing, *IEEE Trans Soft Eng,* **12,** 198-210.
[82] J D Warnier, [1974], *Logical Construction of Programs,* Van Nostrand Reinhold, New York.
[83] A I Wasserman, P A Pircher, D T Shewmake and M L Kersten, [1986], Developing Interactive Information Systems with the User Software Engineering Methodology, *IEEE Trans Soft Eng,* **12,** 326-345.
[84] H Weber and H Ehrig, [1986], Specification of Modular Systems, *IEEE Trans Soft Eng,* **12,** 784-798.
[85] V Weinberg, [1980], *Structured Analysis,* Prentice-Hall, Englewood Cliffs, New Jersey.
[86] N Wirth, [1971], Program Development by Stepwise Refinement, *Comm. ACM,* **14,** 221-227.
[87] B I Witt, [1985], Communicating Modules: A Software Design Model for Concurrent Distributed Systems, *IEEE Computer,* **18**(1), 67-76.
[88] B I Witt, [1985], Parallelism, Pipelines and Partitions: Variations on Communicating Modules, *IEEE Computer,* **18**(2), 105-112.
[89] E Yourdon, [1986], Whatever Happened to Structured Analysis? *Datamation,* **32**(11), 133-138.
[90] E Yourdon and L L Constantine, [1979], *Structured Design,* Prentice-Hall, Englewood Cliffs, New Jersey.
[91] P Zave, [1982], An Operational Approach to Requirements Specification for Embedded Systems, *IEEE Trans Soft Eng,* **8,** 250-269.
[92] P Zave, [1984] The Operational versus the Conventional Approach to Software Development, *Comm. ACM,* **27,** 104-118.

Computers and the Law

Submitted by Antony Cooper
CSIR

The SA Law Commission has established a commission on "The Legal Protection of Information".

The commission is still in its preliminary stages and the assigned researcher, Mr Herman Smuts, is still preparing the working paper. He does not know when it will be finished, but once the working paper has been prepared, they will invite comments for about two years, before preparing the final report. I have contacted Mr Smuts, and he would be most grateful to receive input at this stage, especially regarding the terms of reference of the commission. His address is:

C/o SA Law Commission
Private Bag X668
PRETORIA
0001

In addition, there is an ad-hoc committee at the Registrar of Copyright investigating numerous copyright issues, including those relating to software and data. Mr Smuts' commission will be liaising with the ad-hoc committee.

I feel that SAICS has an obligation to submit evidence to the commission, and I would appreciate it if you would circulate the members of the Council of SAICS, and perhaps the general membership as well, to solicit ideas concerning SAICS's input.

I shall prepare something for the commission, either in my personal capacity, or in my professional capacity here at CSIR. I would be willing to assist in the preparation of any evidence SAICS might submit.

4th National MSc/Phd Computer Science Conference

Report by Danie Behr
University of Pretoria

This conference was held from 7th to 10th September 1989 at the Cathedral Peak Hotel in the Drakensberg. The conference was attended by 61 postgraduate students from 11 South African universities. Most were engaged in MSc studies, although 5 Phd students also attended. These numbers are encouraging for the South African computer science community. This type of conference is rather unique in that it affords students the opportunity of sharing their research, and getting to know other researchers in the country. The number of Afrikaans and English speaking students attending the conference were roughly equal. Presentations were made in the language preferred by the student. Invitations were sent to all universities with computer science departments. The conference was organized by the students themselves.

Some of the more popular research topics that were presented included expert systems, data communications, computer security, graphics, software engineering, user interfaces and data bases. The main sponsor for this year's conference was the Division for Microelectronic Systems and Communication Technology of the CSIR. The conference was opened with an interesting talk on the myths and motivations of post graduate studies by Prof DG Kourie, acting head of the Computer Science Department at Pretoria University.

The next conference will be presented by the University of Port Elizabeth. People requiring further information about the next conference should contact Andre Calitz, Charmaine du Plessis or Jean Greyling of the Department of Computer Science at UPE.

