84-00980

UNLIMITED

DEPARTMENT OF NATIONAL DEFENCE

CANADA

OPERATIONAL RESEARCH AND ANALYSIS ESTABLISHMENT

DIRECTORATE OF LOGISTICS ANALYSIS

D LOG A STAFF NOTE 83/11

ALGORITHMS AND COMPUTER PROGRAMS FOR THE ANALYSIS OF MULTI-ATTRIBUTE, HIERARCHICAL, RANKING PROCESSES

by

F. Audet

Staff Notes are written as informal records of data, analyses, tentative views, comments, methodology, or briefing material, which for one reason or another do not warrant or require formal publication. The contents are the responsibility of the author, and do not necessarily reflect the opinion of the Directorate.

OTTAWA, ONTARIO DECEMBER 1983

National Defence / Défense nationale

ORAE 3561-96513-10 (D Log A

Operational Research and
Analysis Establishment
Ottawa, Ontario
K1A 0K2

16 January 1984

Distribution List

DISTRIBUTION OF D LOG A STAFF NOTE 83/11

References: A. D Log A Staff Note 83/11, "Algorithms and Computer Programs For the Analysis of Multi-Attribute, Hierarchical, Ranking Processes", by F. Audet, December 1983 (copy attached)

B. ORAE Project Report No. PR 232, "Experiment and Analysis to Assess a Preference Ordering Scheme - the Analytical Hierarchy Process", by R. Bijoor, November 1983

1. Reference A is distributed to supplement Reference B. Computer programs that were used in Reference B are documented in this staff note. Moreover, some computational aspects that were not treated in Reference B are considered in Reference A.

2. Any comments or suggestions should be forwarded to François Audet at 992-3735.

Keith Kavanagh
K.R. Kavanagh
Director Logistics Analysis
for Chief
Operational Research and Analysis
Establishment

Attachment

DISTRIBUTION LIST

NDHQ

DGLEM
- PM LLAD (3)
  (Attn: LCol G. Lamarre)
- PM MOST
- DLES

PMO AAP
  (Attn: Capt K. Barnard)

DAR
  (Attn: LOH Project Director)

DMAEM 6
  (Attn: Capt D. Krant)

——>DSIS

ORAE

C/ORAE) (1 copy)
DC/ORAE)
DGGA (1 plus 1 for circ)
DGOR (1 for circ)
ORAE Library (2)

ABSTRACT

The selection of the best option among many alternatives is a difficult problem, especially when based on many criteria. Recently, a hierarchy ranking process has been developed by T.L. Saaty (Reference 1), and was discussed in general terms by R. Bijoor (Reference 8).

This ranking scheme is quite tedious to use by hand; it was therefore felt that an interactive computer implementation of Saaty's ideas would be useful.

Using an illustrative example, this Staff Note shows how to use the computer programs to select the best alternative based on a set of criteria. The computational and mathematical aspects of this computer implementation are also presented in this report.

RESUME

Choisir la meilleur alternative parmi plusieurs possibilités est un problème difficile, surtout lorsque plusieurs critères sont impliqués dans cette sélection. Récemment, un processus de rangement hiérarchique a été développé par T.L. Saaty (référence 1), et fut traité de façon générale par R. Bijoor (voir référence 8).

Cette façon d'ordonner les alternatives est pénible à utiliser manuellement; nous avons donc cru que des programmes informatique utilisant les méthodes de Saaty seraient utiles.

A l'aide d'un example, ce rapport explique comment se servir des programmes afin de sélectionner, à partir d'un ensemble de critères, la meilleur alternative. Les aspects mathématiques et numériques impliqués dans cette implantation sont aussi discutés.

TABLE OF CONTENTS

ALGORITHMS AND COMPUTER PROGRAMS FOR THE ANALYSIS OF MULTI-ATTRIBUTE, HIERARCHICAL, RANKING PROCESSES

INTRODUCTION

1. The method implemented in this Staff Note is a formal way of doing what we all do intuitively when faced with several alternatives, out of which we want to select the 'best'. In such a case, we implicitly 'rank' each of the alternatives according to some scale of criteria. Unfortunately, when decisions are reached intuitively, the scale of criteria tends to fluctuate, which implies that the final selection might be biased. On the other hand, when the selection is done formally, not only is the scale of criteria fixed, but one can measure to what extent one has been consistent in making the comparisons.

