<table>
<thead>
<tr>
<th>South African Computer Journal</th>
<th>Suid-Afrikaanse Rekenaar-tydskrif</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number 5</td>
<td>Nommer 5</td>
</tr>
<tr>
<td>September 1991</td>
<td>September 1991</td>
</tr>
</tbody>
</table>
Regardless of the size of your computer system it is a strategic tool. We are committed to helping you use it most effectively. All it needs is a little imagination.

UNIDATA
Telephone (012) 45-1152

THERE'S A GIANT IN EVERY ONE OF US
The Department of Computer Science at Pretoria University is inviting applications for two annually awarded bursaries for MSc study. These bursaries are aimed at outstanding students who have a Computer Science Honours degree (or equivalent qualification). As far as possible, one such bursary will be awarded to a South African student, while the other will be awarded to a student from another African state. The hope is to thereby promote interaction between South African computer scientists and computer scientists elsewhere in Africa.

The bursaries are funded by the Department's contract research work and will amount to a minimum of R20000 per year. They are intended to cover costs for tuition, housing, transport, books, and an adequate allowance for other living expenses. Successful candidates will be required to work for 20 hours per week as part of the Computer Science Department’s contract research team. The work assigned to each candidate will be aimed at providing him with practical experience in his field of interest.

Study fields for which bursaries will be considered include:
- Computer Networking
- Distributed Systems
- Software Engineering
- Artificial Intelligence
- Intelligent Computer Aided Instruction
- Machine Aided Translation

Students should aim to complete the dissertation that is required for an MSc within two years of study (i.e. one year of full time study). The dissertation should preferably be written in English but it may also be written in Afrikaans.

The continuation of the bursary and adjustments to the allowance will be decided upon annually and will depend on progress. It is a condition of the bursary that, after completing his/her studies, the student should return to his/her country of origin for the same number of years for which the bursary was held.

Applicants may request application forms from:

MSc. Bursaries
The Computer Science Department
University of Pretoria
Hillcrest
Pretoria
0083
South Africa
Editor’s Notes

It is with sincere gratitude that SACJ takes leave of Dr Peter Lay who, until recently, was the assistant editor dealing with Information Systems. He has left academia for what sounds like a more gentle lifestyle. (He has gone farming!) Under Peter’s stewardship the number of high-quality IS papers in SACJ grew steadily. In general, IS papers tend to be accessible and relevant to a wide spectrum of computer professionals, and the quality of IS papers that have been appearing in SACJ has significantly contributed to the increased interest being shown in the journal by the local computer industry. If this growth in interest is to be sustained, it is urgent and important to find a suitable replacement assistant editor. The ideal candidate should not only be respected as an academic by his peers, but should also be disposed to enthusiastically promote SACJ in the private sector. Since a shortlist of candidates is currently being compiled, I would like issue a general appeal for names that might be included on it. Please contact me urgently if you would like to be considered for the job, or if you would like to nominate someone that you consider to be particularly suitable.

My three year term of office as editor expires in October. I have always considered it a great privilege to hold this position, and as a result, I felt honoured when the SAICS executive committee requested that I stay on for a further term. Nevertheless, I initially declined the request on the grounds that the time-demands of the job were significantly eroding my ability to fulfil other duties. Particularly demanding has been the task of seeing to the typesetting of the various contributions - either by doing it myself, or by ensuring that it is adequately done by someone else. Recently, however, Prof G de V Smit (Riël Smit) at UCT has offered to assume the role of production editor. This generous offer so much changes the complexion of what is being asked of me that I am now both willing and honoured to continue as editor for another term. I am very grateful to Riël for his offer and I look forward to working with him. In future, authors whose papers have been accepted for publication will be asked to liaise directly with him regarding the precise form in which the final contribution should be submitted.

The next issue of SACJ will consist largely of a selection of papers that were presented at the 6th South African Computer symposium. The selection will be based on comments from the referees who, at the time, were asked to adjudicate the papers in terms of their appropriateness for both the conference as well as for SACJ publication. Papers which, in the opinion of one or more referees, required major revision will have to be resubmitted to SACJ for refereeing purposes. Authors will soon be contact in this regard.