A list of authors and papers presented at the symposium follows:

S Crosby, University of Stellenbosch
Performance Analysis of Wide Area Computer Communication Networks

A B Joubert, PU for CHE Vaal Triangle Campus
Image Processing Libraries

A Calitz, University of Port Elizabeth
An Expert System Toolbox to assist in the classification of objects

L von Backström, University of Pretoria
Integrated Network Management

R Foss, Rhodes University
The Rhodes Computer Music Network

A McGee, University of Natal
On Fixpoints and Nondeterminism in the Sigma-Lambda Calculus

P G Mulder, Randse Afrikaanse Universiteit
A Formal Language and Automota approach to Data Communications

A Tew, Randse Afrikaanse Universiteit
Drie dimensionele grafiek grammatikas

T C Parker-Nance, University of Port Elizabeth
Human-Computer Interaction: What Determines Computer Acceptance

E Coetzee, PU vir CHO Vaaldriehoekkampus
Opsporing van rande in syferbeelde dmv verskerping en drempelbepaling
D A Sewry, Rhodes University
Visual Programming
A Cooper, University of Pretoria
Improvements to the National Exchange Standard
E S Badier, University of Port Elizabeth
A Computer Assisted Diagnostic System (CADS)
C du Plessis, Universiteit van Port Elizabeth
Persoonsidentifikasie dmv naampassing in 'n genealogiese databasis
J Greeff, University of Stellenbosch
The Entity-Relationship Model and its Implementation
D A de Waal, PU vir CHO
Flat Concurrent Prolog (FCP) en Flat Guarded Horn Clauses (FGHC): 'n Vergelyking
E Naude, UNISA
Interne metodes in Liniêre Programmering
A Deacon, University of Stellenbosch
Global consistency in non-locking DDBMS
A Wilks, Rhodes University
The Synchronisation and Remote Configuration of the Resources in a Computer Music Network
J Greyling, University of Port Elizabeth
The design of a User Interface with special reference to an Interactive Molecular Modelling Program
L Drevin, PU vir CHO
Rekenaarsekuriteit: Verskillende vlakke van kontrole
Dieter C Barnard, University of Stellenbosch
The design and implementation of a modest, interactive proof checker
R A Schmidt, University of Cape Town
Knowledge Representation Systems and the Algebra of Relations
J Hartman, Randse Afrikaanse Universiteit
Die Gebruik van Objek-georiënteerde Programmering in die Moderne Sneltrein Omgewing
S Lawrie, Rhodes University
The Design and Implementation of a System for the Interactive Control of a MIDI-based Studio
E Mulder, Rand Afrikaans University
A Formalisation of Object-Oriented Principles
C J Tolmie, UOVS
Die Ontwikkeling van 'n Ekspertrekenaarstelsel vir die beoordeling van die resultate van die Technicon H1-Bloedselanaliseerder
R Breedt, University of Pretoria
Realism with Ray Tracing
J van Jaarsveld, University of Pretoria
Developing Medical Expert Systems: A knowledge acquisition perspective
W Appel, University of Pretoria
TCP/IP Implementation on Ethernet
E Goedeke, University of Natal
Eggspert's Control Structure
M Harmse, University of Stellenbosch
Modelling of I/O Subsystems
H L Viktor, University of Stellenbosch
A Quantitative Model for Comparing Recovery Techniques in a Distributed Database
M Olivier, Randse Afrikaanse Universiteit
Rekenaarvirusse in Suid-Afrika

Book Reviews

An Introduction to Functional Programming Through Lambda Calculus
by Greg Michaelson, Addison-Wesley, 1988.
Reviewer: Dr. E P Wentworth, Rhodes University

Recently we have seen a number of excellent *second generation* texts on Functional Programming. Michaelson's text assumes some previous programming experience with imperative languages, and presents the functional approach as an alternative paradigm. He begins with a very accessible exposition of the Lambda Calculus, and carefully develops this foundation to encompass the important aspects and paradigms of functional programming. The programming notation is language-independent, although the last chapters are devoted to a brief look at two specific languages, Standard ML and Lisp. The examples and exercises are mainly utility in nature, e.g. "insert a sublist after the first occurrence of another sublist in a list", and can generally be solved in a couple of lines. Answers to the exercises are provided in an appendix.