2. Usually, the criteria on which the selection is based are of unequal importance; therefore they have to be 'ranked', i.e. their relative values must be found. Once this is done, the alternatives are ranked with respect to each criterion. In this way, for each alternative, its relative value with respect to each criterion is known. Since the relative importance of those criteria is also known, one can 'roll up' those values and obtain a final ranking.

3. How the method works is best conveyed through an example; this is the purpose of the next section. In Annex A some of the computational and mathematical aspects related to the computer implementation are discussed. The computer programs as well as a description of their interactions can be found in Annex B.

AN EXAMPLE

4. As an illustrative example, the method is applied to the selection of a new car. When faced with this problem, one has several criteria (sometimes conflicting) and several choices. For the sake of illustration, let's suppose that the criteria are the following:

.../2

a. Budget: cost of acquisition, cost of maintenance and cost of fuel.

b. Performance: speed, acceleration, etc.

c. Comfort: this includes accessories, seats, room.

d. Security: brake system, steering, ease of handling, visibility.

e. Body: aesthetics, ease of access, finish.

5. Moreover, suppose that the selection has to be made from the following five cars:

a. Honda Accord;

b. Chevette;

c. BMW 320i;

d. Pontiac Grand Lemans;

e. Ford Thunderbird.

6. First, the criteria are ranked. To do this, they are compared in a pairwise fashion using the following scale:

1/7: Decisively less important;

1/5: Less important;

1/3: Slightly less important;

1 : Equally important;

3 : Slightly more important;

5 : More important;

7 : Decisively more important.

with 1/6, 1/4, 1/2, 2, 4, 6 used as intermediate values.

Suppose that with the above scale, one obtains the following table for the criteria comparisons:

	Budget	Performance	Comfort	Security	Body
Budget	1	3	4	1	7
Performance	1/3	1	2	1/2	3
Comfort	1/4	1/2	1	1/2	2
Security	1	2	2	1	5
Body	1/7	1/3	1/2	1/5	1

7. Some comments are in order at this point. First, note that the diagonal of the above table consists of all 1s. This makes sense since any criterion is equally important as itself! Second, there is some symmetry in the table; for instance, in the first row, we have Budget being slightly more important (3) than performance, while looking at the performance row, we see that performance is slightly less important than budget (1/3). Tables of this kind are called Reciprocal matrices. It is clear that knowing the entries forming the upper triangular part of the table is sufficient; this is why only those entries are input to the programs. In the present case, the array 3 4 1 7 2 1/2 3 1/2 2 5 is entered and one gets the eigenvector (essentially a ranking vector) 0.3882, 0.16, 0.1056, 0.2913, 0.05492 giving the relative importance of each criterion (see Figure 1), which means that Budget in the most important criterion with a value of 0.3882, followed by Security at 0.2913, Performance at 0.16, Comfort at 0.1056 and Body at 0.05492. The consistency ratio measures how far the reciprocal matrix is from perfect consistency; see Saaty (Reference 1) for a thorough discussion of consistency. A consistency ratio less than 0.1 is considered quite acceptable.

.../4

```
      RANKING
INPUT PRIORITY ENTRIES
⎕:
      3 4 1 7 2 .5 3 .5 2 5

EIGENVECTOR: 0.3882 0.16 0.1056 0.2913 0.05492
MAX EIGENVALUE 5.076
CONSISTENCY RATIO: 0.01706


INPUT ENTRIES CORRESPONDING TO BUDGET
⎕:
      .333333 5 3 4 7 5 6 .5 .5 .5

EIGENVECTOR: 0.2563 0.511 0.0524 0.08191 0.09838
MAX EIGENVALUE 5.181
CONSISTENCY RATIO: 0.0403


INPUT ENTRIES CORRESPONDING TO PERFOR
⎕:
      3 .2 1 .5 .14285 .2 .166666 4 3 .5

EIGENVECTOR: 0.1126 0.04138 0.4938 0.1335 0.2188
MAX EIGENVALUE 5.137
CONSISTENCY RATIO: 0.03067


INPUT ENTRIES CORRESPONDING TO COMFOR
⎕:
      7 1 3 3 .14285 .333333 .333333 3 3 1

EIGENVECTOR: 0.3533 0.04565 0.3533 0.1238 0.1238
MAX EIGENVALUE 5.008
CONSISTENCY RATIO: 0.001691


INPUT ENTRIES CORRESPONDING TO SECURI
⎕:
      5 .5 2 2 .14285 .333333 .333333 4 4 1

EIGENVECTOR: 0.2426 0.04906 0.456 0.1262 0.1262
MAX EIGENVALUE 5.031
CONSISTENCY RATIO: 0.006971


INPUT ENTRIES CORRESPONDING TO BODY
⎕:
      5 .5 2 3 .14285 .25 .333333 3 2 2

EIGENVECTOR: 0.2726 0.04652 0.3902 0.168 0.1227
MAX EIGENVALUE 5.153
CONSISTENCY RATIO: 0.03409


FINAL RANKING
-------------

BMW320   0.2909
HONDA    0.2405
CHEVET   0.2266
THUNDE   0.1298
GRAND    0.1122
```