At the time of writing, the updated list of "approved" publications for the first half of 1991 had not yet been released by the relevant authorities. For the sake of past, present and future contributors I sincerely hope that SACJ will be on the list when it eventually comes out. However, I have become increasingly aware that there is a real danger of laying too much store on papers published in so-called approved journals as a basis for evaluating and rewarding research. I hope to expand more fully on this theme in a future edition of SACJ. Keep watching this space!

Derrick Kourie
Editor

This SACJ issue is sponsored by
a generous donation from
UNIDATA

SACJ/SART, No 5, 1991
An Efficient Primal Simplex Implementation for the Continuous 2-Matching Problem

T. H. C. Smith¹, T. W. S. Meyer², and L. Leenen²

¹Department of Computer Science, Rand Afrikaans University, P. O. Box 524, Johannesburg 2000
²Department of Computer Science, University of South Africa

Abstract

The continuous 2-matching problem (RMP2) is the relaxation of the symmetric travelling salesman problem (STSP) used by Padberg & Rinaldi to develop a highly successful branch-and-cut algorithm for the STSP. They used a standard linear program solver for solving RMP2. We note that RMP2 is a generalized network problem with additional special structure and exploit this to provide an efficient implementation of the primal simplex algorithm for RMP2. Our computational experience with the implementation demonstrates that it is several orders of magnitude faster than a standard linear program solver, suggesting that it should be worthwhile using this implementation in the Padberg-Rinaldi algorithm.

Keywords: matching, travelling salesman problem, linear programming

Computing Review Categories: G.2.2, G.4

Received September 1990, Accepted March 1991

1 Introduction

Let $G$ be a complete graph with node set $N = \{1, 2, ..., n\}$ and edge set $E = \{1, 2, ..., m\}$ where $n > 5$ and $m = n(n - 1)/2$. Each edge $e = (i, j)$ connects two distinct nodes $i$ and $j$ and has an associated cost $c_e$. The cost of a subgraph or subset of $E$ is the total cost of all the edges in the subgraph or subset.

A path is a sequence $P$ of distinct edges such that any two adjacent edges in $P$ has a common node. A cycle is a path in which each node is met by exactly two edges. The length of a path is the number of edges in it. A subgraph $G'$ is connected if for any pair of nodes in $G'$ there is a path in $G'$ connecting the two nodes. A 1-tree is a connected subgraph containing exactly one cycle while a 2-matching is a subgraph with exactly two edges meeting each node in $N$.

The symmetric travelling salesman problem (STSP) is the problem of finding a minimum cost tour (a cycle of length $n$). The 2-matching problem (MP2) is the problem of finding a minimum cost 2-matching.

MP2 can be formulated as the following integer linear programming problem:

minimize $cx$
subject to $Ax = 2$
$0 \leq x_e \leq 1$ and $x_e$ integral for all $e$ in $E$,
where $A$ is the $n$ by $m$ incidence matrix of $G$, $2$ is an $n$-vector with all elements equal to 2 and $c$ and $x$ are $m$-vectors which respectively contains the values $c_e$ and $x_e$ for all $e$ in $E$.

The continuous 2-matching problem (RMP2), obtained from MP2 by relaxing the integrality restrictions on the variables, is a relaxation of the STSP. Padberg & Rinaldi [6] use this relaxation with great success to develop a branch-and-cut algorithm for the STSP. Their algorithm first solves RMP2 with a standard linear program solver and then repeatedly identifies and adds violated STSP constraints to RMP2 which is then resolved.

Since $A$ has exactly two non-zero entries (both equal to 1) in each column, RMP2 can be viewed as a generalized network problem (see Chapter 5 in Kennington & Helgason [4]). We discuss below the exploitation of the special structure of RMP2 in an implementation of the primal simplex algorithm for RMP2. Our computational experience with such an implementation is presented, demonstrating that it is several orders of magnitude faster than a standard linear program solver on instances of RMP2.

2 Basic solutions for RMP2

The submatrix of $A$ with rows and columns corresponding to the nodes and edges in a cycle can, after possible row and column permutations, be written as follows:

$$H = \begin{bmatrix}
1 & 0 & 0 & \ldots & 0 & 1 \\
1 & 1 & 0 & \ldots & 0 & 0 \\
0 & 1 & 1 & \ldots & 0 & 0 \\
. & . & . & \ldots & . & . \\
. & . & . & \ldots & . & . \\
0 & 0 & 0 & \ldots & 1 & 0 \\
0 & 0 & 0 & \ldots & 1 & 1
\end{bmatrix}$$

It then follows from Proposition 5.1 in [4] that $\det(H) = 2 / 0$ for a cycle of odd / even length.

