SHOULD DATA STRUCTURES BE TAUGHT IN (IS) PROGRAMS?

Dr. Balakrishnan Muthuswamy
Applied Computer Science Department
Illinois State University
Normal, IL 61761

ABSTRACT: The focus of most academic Information Systems (IS) programs is to teach computer usage, systems analysis and design, application programming, and business management concepts. The focus of such programs is to train IS students to analyze organizational systems and understand user needs.

The job functions of systems analysts in today's organizations are demanding. They need to understand changing user requirements as well as how to maintain existing systems. Additionally they must be able to develop innovative applications integrating new technologies such as Neural Networks, Expert Systems, Distributed Processing, Distributed Artificial Intelligence, and CASE tools. Often they will have to implement prototypes using traditional development approaches. Further, organizational users today (and more so in future) are themselves highly computer literate, and may be experts in specific areas of computing.

To cope with these increasing demands, IS professionals need to have an understanding of the technical aspects of computing technology. In this paper we focus on a particular aspect of software technology, namely data structures. To demonstrate the need for a data structures course in IS programs, we examine situations where knowledge of data structures is needed by the IS professional. We also provide some high level directions for developing a data structures course appropriate for IS students.

KEYWORDS: Information Systems, Data Structures, Systems Analysis and Design, IS Analyst/ Designer.

INTRODUCTION

A course on Data Structures is a core course in most undergraduate Computer Science (CS) degree programs. The goal of such programs is to provide a coherent and broad-based coverage of the discipline of computing [5], and to impart good programming and system design skills. The goal of most undergraduate **Information Systems**[1] (IS) degree programs is to develop good application designers and programmers. These programs focus on application programming and topics related to systems analysis and design. According to ([3], source [18]) there are 504 different colleges and universities that offer IS programs; CIS and MIS are the most popular program names; and a majority (about 66%) of these are in colleges with fewer than 5000 undergraduates. [8] lists 422 colleges and universities in the United States that offer IS programs.

In general, IS programs do not include a course on data structures. Only about 15% of the 422 departments (in US universities and colleges) that offer IS programs have at least one faculty member with 'data and file structures' as a primary area of teaching interest (see Table 1).

IS programs vary according to whether they reside in a Business or non-business school/college. Non-business IS programs tend to require more technical

[1]In this paper we treat the terms IS, Computer Information System (CIS), and Management Information System (MIS) synonymously. Similarly, we consider "systems analyst/designer" to be synonymous with "IS professional", though there are other IS functions such as Database Administrator and Business Analyst.

Table 1: SCHOOLS THAT HAVE FACULTY WITH DATA STRUCTURES AS A PRIMARY AREA OF TEACHING INTEREST[2]

Number of departments (schools) in United States that have an IS program :	442
Number of these departments that have at least one faculty member with IS02 as primary area of teaching interest :	66
Breakdown:	
Number of departments that have 1 faculty with IS02 as a primary area of teaching interest	54
Number of departments that have 2 faculty with IS02 as a primary area of teaching interest	7
Number of departments that have 3 faculty with IS02 as a primary area of teaching interest	4
Number of departments that have 4 faculty with IS02 as a primary area of teaching interest	0
Number of departments that have 5 faculty with IS02 as a primary area of teaching interest	1
Total	66

courses and less management courses than their business counterparts [3]. CEO's and Systems Managers from large organizations consider technical skills as one of the major skills needed in their MIS staffs, yet they don't feel MIS/CIS graduates are receiving the necessary training in their undergraduate education [13]. Sharifi et al., [21] state that in the area of accounting, employers are demanding that their professionals have adequate backgrounds in computer science and information systems. At least one study reports that in the opinion of MIS graduates, more technical education would help them to better compete in the job market [20].

The central theme of this paper is to motivate the need, and more importantly the direction, for a data structures course in IS programs. Several schools (see Table 1) already have an IS data structures course. However, it is the experience of this author that such courses are often taught similar to a CS data structures course, use a CS text book, and place undue emphasis on the programming of data structures. In fact, data structure text books written for non-CS majors are few in number. We suggest that the contents of these courses be revised to orient them towards the job functions of IS professionals. In order to motivate these viewpoints we first examine the IS environment in today's organizations.

IS Environment in Today's Organizations

The functions of an IS professional are getting to be more complex and demanding. There are two main reasons for this.