Figure 1 - Program Output Using Data From The Example

8. Next, the cars are compared in a pairwise fashion with respect to each criterion. Suppose that one obtains the following tables.

BUDGET

	Honda	Chevette	BMW	Lemans	Thunderbird
Honda	1	1/3	5	3	4
Chevette	3	1	7	5	6
BMW	1/5	1/7	1	1/2	1/2
Lemans	1/3	1/5	2	1	1/2
Thunderbird	1/4	1/6	2	2	1

PERFORMANCE

	Honda	Chevette	BMW	Lemans	Thunderbird
Honda	1	3	1/5	1	1/2
Chevette	1/3	1	1/7	1/5	1/6
BMW	5	7	1	4	3
Lemans	1	5	1/4	1	1/2
Thunderbird	2	6	1/3	2	1

.../6

COMFORT

	Honda	Chevette	BMW	Lemans	Thunderbird
Honda	1	7	1	3	3
Chevette	1/7	1	1/7	1/3	1/3
BMW	1	7	1	3	3
Lemans	1/3	3	1/3	1	1
Thunderbird	1/3	3	1/3	1	1

SECURITY

	Honda	Chevette	BMW	Lemans	Thunderbird
Honda	1	5	1/2	2	2
Chevette	1/5	1	1/7	1/3	1/3
BMW	2	7	1	4	4
Lemans	1/2	3	1/4	1	1
Thunderbird	1/2	3	1/4	1	1

BODY

	Honda	Chevette	BMW	Lemans	Thunderbird
Honda	1	5	1/2	2	3
Chevette	1/5	1	1/7	1/4	1/3
BMW	2	7	1	3	2
Lemans	1/2	4	1/3	1	2
Thunderbird	1/3	3	1/2	1/2	1

9. As before, only the entries forming the triangle above the diagonal need to be entered (see Figure 1). For the first criterion (Budget), the Chevette clearly dominates with a relative value of 0.511. On the other hand, for the second criterion (Performance), the BMW dominates with a relative value of 0.4938, while the Chevette has only 0.04138. The final ranking on Figure 1, is just the product of the matrix, whose columns are the eigenvectors for the five criteria, and the eigenvector resulting from the criteria comparisons. The BMW leads with a relative value of 0.2909 followed by the Honda with 0.2405 and the Chevette with 0.2266. Note that those results only express the author's preference and should not be taken too seriously!

10. Now, what would happen if instead of entering a value of 7 for the Budget-Body comparison in the first table, one had input a value of 1? This would have been a contradiction since the Budget would then be as important as Body which is less important than Performance which in turn is less important than Budget. Hence, it could be deducted that Budget is less important than Budget, clearly a contradiction. As shown in Figure 2, the program has the ability to detect those 'Loops' and allows the user to make changes.

CONCLUSIONS

11. Through a simple example, it was shown how easy it is to use the program for doing ranking. The program is flexible - no dimensioning needs to be done, only part of the tables need to be input; it is also 'intelligent' since it can find loops in the input data and prompts the user for modification. This program should be useful in any ranking process, and can easily be extended to cases of more levels of criteria.

```
    RANKING
INPUT PRIORITY ENTRIES
⎕:
    3 4 1 1 2 .5 3 .5 2 5
YOU ARE INTRANSITIVE...
BUDGET=BODY  <COMFOR<PERFOR<BUDGET => BUDGET<BUDGET

DO YOU WANT TO MAKE CHANGES TO PRIORITY MATRIX ?(YES/NO)
YES
INPUT PRIORITY ENTRIES
⎕:
    3 4 1 7 2 .5 3 .5 2 5

EIGENVECTOR: 0.3882 0.16 0.1056 0.2912 0.05492
MAX EIGENVALUE 5.076
CONSISTENCY RATIO: 0.01706
```

Figure 2 - Intransitivity Detection

Annex A
to D Log A Staff Note 83/11
dated December 1983

NOTATION

Capital letters are used for matrices, Greek letters for scalars, x, x^1, $x_1 \ldots x_N$ denote column vectors.