The submatrix of $A$ with rows and columns corresponding to the nodes and edges in a 1-tree can, after
possible row and column permutations, be written as follows:

\[ E = \begin{bmatrix} F & 0 \\ G & H \end{bmatrix} \]

where \( F \) is a lower triangular matrix with 1's on the diagonal and \( H \) is the submatrix corresponding to the nodes and edges in the 1-tree's cycle. This can be accomplished using the following algorithm (expressed in a C-like pseudo-language):

\[
i = 1;
while(1\text{-tree contains a node } j \\

teeting only one edge \( e \))
\{
\begin{align*}
&\text{let } i - th \text{ row of } E \text{ correspond to node } j; \\
&\text{let } i - th \text{ column of } E \text{ correspond to edge } e; \\
&\text{remove node } j \text{ and edge } e \text{ from } 1\text{-tree}; \\
&i++;
\end{align*}
\}

It then follows that \( |\text{det}(E)| = |\text{det}(F)|.|\text{det}(H)| = 2 / 0 \) if the 1-tree contains a cycle of odd / even length.

Consider the 1-tree consisting of the cycle with nodes 1, 2, and 3 and the path with edges \((3, 4), (4, 5), ..., (n - 1, n)\). As shown above the submatrix of \( A \) corresponding to the nodes and edges in this 1-tree is nonsingular. Therefore \( A \) contains at least one basis (a nonsingular submatrix consisting of \( n \) columns of \( A \)).

The subgraph with edge set \( E(B) \) corresponding to the columns of a basis \( B \) is called a basis graph. It follows from Proposition 5.9 in [4] that the basis graph consists of one or more 1-trees (and from the above that the cycle in any one of the 1-trees has odd length).

A basis structure \((L, U)\) corresponds to the edge set \( E(B) \) of a basis graph and is formed by partitioning the edges in \( E - E(B) \) into two sets \( L \) and \( U \). The basic solution corresponding to a basis structure \((L, U)\) is obtained by setting \( x_e = 0 \) for all \( e \) in \( L \) and \( x_e = 1 \) for all \( e \) in \( U \) and then solving \( Bx_B = b \) where the components of \( x_B \) and the columns of \( B \) correspond to the same edges and \( b = 2 \cdot \text{sum of columns in } A \text{ corresponding to edges in } U \). The basic solution is primal feasible if \( 0 \leq x_e \leq 1 \) for all \( e \) in \( E(B) \).

If \( p \) is the number of 1-trees in the basis graph then the basis \( B \) can be arranged as a block diagonal matrix with blocks \( B^1, B^2, ..., B^p \) corresponding to the 1-trees (see Proposition 5.9 in [4]). The solution of \( Bx_B = b \) is therefore given by \( x_B = (B^i)^{-1}b^i \), where \( x_B^i \) and \( b^i \) correspond to the columns and rows of \( B^i \), for \( i = 1, ..., p \).

Since \( |\text{det}(B^i)| = 2 \) each element in \( (B^i)^{-1} \) is an integer divided by 2 and it follows that in a primal feasible basic solution \( x_e \) has one of the values \{0, 1/2, 1\} for all \( e \) in \( E \). It follows from the structure of the basis graph that \( x_e = 1/2 \) only if \( e \) is in a cycle of the basis graph and then only if \( x_e = 1/2 \) for all \( e \) in that cycle.

An initial primal feasible basic solution can be found by first constructing a tour (in our implementation we start out with a "cycle" through nodes 1 and 2 and then insert nodes 3, 4, ..., \( n \) sequentially in the cheapest way). If \( n \) is odd then the tour forms an initial basis graph and \( U \) is empty for the initial basis structure. Otherwise, if \( e = (1, i) \) and \( f = (1, j) \) are the two edges in the tour meeting node 1, then the 1-tree consisting of the edge \( e \) and the cycle formed by the edge \( g = (i, j) \) and all edges in the tour except \( e \) and \( f \), forms an initial basis graph and \( U = \{ f \} \) for the initial basis structure.