More Literate Users:

With the availability of micro computers at affordable prices and training facilities at schools, colleges, and universities, present day organizational users are more computer literate [6]. They also have higher expectations of the functions provided by computer systems and demand systems that are faster, intelligent, integrated, and more user friendly. The body of knowledge relating to the IS area has expanded both in terms of specialized and generalized knowledge and skills; consequently, yesterday's computer literacy (of IS professionals) is unlikely to meet today's needs [13].

Applications are Becoming More Complex and Use New Technologies:

Systems analysts are no longer merely involved in developing structured systems such as payroll, accounting, and inventory management. With the availability of CASE tools, developing such systems is easier and more systematic. A more difficult role of a systems analyst is in developing innovative applications. This involves integrating traditional systems with those using newer technologies. For example, developing Decision Support Systems (DSSs) involves integrating existing organizational systems with other heterogeneous systems such as Expert Systems (ESs), and Model Management Systems (MMSs). Problem solving in organizations requires the use of state-of-the-art technologies such as Neural Networks, Distributed Processing, Object Oriented Modeling, Expert Systems, and Artificial Intelligence. The traditional systems development function needs to be redesigned and retooled to support organizational change related to information technology and use [7].

To cope with the user demands, successful IS professionals need to gain expertise in using new technologies and even foster their development. Because of the emphasis on performance in information systems and the increasing complexity of applications, a systems analyst/designer must understand the logical structure and physical structure of both programs and data [16]. The latest DPMA Model Curriculum [10] recommends knowledge of data structures at some level between 'understanding concepts' and 'detailed understanding'. We assume that understanding and developing applications using state-of-the-art technologies is facilitated by an understanding of data structures and other topics from the

[2]Source: [8].

Note 1: IS02 is the ACM term for "data and file structures" course.

Note 2: [8] is published from information provided by faculty and since some faculty (and schools) might not have provided this information, there may be other schools that offer IS programs.

Note 3: Most of the departments in the list were in the College of Business. The department with 5 faculty members with data structures as primary teaching area was a "computer technology" department.

foundational areas of systems theory, computer science, and management.

Our suggestion that a course (or part-course) on data structures be included in IS programs is in line with the DPMA Model Curriculum recommendations [10]. Compared to their previous model curriculum [9], the present model curriculum [10] generally has more CS concepts included in the 'Information Systems Body of Knowledge'. Our suggestion is directed at programs that produce future system analysts/designers.

We suggest that a data structure course for IS majors should be application oriented and taught using examples and cases from IS job functions. It should focus on problem solving and treat data structures as tools to achieve specific goals. Teaching data structures using a traditional CS approach is inappropriate because most IS students lack the mathematical and programming skills of CS students. Developing course material for an IS data structures course is a challenge. In the seventies it was a challenge to teach database concepts to IS students in an application oriented manner without using the mathematical vigor of a CS database course. A similar effort is necessary to teach data structures to IS students.

HISTORY OF IS AND IS CURRICULA

Computer scientists by training are good at the software and hardware aspects of systems, but are not trained in communication, group management, and application domain areas such as economics, finance, accounting, operations research, marketing, and management. In the sixties and seventies computer scientists were the primary players in developing organizational systems. They focused on system performance, hardware, and software considerations.

The resulting systems performed the computations that they were designed for, but were not user friendly and did not satisfy user needs. Users tended to either 1) not use the system, 2) reluctantly use the system due to lack of a better alternative, 3) require extensive training to use the system, or 4) move to other jobs to avoid using bad systems.

Consequently, there was a lot of research attention on the issue of 'success' of systems and on designing successful systems (for example [2, 22]). The recommendations that followed may be summarized in one sentence: 'get the users involved in the systems analysis and design process, and bridge the gap between the user and the designer.'

> ***...a data structure course for IS majors ... should focus on problem solving and treat data structures as tools to achieve specific goals.***

This led to the evolution of the discipline of IS. Topics such as data flow diagrams, systems development life cycle, modular design of systems, and user interface design received increased attention. Several universities and colleges established programs to teach this new discipline. The goal of these programs was to provide students (future IS professionals) with a comprehensive understanding of topics from CS, systems theory, communication, and business administration. The role of such professionals was (and is) to bridge the gap between computer scientists/programmers and business managers/users. Today, several universities have successful IS programs and the industry has been recruiting IS graduates for a variety of career positions.