A_{NxN} = an NxN matrix

I_{NxN} = NxN identity matrix

x^T = Transpose of x i.e. if $x = [\xi_1, \ldots, \xi_K]$ then $x^T = \begin{bmatrix} \xi_1 \\ \cdot \\ \cdot \\ \xi_K \end{bmatrix}$

$\|x\|_2$ = 2 norm of x i.e. $\left(\sum_{i=1}^{n} \xi_i^2 \right)^{1/2}$

$\|x\|_1$ = 1 norm of x i.e. $\sum_{i=1}^{n} |\xi_i|$

$\triangleq$ = By definition

$\prod \psi_j$ = Product of ψ_j

λ_{max} = Maximum eigenvalue (in magnitude) of a given matrix.

MATHEMATICAL FORMULATION

1. In this annex we explain why we compute the maximum eigenvector of the reciprocal matrices. We also introduce a numerical method to find the maximum eigenvector. We then proceed to show how some of the inconsistencies in the reciprocal matrices can be displayed and, at the discretion of the user, removed.

2. Suppose that we have N items and that we know the value W_i of each item i (i = 1,...,N). Then when doing pairwise comparisons between any items i and j we find that item i is W_i/W_j 'worth' item j. If we do this for all possible pairs, we get the following matrix:

$$\begin{array}{c} \\ \text{item 1} \\ \text{item 2} \\ \vdots \\ \text{item N} \end{array} \begin{array}{c} \begin{array}{cccc} \text{item 1} & \text{item 2} & \cdots & \text{item N} \end{array} \\ \left[\begin{array}{cccc} 1 & W_1/W_2 & \cdots & W_1/W_N \\ W_2/W_1 & 1 & & W_2/W_N \\ \vdots & & \ddots & \\ W_N/W_1 & W_N/W_2 & \cdots & 1 \end{array}\right] \end{array} \triangleq A \qquad (1)$$

if we let $\mathbf{x}^T = [W_1, W_2, \ldots, W_N]$, then it is clear that $A\mathbf{x} = N\mathbf{x}$ i.e. $\mathbf{x}$ is an eigenvector of A with corresponding eigenvalue N.

Usually though, we do not know the exact ratios W_i/W_j, but as shown in ref. 1, if we are sufficiently consistent when doing the pairwise comparisons, the eigenvector corresponding to the maximum eigenvalue of the resulting matrix is a good measure of the relative value of each item. We now show that if A is as in (1), then N is the maximum eigenvalue, and all the other eigenvalues are zero.

3. Proposition: $\det(A_{N\times N} - \lambda I_{N\times N}) = (-1)^N \lambda^{N-1}(\lambda - N)$ (2)

Proof: First note that for any $K \geq 1$ we have

$$\det(A_{K\times K}-\lambda I_{K\times K}) = \begin{vmatrix} 1-\lambda & W_1/W_2 & \cdots & W_1/W_k \\ W_2/W_1 & 1-\lambda & \cdots & W_2/W_k \\ \vdots & & \ddots & \\ W_k/W_1 & W_k/W_2 & \cdots & 1-\lambda \end{vmatrix}$$

$$= \frac{1}{W_1 W_2 \ldots W_k} \begin{vmatrix} (1-\lambda)W_1 & W_1 & \cdots & W_1 \\ W_2 & (1-\lambda)W_2 & & W_2 \\ \vdots & & & \\ W_k & W_k & & (1-\lambda)W_k \end{vmatrix} = \begin{vmatrix} (1-\lambda) & 1 & \cdots & 1 \\ 1 & (1-\lambda) & \cdots & 1 \\ \vdots & & & \\ 1 & 1 & \cdots & (1-\lambda) \end{vmatrix}$$

$$= \begin{vmatrix} 1-\lambda & \lambda & \cdots & \lambda \\ 1 & -\lambda & 0 & 0 \\ \vdots & & & \\ 1 & 0 & \cdots & -\lambda \end{vmatrix} \qquad (3)$$

Now we are in a position to prove (2). We proceed by induction on N. Clearly for N = 1 (1) holds trivially. Assume the result is true for $K = N-1 \geq 1$, we want to show it is true for N.