### 3 Dual values for RMP2

For any basis \( B \) the dual values \( u_i \), for all \( i \) in \( N \), are the unique solution to the following set of dual equations

\[ u_i + u_j = c_e \text{ for all } e = (i, j) \text{ in } E(B). \]

The dual values define the reduced costs

\[ r_e = c_e - u_i - u_j \text{ for all } e = (i, j) \text{ in } E. \]

A basis structure \((L, U)\) is dual feasible if \( r_e \geq 0 \) for all \( e \) in \( L \) and \( r_e \leq 0 \) for all \( e \) in \( U \) and optimal if it is both primal and dual feasible.

The dual value of any node in a cycle of the initial basis graph is easily computed. If \( 1, 2, ..., q \) is the sequence of edges in the cycle and node \( i \) is the common node of edges \( l \) and \( q \), then by traversing the cycle from node \( i \) and alternately adding and subtracting the dual equations for the edges in the cycle one obtains the following equation from which \( u_i \) can be computed:

\[ 2u_i = c_1 - c_2 + ... + c_q \]

It follows that if the edge costs are even integers then all the dual values as well as the reduced costs are integers.

Once \( u_i \) is computed the dual values for all the other nodes in the 1-tree of the initial basis graph can be computed by repeatedly selecting an edge in the 1-tree for which the dual value of only one node has been computed and computing the dual value of the other node using the dual equation for that edge.

For any basis \( B \) and edge \( e \) in \( E(B) \) the subgraph with edge set \( E(B) - \{ e \} \) contains precisely one tree (a connected subgraph without a cycle) This tree contains a node, say \( t \), incident with \( e \) and one can partition its nodes into the following two sets:

\[ P = \{ i : \text{nodes } i \text{ and } t \text{ are connected by a path of even length} \} \]

\[ Q = \{ i : \text{nodes } i \text{ and } t \text{ are connected by a path of odd length} \}. \]

The updating of the dual values after a basis change can be done efficiently using the following theorem which indicates that not all dual values need to be changed:

**Theorem** Suppose the edge \( e \) enters the basis graph and \( E(B) \) is the edge set of the new basis graph. If \( u_i / v_j \) is the dual value of node \( i \) before / after the basis change then...
Proof: For any edge \( f \) in \( E(B) - \{ e \} \) either one node is in \( P \) and the other node in \( Q \) or both nodes are in \( N - P - Q \) and it follows that \( s_f \), the reduced cost of edge \( f \) after the basis change, equals \( r_f (= 0) \). If \( e \) is in a cycle of the basis graph then, since the length of the cycle is odd, both nodes of \( e \) are in \( P \) so that \( s_e = r_e - 2\Delta = 0 \) if \( \Delta = r_e / 2 \). Otherwise one node of \( e \) is in \( P \) and the other node is in \( N - P - Q \) so that \( s_e = r_e - \Delta = 0 \) if \( \Delta = r_e \).

In order to implement the dual value changes described above, it is convenient to be able to visit the nodes in \( P \cup Q \) in increasing order of the length of the path from node \( t \). For that reason we use the augmented predecessor representation of the basis graph described by Glover, Klingman & Stutz [3] rather than the threaded list representation used by Brown & McBride [1].

4 Ratio test for RMP2

Suppose the current primal feasible basic solution is not dual feasible and an entering edge \( g = (i, k) \) has been selected. To perform the ratio test, it is necessary to determine the nonzero components of the \( n \)-vector \( y \) such that \( By = A(g) \) where \( B \) is the current basis and \( A(g) \) is the column of \( A \) corresponding to edge \( g \). \( A(g) = e_i^t + e_k^t \), i.e. the sum of two unit \( n \)-vectors.

If the two vectors \( w \) and \( z \) such that \( Bw = e_i^t \) and \( Bz = e_k^t \) are known, it follows that \( y = w + z \).

It follows from Section 5.4 in [4] that \( w_0 \) alternates between 1 and -1 for \( e \) in the path \( P_i \) from node \( i \) to the cycle \( C_i \) in the 1-tree of the basis graph containing \( i \) and between 1/2 and -1/2 for \( e \) in \( C_i \) (with all other components of \( w \) equal to 0). Similarly \( z_0 \) alternates between 1 and -1 for \( e \) in the path \( P_k \) from node \( k \) to the cycle \( C_k \) in the 1-tree of the basis graph containing \( k \) and between 1/2 and -1/2 for \( e \) in \( C_k \) (with all other components of \( z \) equal to 0).