During the past decade IS faculty were in short supply. Most IS faculty (generally not trained in CS topics) were interested in teaching COBOL, file processing, spreadsheet usage, database design, systems analysis and design and other application related areas. Consequently, an important factor controlling the content of CS topics in IS programs has been the background of the faculty, rather than an objective assessment of the skills required to perform IS job functions. Therefore, some IS programs had a course on data structures while others did not.

This, however, is changing. In the eighties several universities developed IS doctoral programs that provide students with a balanced understanding of IS and CS topics, and the opportunity to specialize in specific areas. The number of faculty in IS departments who have received such training is increasing. This change in the background of faculty in IS departments has an effect on the courses offered. Several universities (for example see [4]) have recognized the need to include more CS content in IS programs and are including such courses as data structures, networks and telecommunications, expert systems, artificial intelligence, and object oriented programming. Having discussed the history of IS, in the following section we examine the role of data structures in IS professions.

ROLE OF AND NEED FOR DATA STRUCTURES IN IS CURRICULUM

In this section we discuss the relationship between data structures and IS functions such as programming, data base management, software design and maintenance, innovative application development and distributed processing. The objective is to suggest that knowledge of data structures is useful and perhaps necessary to cope with the job functions of today's IS professionals.

Data structures are an extension to programming concepts

One of the functions of IS professionals is application programming [10]. It is difficult to understand how one could analyze and design complex systems and develop good application programs without good programming skills. Problem solving is so domain specific that it is best integrated into the existing disciplines [15]. Good programming and problem solving requires knowledge of data structures. Programs are concrete formulations of abstract algorithms based on particular representations and structures of data [23]. An understanding of data structuring is crucial to a successful programmer;

algorithms and data structures are interdependent elements in any problem-solving context [14]. The degree to which the appropriate selection and use of data structures affects design and quality of programs varies with the application; however, without a sufficient understanding of data structures, the general level at which a programmer can work will be severely limited [11].

An IS data structures course should focus on teaching concepts related to the use of data structures (such as arrays, trees, link lists, stacks, and queues) in application development, and associated topics such as analysis of algorithms, complexity, recursive programming, and abstract data types. These concepts are a natural addition to basic programming constructs such as sequence (flow of program control), control (IF-THEN-ELSE), repetition (DO-LOOP), case (selection between several alternatives), modularity (subprocedures), and controlling input/output of data, which are usually taught in an earlier course. Teaching programming without teaching data structures, to use an analogy, is like teaching a language's (like English or French) words, spelling, and some grammar, but not the phrases, idioms, letter-writing conventions, synonyms and antonyms as well as other language details that are so essential to communicate properly.

Databases and data structures

It is true that some relational database systems provide higher levels of 'data independence', that is, the programs and data are less dependent on one another; and navigation within a database is application independent. This means that software developers need not understand the data structures that underlie the higher-level structures. However, knowledge of data structures is helpful to understand the underlying processing of data base operations (such as, the Bill of Materials processing in IDMS or DB2 [17], a 'GET NEXT child' in IMS database, or join-indexing in a relational database), and important for designing DBMS procs (procedure CALLS), query processing management, and integrity verification.

These operations involve (or use concepts of) sorting, searching, stacks, trees, transitive closure, complexity, and recursive procedures. Without an understanding of these techniques, an IS professional cannot claim expertise to perform such functions as designing databases, designing application programs, maintaining integrity through triggered database procs, and integrating databases with other organizational systems. Further, there may be occasions when the basic structures provided by a DBMS, a file system, or an operating system are inappropriate to the task in hand, and the software developer needs to revert to first principles [12].

Initiating, understanding, designing, and implementing changes to an existing system is the responsibility of systems analysts. This requires understanding the real world changes and the existing system processing.

Knowledge of data structures is also important to design systems at a logical level of abstraction, while considering lower level factors such as (physical) operations, and performance (complexity) of the application programs. Another design activity in systems development is evaluating database software and system hardware alternatives. These are all within the domain of responsibilities of such IS positions as systems analyst or a Data Base Administrator (DBA).