By (3) we have $\det(A_{N\times N}-\lambda I_{N\times N}) = \begin{vmatrix} 1-\lambda & \lambda & \cdots & \lambda \\ 1 & -\lambda & \cdots & 0 \\ & & \ddots & \\ 1 & 0 & & -\lambda \end{vmatrix}_{N\times N}$

expanding using the last row gives

$$\det(A_{N\times N}-\lambda I_{N\times N}) = -\lambda \begin{vmatrix} 1-\lambda & \lambda & \cdots & \lambda \\ 1 & -\lambda & & 0 \\ 1 & 0 & \cdots & -\lambda \end{vmatrix}_{n-1\times n-1} + (-1)^{N-1} \begin{vmatrix} \lambda & & \cdots & \lambda \\ -\lambda & 0 & \cdots & 0 \\ & & -\lambda & 0 \end{vmatrix}_{n-1\times n-1} \qquad (4)$$

by (3) and induction hypothesis the first determinant is nothing but

$$(-1)^{N-1}\lambda^{N-2}(\lambda-(N-1)) \qquad (5)$$

while the second determinant equals

$$\lambda^{N-1}\begin{vmatrix} 1 & \cdots & 1 \\ -1 & 0\cdots & 0 \\ & -1 & 0 \end{vmatrix} = \lambda^{N-1}\begin{vmatrix} 0 & \cdots & 0 & 1 \\ -1 & \ddots & & \\ & & \vdots & \\ & & -1 & 0 \end{vmatrix} = \lambda^{N-1}(-1)^{N}\cdot(-1)^{N-2} \qquad (6)$$

$$= \lambda^{N-1}.$$

Finally, substituting (5) and (6) in (4) yields the result. □

4. From the foregoing equations, it can be seen that the magnitude of the W_i's has no bearing on the result just proved. Saaty calls the matrix A consistent if each paired comparison is transitive and linear in a multiplicative sense i.e. if a is preferred v fold to b and b u fold to c then a is to be preferred $u\cdot v$ fold to c. Note that if the recommended scale is less than $u\cdot v$, a certain amount of inconsistency is to be expected. In Reference 1, it is shown that if a positive matrix $B_{N\times N}$ is close to being a consistent reciprocal matrix, then $\lambda_{max} \gtrsim N$, and all the other eigenvalues are close to zero. It is further shown that the eigenvector corresponding to λ_{max} is close to the vector x whose i^{th} entry is

$$\alpha\left(\prod_{j=1}^{n} B_{ij}\right)^{1/n}$$ where α is an appropriate normalizing factor.

5. An algorithm to determine λ_{max} and its corresponding eigenvector is now presented. Let $\lambda_1, \ldots, \lambda_n$ be the eigenvalues of A, and $x_1, \ldots, x_n$ be its corresponding eigenvectors. Then, it can be shown (see Stewart Reference 2) that $(\lambda I - A)^{-1}$ has eigenvalues $(\lambda - \lambda_1)^{-1}, \ldots, (\lambda - \lambda_n)^{-1}$ with corresponding eigenvectors $x_1, \ldots, x_n$. Now let x be an arbitrary vector, then if $x_1, \ldots, x_n$ are linearly independent, we can write

$$x = \sum_{i=1}^{N} a_i x_i \qquad \text{for some } a_i$$

Hence we have

$$x^1 \triangleq (\lambda I-A)^{-1}x = \frac{\alpha_1 x_1}{\lambda-\lambda_1}+\frac{\alpha_2 x_2}{\lambda-\lambda_2}+\dots+\frac{\alpha_n x_n}{\lambda-\lambda_n} \quad (7)$$