If nodes \( i \) and \( k \) are in different 1-trees of the basis graph then \( y_e = w_e \) for \( e \) in \( P_i \cup C_i / P_k \cup C_k \). If \( i \) and \( k \) are in the same 1-tree of the basis graph then \( C_i = C_k \) and one of the following two cases occurs:

(a) \( P_i \) and \( P_k \) join at a node \( j \). Then \( y_e = w_e / z_e \) for \( e \) in \( P_i - P_j / P_k - P_j \) (where \( P_j \) is the path from \( j \) to \( C_j \)) and \( y_e = w_e + z_e \) for \( e \) in \( P_j \cup C_j \). If the lengths of the paths \( P_i - P_j \) and \( P_k - P_j \) are both odd or both even, then \( y_e \) alternates between 2 and -2 for \( e \) in \( P_j \) and between 1 and -1 for \( e \) in \( C_j \). Otherwise \( y_e = 0 \) for all \( e \) in \( P_j \cup C_j \).

(b) \( P_i \) and \( P_k \) join \( C_i \) at two different nodes \( j_i \) and \( j_k \). Then \( y_e = w_e / z_e \) for \( e \) in \( P_i / P_k \) and \( y_e = w_e + z_e \) for \( e \) in \( C_i \). Therefore \( y_e = 0 \) for \( e \) in one of the two paths in \( C_i \) between \( j_i \) and \( j_k \) while \( y_e \) alternates between 1 and -1 for \( e \) in the other path.

As in [3] we assume that each cycle in the basis graph is oriented clockwise. If node \( i \) is not in a cycle of the basis graph we let \( d_i \) = length of the path in the basis graph from node \( i \) to a cycle and \( p_i \) = next node in the path from \( i \) to a cycle. If node \( i \) is in a cycle of the basis graph we let \( d_i = 0 \) and \( p_i \) = predecessor in the cycle of node \( i \). Then the following algorithm can be used to compute the nonzero components of the vector \( y \) (the \( y \)-value for the edge \((i,p_i)\) is stored in \( y_i \)):

\[
y_i = 1;
\]
\[
\text{while}(d_i > d_k) \{ y_i = y_i; y_i = -y_i; i = p_i; \}
\]
\[
y_k = 1;
\]
\[
\text{while}(d_i < d_k) \{ y_k = y_k; y_k = -y_k; k = p_k; \}
\]
\[
\text{if}(i == k)
\]
\[
\{ /* two paths intersect */
\]
\[
\text{if}(y_i == y_k) \{ y_i = 2 * y_i; \}
\]
\[
\text{while}(d_i > 0) \{ y_i = y_i; y_i = -y_i; i = p_i; \}
\]
\[
y_i = y_i/2;
\]
\[
k = i;
\]
\[
\{ do \{ y_i = y_i; y_i = -y_i; i = p_i; \} \text{while}(i! = k); \}
\]
\[
\text{else} \{ /* now at cycle on both paths */
\]
\[
\text{if}(y_i == y_k) \{ y_i = 2 * y_i; \}
\]
\[
\text{while}(d_i > 0) \{ y_i = y_i; y_i = -y_i; i = p_i; \}
\]
\[
y_i = y_i/2;
\]
\[
k = i;
\]
\[
\{ do \{ y_i = y_i; y_i = -y_i; i = p_i; \} \text{while}(i! = k); \}
\]
\];
\]
\[
\text{else} \{ /* i and k are in different cycles */
\]
\[
\text{if}(y_i == y_k) \{ y_i = 2 * y_i; \}
\]
\[
\text{while}(d_i > 0) \{ y_i = y_i; y_i = -y_i; i = p_i; \}
\]
\[
y_i = y_i/2;
\]
\[
k = i;
\]
\[
\{ do \{ y_i = y_i; y_i = -y_i; i = p_i; \} \text{while}(i! = k); \}
\]
\];
\]
\];
\]
\];
\]
\];
\]
\];
\]
memory requirements as well as calculation time for reduced costs.

A two-phase approach was used in solving RMP2. In the first phase we consider for each node only the 10% cheapest edges meeting the node, i.e. \( [(n-1)/10] \) edges per node, as candidates to enter the basis graph. The other edges are only considered in the second phase. In all but one test problem we found that the optimal RMP2 solution was found in the first phase, i.e. no pivots were required in the second phase.