The approach is slanted towards developing a solid base for understanding functional languages and computing. In this respect the book achieves a good balance between the theoretical underpinnings and their practical application. On the practical side, however, I found the lack of more substantial examples and exercises disappointing. Most programming texts tackle a set of 'standard' problems which are well-understood in the academic community and provide an informal benchmark for comparisons. Since the book is targeted for those already versed in imperative languages and standard algorithms, one might expect the examples to clearly demonstrate the elegance and power of the *problem-oriented* functional approach in these areas. Having laid an excellent foundation I was left with the feeling that the book failed to capitalize and deliver the cherry on the top.

The book is highly recommended as one of the new breed of Computer Science books which gives substantial attention to the fundamentals of the subject without becoming bogged down in over-rigorous formality.

Artificial Intelligence and the Design of Expert Systems
by George F Luger & William A Stubblefield, The Benjamin/Cummings Publishing Co., 1989.
Artificial Intelligence: A Knowledge-based Approach
by Morris W Firebaugh, PWS-Kent Publishing Co., 1989.
Reviewer: Prof G D Oosthuizen, University of Pretoria

One of the primary goals of an Honours course is to introduce students to a field in such a way that they arrive at enough insight into relevant issues to enable them to conduct further research on their own. To this end a text book which is used ought to reflect the current view of the field. Because of the rapid expansion of the field of Artificial Intelligence (AI), we have now finally outgrown the era dominated by the books by Winston and Charniak and McDermott. In the past five to ten years much new work has been done, and new insights have been gained. Introducing AI, therefore, requires a marked shift from the previous emphasis on a few historical systems embodying a number of famous methods, to a more generic approach - an approach which highlights those fundamental representation and search models that span all the different application areas and strategies of problem solving. Of course, since AI still does not have a well developed theory, references to seminal systems continues to fulfil an important role.

Both of the above books are good text books, characterised by a balanced coverage of Prolog and Lisp. They also reflect and consolidate much of the work of the past few years done in areas such as knowledge representation, machine learning, the work done under the heading of Expert Systems and even the recent work on neural networks. But the most important feature that they share is the accurate and up to date overall picture of the subject provided; the broad framework for the understanding of AI that is created without neglecting work of historical importance. There are still references to these works, but they are placed in perspective in relation to new developments.

The book of Luger & Stubblefield (L&S) is more language oriented than Firebaugh's book. A characteristic of L&S is that AI approaches to representation are related to the Object Oriented approach. Whereas L&S includes chapters on advanced AI programming techniques in Prolog and Lisp, it does not address pattern recognition, computer vision and robotics. (Firebaugh has chapters on each of these themes.) These omissions are understandable, since AI has diversified so much recently that it is difficult to cover all applications in one book.

If I had to select one of the books, it would be L&S. Although L&S gives poor coverage of Machine Learning, the book's overall presentation is very good. In particular, the chapters are well-organised, and the overall approach to AI - starting with the core aspects of *representation* and *search*, followed by chapters on AI languages - is coherent. The authors also make very good use of graphical representations and illustrations to convey ideas.

Books Received

The following books have been sent to SACJ. Anyone willing to review a book should contact the editor. The book will be sent to him for review, and may be kept provided that a review is received.