Designing and evaluating programs and software

A systems analyst designs software modules and develops pseudo-code. Selecting between alternative software designs requires estimating their performance. Similarly an IS professional has to make decisions regarding software and hardware alternatives during the feasibility, analysis, and design stages of systems development. Such evaluation requires computing performance estimates such as average retrieval time, storage costs, and how performance can be improved or fine tuned. These activities require knowledge of data structures and related concepts.

Maintaining existing applications

Existing computer programs in organizations may use data structures. Most programs require maintenance to reflect changes in the real world or to improve the efficiency of information processing. Maintaining existing programs and integrating new systems with existing systems is an important activity to enable continued data processing and user support, or to reuse existing code in new applications. An extreme case of maintenance is a switch over to a new environment. A common example is the case when an organization switches from a non-database environment to a database environment. In larger organizations, this may take several years.

Systems analysts have to understand existing methods of processing (which may use data structures), develop new/better methods, manage the impact of the changes on user procedures, and actually design the new system. Initiating, understanding, designing, and implementing changes to an existing system is the responsibility of systems analysts. This requires understanding the real world changes and the existing system processing. If the current system uses data structures in its programs, the systems analyst has to know data structures to understand the working of such systems. Also, understanding the details of current systems is an important step in reverse engineering.

Knowledge of data structures is needed to develop innovative applications

We have already discussed the increasing complexity of IS environment in today's organizations. Users are becoming more computer literate and technologies are rapidly emerging and changing. These changes have had an

impact on the role of the systems analyst. The job functions of today's IS professionals are increasingly demanding.

An important role of a systems analyst/designer is to use new technologies (such as Expert Systems, Neural Networks, and Distributed Artificial Intelligence) to develop innovative applications. This involves using different shells and/or languages and integrating them. Developing such applications requires extensive programming skills and use of data structure concepts. Proficiency in such skills would enable IS professionals to be creative and successful in developing innovative prototypes in a cost and time efficient manner, without having to depend on a computer science professional for developing a prototype, or the availability of expensive prototyping tools.

For example, developing, designing, and prototyping a Decision Support System involves working with databases, expert systems, and model management systems. Let us just take the case of an expert system. Processing knowledge involves the use of basic data structures, for instance, in a (rule based) production system the order in which the rules are modeled may be important. The inferencing mechanisms (backward chaining, forward chaining) that these systems use are similar to processing data represented in the form of a tree. Knowledge of graphs, trees, and associated processing algorithms would be useful in this context.

Distributed Processing

In distributed database/network design some of the important decision issues that a systems designer/manager encounters are problems that relate to system response time. Understanding and solving these problems requires knowledge of methods of file allocation, file fragmentation, query processing and optimization, knowledge of distributed artificial intelligence concepts and algorithms, managing network traffic (queues), and integrity maintenance. These functions are not provided by a data base management system, nor by the vendor of a communication package. These functions are the responsibility of the database/ network administrator and he/she must understand the basic concepts involved, even if he/she may be assisted by computer science personnel trained in certain specialized areas.

In summary, the job requirements of today's IS professionals are increasing in complexity and requires interdisciplinary knowledge/capability to develop expertise in emerging technologies and methodologies. In order to cope with these demands, it is important that future systems analysts receive better training on CS topics. With this objective we suggest that IS programs include a course or part-course on data structures as a core requirement. In the next section we provide some general guidelines for developing such a course.

DIRECTIONS FOR A DATA STRUCTURES (PART) COURSE FOR IS STUDENTS

A data structures course (using a language such as Pascal, or C) is part of most undergraduate CS degree programs. The contents of this course has not changed much in the last two decades except for the inclusion of some new algorithms. This is perhaps because data structures are so fundamental (like linear algebra) to CS professionals that they cannot afford not to teach it. The job functions of a CS professional involves activities associated with system programming and application program implementations. These demand detailed implementation level knowledge of data structures.

On the other hand, for IS professionals, data structures are mere tools for problem analysis, representation, and solution development. They are not interested in each minute detail of the tool, but in knowing the practical applications, programming concepts related to the tools, advantages and limitations of each tool, the circumstances in which they may used, and how to modify the tools to the needs of the problem at hand. A data structures course for CIS should be treated as an intermediate-level programming course that precedes an in-depth course on DBMS [11].