In our code we used the "node most dual infeasible edge" rule to select the next entering edge, i.e. whenever a dual infeasible edge meeting a node \( i \) is detected, the most dual infeasible edge meeting node \( i \) is selected as the entering edge. We experimented with the Gibby selection rule [2] but although it reduced the total number of pivots on average by 35 percent, the average computation time was not reduced.

The performance of the RMP2 code was experimentally compared on a 33 Mhz 80386-based PC with a 80387 math co-processor against a standard linear program solver, viz. Version 4.00/387 of the XA system [7]. The problem set used in the comparison consisted of the five 100 node problems, two 150 node problems and two 200 node problems from Krolak, Felts & Marble [5]. Our results are reported in the table below which contains the average solution times (in centiseconds) using the two codes as well as the ratios between the average solution times of the XA code and the RMP2 code.

<table>
<thead>
<tr>
<th>Problem size</th>
<th>Average RMP2-time</th>
<th>Average XA-time</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>22</td>
<td>7900</td>
<td>359</td>
</tr>
<tr>
<td>150</td>
<td>42</td>
<td>27150</td>
<td>646</td>
</tr>
<tr>
<td>200</td>
<td>82</td>
<td>63750</td>
<td>777</td>
</tr>
</tbody>
</table>

Table 1: Comparison of RMP2 and XA

The results show that the RMP2 code is much more efficient than the XA code on instances of the RMP2 problem. It also suggests that the solution time of the RMP2 code grows as \( n^2 \) while the solution time of the XA code on RMP2 problems grows as \( n^3 \), i.e. the relative efficiency of our code increases linearly with the problem size.

6 Conclusion

Our computational experience demonstrates that a special purpose implementation of the primal simplex algorithm for RMP2 is several orders of magnitude faster than a standard linear program solver. Since such a standard linear program solver was used by Padberg & Rinaldi [6] to solve RMP2 in their successful algorithm for the STSP, we are optimistic that our current investigation of the use of our RMP2 code in an algorithm for the STSP will prove beneficial.

Acknowledgements

We wish to thank William Stewart of the College of William and Mary, Williamsburg, for providing us with machine-readable copies of the Krolak test problems and Kobus Wolvaardt of the University of South Africa, Pretoria, for allowing us to use his XA system.

References

Notes for Contributors

The prime purpose of the journal is to publish original research papers in the fields of Computer Science and Information Systems, as well as shorter technical research papers. 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, if they are not clear in the text.
- 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:
- Manuscripts accepted for publication should comply with the above guidelines, and may be provided in one of the following formats:
  - in a typed form (i.e. suitable for scanning);
  - as an ASCII file on diskette; or
  - as a WordPerfect, TeX or LaTeX file; or
  - in camera-ready format.

Charges
Charges per final page will be levied on papers accepted for publication. They will be scaled to reflect scanning, typesetting, reproduction and other costs. Currently, the minimum rate is R20-00 per final page for camera-ready contributions and the maximum is R100-00 per page for contributions in typed format. 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.

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.
Contents

GUEST CONTRIBUTION

Why all the Fuss About Neural Networks?
G Barth ..................................................... 4

RESEARCH ARTICLES

The Placement of Subprograms by an Automatic Programming System
J P du Plessis and H J Messerschmidt ........................... 9

An Investigation into the Separation of the Application from its User Interface
J H Greyling and P R Warren .................................. 16

The Physical Correlates of Local Minima
L F A Wessels, E Barnard and E van Rooyen ................ 22

An Efficient Primal Simplex Implementation for the Continuous 2-Matching Problem
T H C Smith, T W S Meyer and L Leenen .................... 28

Concept Network Framework for a Multi-paradigm Knowledge Base
J Kambanis .................................................. 32

PEST - A Microcomputer Pascal Based Expert System Shell
A G Sartori-Angus and R Neville .............................. 39

A Linda Solution to the Evolving Philosophers Problem
S E Hazelhurst ................................................ 44

TECHNICAL NOTE

Knowledge Representation using Formal Grammars
S H von Solms, E M Ehlers and D J Enslin ..................... 54

COMMUNICATIONS AND REPORTS

Book Review & Books Received .................................. 57
Cost-effective Visual Simulation Based on Graphics Workstations
A Caduri and D G Kourie ........................................ 58
A Method of Controlling Quality of Application Software
T D Crossman .................................................. 70
An Update on UNINET-ZA: The Southern African Academic and Research Network
V Shaw .......................................................... 75