- D Bustard, J Elder & J Welsh, [1988], *Concurrent Program Structures*, Prentice-Hall Inc., Englewood Cliffs.
- R Cafolla & A D Kauffman, [1988], *Turbo Prolog Step by Step*, Merrill Publishing Company, Columbus, Ohio.
- S Hekmatpour, [1988], *Introduction to LISP and Symbol Manipulation*, Prentice-Hall Inc., Englewood Cliffs.
- K L Clark & F G McCabe, [1984], *micro-PROLOG: Programming in Logic*, Prentice-Hall Inc., Englewood Cliffs.
- D Crookes, [1988], *Introduction to Programming in Prolog*, Prentice-Hall Inc., Englewood Cliffs.
- M J C Gordon, [1988], *Programming Language Theory and its Implementation*, Prentice-Hall Inc., Englewood Cliffs.
- J G Hughes, [1988], *Database Technology : A software engineering approach*, Prentice-Hall Inc., Englewood Cliffs.
- R Milner, [1989], *Communication and Concurrency*, Prentice-Hall Inc., Englewood Cliffs.
- T J Myers, [1988], *Equations, Models and Programs*, Prentice-Hall, Inc., Englewood Cliffs.
- N C Rowe, [1988], *Artificial Intelligence through Prolog*, Prentice-Hall Inc., Englewood Cliffs.
- D A Protopapas, [1988], *Microcomputer Hardware Design*, Prentice-Hall Inc., Englewood Cliffs.
- H Eisner, [1988], *Computer-aided Systems Engineering*, Prentice-Hall Inc., Englewood Cliffs.
- S H Unger, [1989], *The essence of logic circuits*, Prentice-Hall Inc., Englewood Cliffs.
- R J Young, [1989], *Practical Prolog*, Van Nostrand Reinhold, New York.

NOTES FOR CONTRIBUTORS

The prime purpose of the journal is to publish original research papers in the fields of Computer Science and Information Systems. However, non-refereed review and exploratory articles of interest to the journal's readers will be considered for publication under sections marked as a Communications or 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 double-space typing on one side only of A4 paper, and provide wide margins.
- The first page should include:
  - title (as brief as possible);
  - author's initials and surname;
  - 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 on separate sheets of A4 paper, and should be numbered and titled. Figures should be submitted as original line drawings, and not photocopies.
- Mathematical and other symbols may be either handwritten or typed. Greek letters and unusual symbols should be identified in the margin. Distinguish clearly between such cases as:
  - upper and lower case letters;
  - the letter O and zero;
  - the letter I and the number one; and
  - the letter K and kappa.
- 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 in square brackets. References should thus take the following form:

  [1] E Ashcroft and Z Manna, [1972], The translation of 'GOTO' programs to 'WHILE' programs, *Proceedings of IFIP Congress 71*, North-Holland, Amsterdam, 250-255.

  [2] C Bohm and G Jacopini, [1966], Flow diagrams, Turing machines and languages with only two formation rules, *Comm. ACM*, **9**, 366-371.

  [3] S Ginsburg, [1966], *Mathematical theory of context free languages,* McGraw Hill, New York.

Manuscripts *accepted* for publication should comply with the above guidelines, and may provided in one of the following three formats:

- in a **typed form** (i.e. suitable for scanning);
- as an **ASCII file** on diskette; or
- in **camera-ready** format.

A page specification is available on request from the editor, for authors wishing to provide camera-ready copies.

Charges

A charge per final page, scaled to reflect scanning, typesetting and reproduction costs, will be levied on papers accepted for publication. The costs per final page are as follows:

Typed format:	R80-00
ASCII format:	R60-00
Camera-ready format :	R20-00

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

Proofs

Proofs of accepted papers will 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.

Note that, in the case of camera-ready submissions, it is the author's responsibility to ensure that such submissions are error-free. However, the editor may recommend minor typesetting changes to be made before publication.

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 specialized production costs which will be borne by the advertiser. Enquiries should be directed to the editor.

South African Computer Journal

Number 1, January 1990
ISSN 1015-7999

Suid-Afrikaanse Rekenaar-tydskrif

Nommer 1, Januarie 1990
ISSN 1015-7999

Contents