Teaching data structures to IS students should consider the differences between the job functions of IS and CS professionals. Another important factor that should be considered in the design of IS data structure courses is the background of IS students. They often do not have strong mathematical skills, but have knowledge of business domains, and organizational, communication, and problem modeling skills.

We suggest that a realistic way to teach data structures to IS students is to expose them to applied (organizational) problems, different types of organizational data, and introduce data structures as techniques to organize and process the data. The depth of the material on data structures need not be as detailed as in CS programs, but should focus on applying data structures to model business applications using data structures as abstract data types. The course should be oriented to complement the role of IS professionals, namely, bridging the gap between technology and the user. It should be application oriented and taught using data and examples from business application domains such as finance, accounting, economics, production, and marketing.

CS students are generally taught data structures (and other subjects) in a bottom-up fashion - they learn the structures and then use it to solve problems. IS students need to be taught in a top-down manner - first the real-world problem, analysis of the problem, representation of the problem, and finally the analysis of the data structures useful to the problem. In Table 2 we present some examples for developing assignments and exercises in a course on data structures for IS students. It is not a comprehensive list of such examples. The objective in presenting this table is to motivate the direction of the IS data structures course.

Finally, teaching data structures to non-CS majors is a challenging task. Using state-of-the-art technology to develop Computer Aided Instruction packages is suggested. Also, we need to develop higher level abstractions of data structures, and provide higher level language structures to

Table 2: EXAMPLES OF ASSIGNMENTS FOR AN (IS) DATA STRUCTURES COURSE

Structure	Examples
Arrays (two dim)	Distance between cities (like in an atlas). Organizational data - Employee data in a character array. Student grades in exams, programs, and quizzes. Cost matrix in a transportation problem. Minimax and maximin data in an array.
Trees	Organization chart to explain concept of hierarchy. Decision tree using bank loan decision making with probabilities. Backtracking in and Expert System (using a set of rules). Family genealogy tree.
Link Lists	Railway train (link of wagons) example. Linking people (from different departments) working on a project. Linking sub-parts of a part. Flow of data in an organization.
Stacks	IN and OUT baskets on a manager's desk. Finding a goal in a rule based system. Inventory management - First In First Out and Last In First Out.
Queues	Barber shop example. Priority of a manager's activities. Distributed databases - sending messages.
Recursion	Retrieving sub-parts of a part from a relational database. Factorial example. Finding descendants of a person in a genealogy tree.
Process time	Time taken to retrieve, delete, insert, sort for different file structures, computations and data models. Queue delays.

manipulate, use, and manage data structures. This will enable IS professionals to use these routines without having to code the details of the processing. Innovative methods and user friendly systems (for example, see [1, 4, 19, 24]) for teaching data structures need to be developed.

CONCLUSIONS

This paper advocates a course or part course on data structures in IS programs, under the general assumption that IS professionals need an understanding of data structures to be effective systems analysts and designers. It is important that IS faculty realize the need for such a course. Teaching data structures to IS majors needs innovative approaches, cases, and examples. The emphasis should be on practical applications and not on manipulating data structures.

REFERENCES

1. J. T. Allen., H. Porter., T. R. Nanney., and K. Abernethy., "Reexamining the Introductory Computer Science Course in Liberal Arts Institutions," SIGCSE Bulletin, Volume 22, No.1, 1990, pp. 100-104.
2.. L. Alter., "How Effective Managers User Information Systems," Harvard Business Review, November-December 1976, pp. 97-104.
3. Athey., "A Comparison of Undergraduate Information Systems Programs and the DPMA Model," Interface - The Computer Education Quarterly, Vol. 10, Issue 4, Winter 1988/89, pp. 68-73.
4. A. W. Biermann., "An Overview Course in Academic Computer Science: A New Approach for Teaching Nonmajors," SIGCSE Bulletin, Volume 22, No.1, 1990, pp. 236-239.
5. Computing Curricula 1991: A Summary of the ACM/IEEE-CS Joint Curriculum Task Force Report, Communications of the ACM, June 1991, pp. 69-84.
6. J. D. Couger., E. A. Stohr., L. Drake., J. Hammitt., "Information Systems as a Major Area of Interest in Business Schools," Panel Discussion Session, Proceedings of the Ninth International Conference on Information Systems, 1988, pp. 329.
7. G. B. Davis., R. O. Mason., J. F. Rockart., and J. H. Scott., "The Future of Information Systems as an Academic Field: Your Fate in 1998," Panel Discussion Session, Proceedings of the Ninth International Conference on Information Systems, 1988, pp. 337-340.
8. Directory of Management Information Systems Faculty, McGraw-Hill Publishing Co., 1992.
9. The DPMA Model Curriculum for Undergraduate Computer Information Systems, First Edition, Data Processing Management Association, October 1985.
10. Information Systems: The DPMA Model Curriculum for a Four Year Undergraduate Degree, Data Processing Management Association, 1991.
11. R. S. Ellzey., in Preface to Data Structures for Computer Information Systems, 2nd Ed., Science Research Associates, Inc., 1989.
12. D. Giles., Editorial in special issue on Data Structures and their Algorithms, The Computer Journal, Vol. 34, No. 5, 1991, pp. 385.
13. T. Luce., "Are MIS/CIS Graduates Getting the Technical Training They Need?" Interface - The Computer Education Quarterly, Vol. 11, Issue 3, Fall 1989, pp. 50-55.
14. L. E. MacDonald., "Data Structure and Program Design," Interface -

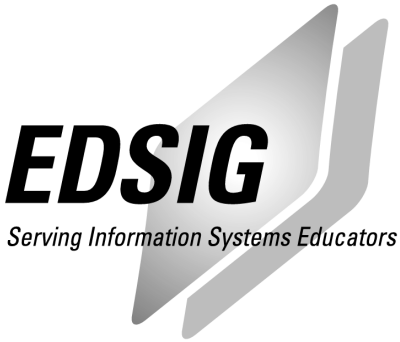
The Computer Education Quarterly, Vol. 10, Issue 3, Fall 1988, pp. 58-62.

15. D. Moursund., "Problem Solving" The Computing Teacher, Volume 16, Number 4, December/January 1988, pp. 5.
16. J. F. Nunamaker Jr., J. D. Couger, and G. B. Davis, Eds., "Information Systems Curriculum Recommendations for the 80s: Undergraduate and Graduate Programs," A Report of the ACM Curriculum Committee on Information Systems, Communications of the ACM, Vol. 25, No. 11, November 1982, pp. 781-805.
17. R. Pasley., "Physical Design for Difficult Data Structures," Database Programming and Design, Vol. 2., No. 5, May 1989, pp. 46-59.
18. Peterson's Guide to Undergraduate Study, 4-year Colleges, 1987.
19. T. W. Prat., "Teaching Programming: A New Approach Based on Analysis Skills," SIGCSE Bulletin, Volume 20, Number 1, 1988, pp. 249-253.
20. A. G. Reitsch., and F. E. Nelson., "The Use of Industry and Student Perceptions in the Redesign of an MIS Curriculum," Interface - The Computer Education Quarterly, Vol. 12, Issue 3, Fall 1990, pp. 36-40.
21. M. Sharifi., D. R. Okopny, and G. B. McCombs, "A Model Baccalaureate Degree Program in Accounting Information Systems," Interface - The Computer Education Quarterly, Vol. 12, Issue 3, Fall 1990, pp. 36-40.
22. V. J. Soden., "Understanding MIS Failures," Data Management, July 1975, pp. 29-33.
23. N. Wirth., Algorithms + Data Structures = Programs, Prentice-Hall Inc., 1976.
24. M. Zimmermann., F. Perrenoud., and A. Schiper., "Understanding Concurrent Programming Through Program Animation," SIGCSE Bulletin, Volume 20, Number 1, 1988, pp.27-31.

AUTHOR'S BIOGRAPHY

Balakrishnan Muthuswamy is a faculty in the Applied Computer Science Department, College of Applied Sciences and Technology, Illinois State University. He holds a B. Tech. in Electronics and Telecommunications from the Jawaharlal Nehru Technological University, Hyderabad, India, a MBA with a major in Marketing from the Mysore University, Mysore, India, and a Ph.D in Management Science with specialization in Management Information Systems from the University of South Carolina, Columbia. He has previously held faculty positions at the Arizona State University and The American University. His research areas include Distributed Artificial Intelligence, Expert Systems, Database Design, Distributed Systems, MIS, and Systems Analysis and Design.

Information Systems & Computing
Academic Professionals



STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 1055-3096