Now, if λ is much closer to λ_1 than to $\lambda_2, \dots, \lambda_n$, then $(\lambda-\lambda_1)^{-1}$ will be much larger than $(\lambda-\lambda_2)^{-1}, \dots, (\lambda-\lambda_n)^{-1}$. Therefore if α_1 is not unreasonably small, the term $\alpha_1(\lambda-\lambda_1)^{-1}$ will dominate all the others and x^1 will tend to lie in the direction of x_1. Using x^1 instead of x one can further improve the approximation. One possible improvement to the above scheme is to determine λ at each iteration, such that $\|Ax - \lambda x\|_2$ is minimized. Such a λ is called the Rayleigh quotient. It is easy to see that it has the form

$$\lambda = \frac{x^T Ax}{x^T x} \quad (8)$$

6. In order to determine x_1 using iteration of the form in (7), the following conditions must be satisfied:

a. λ_1 must be well separated from $\lambda_2, \dots, \lambda_n$ and,

b. either a good approximation to λ_1, or a good approximation to x_1 must be available.

For a reciprocal matrix A close to being consistent, $\lambda_{max} \triangleq \lambda_1$ is close to N while $\lambda_2, \dots, \lambda_n$ are close to 0. Therefore, conditions (a) and (b) are simultaneously satisfied. Moreover, we also have a good approximation x to x_1 for starting the iterations i.e. vector x whose i^{th} component is

$$\left(\prod_{j=1}^{n} A_{ij}\right)^{1/n}$$

7. We now present the complete algorithm for finding the maximum eigenvalue and its corresponding eigenvector using what is called the Rayleigh quotient iteration which is a variant of the inverse power method. Given $A_{n\times n}$ a positive reciprocal matrix and tolerance ε, the following algorithm computes λ_{max} and its corresponding eigenvector within a tolerance of ε.

1. Compute vector $\tilde{x}$, which has i^{th} entry equal to $\left(\prod_{j=1}^{n} A_{ij}\right)^{1/n}$
2. Let $x \leftarrow \frac{\tilde{x}}{\|\tilde{x}\|_1}$
3. $\lambda \leftarrow \frac{x^T A x}{x^T x}$
4. If $\|Ax - \lambda x\|_2 < \varepsilon$ then output x and λ; STOP
5. Else do: Solve the system $(\lambda I - A)\tilde{x} = x$
   Go to step 2.

Some words of caution: This algorithm should only be used when conditions a. and b. previously mentioned are satisfied and that the matrix A is small, say at most 15 by 15. If conditions a. and b. hold but the matrix is large then one should first reduce the matrix A to an upper Hessenberg prior to using the above algorithm. This would save computations in step 5. See Reference 2 for more information on Hessenberg form. Finally, if conditions a. and b. do not hold, one should reduce A to upper Hessenberg form and then use the QR algorithm. Reference 3 is the definitive treatise on eigenvalues and eigenvectors; it discusses the QR algorithm. The EISPACK package has the most complete, efficient and reliable subroutines for solving the eigenvalue problem for various types of matrices. (See Reference 4 for more details). Note, for all the matrices of the example run, only one iteration was necessary to find λ_{max} and corresponding eigenvector correct to at least 2 decimal places.

8. As a safeguard against intransitivity, which is the most extreme form of inconsistency, we introduce some graph theory and an algorithm to find cycles (see Reference 5 for more details). From the hierarchy matrix, one can get a graph representation in which there is a directed edge from node u to node v, if u is less or equally important than v. Therefore, any intransitivity in the hierarchy matrix results in a cycle in its graph representation. Before presenting the algorithm, some definitions are needed. A node y is said to be a descendant of a node x if y can be reached from x by following directed edges. A node is said to have been "visited" if the algorithm has done the required processing for that node. The last definition above makes sense provided there is

no cycle in the graph; otherwise, as soon as from a node y, an unvisited vertex x is reached, which was already traversed, a cycle has been found. By keeping track of the edges traversed in the paths from x to y, and from y back to x, the cycle can be displayed. This is in essence what the algorithm FindCYCLE does.

Description of Variables Used in FindCYCLE:

N: Number of nodes in the graph.

ADJACENCY: N x (N+1) matrix. Adjacency (i, j)= $\begin{cases} 1 & \text{if (i.j) is a directed edge in the graph} \\ 0 & \text{otherwise} \end{cases}$

COUNT: Vector of size N. COUNT(i) = j if node i was the j^{th} node to be traversed.

FATHER: Vector of size N used to keep track of the path which leads to the node currently being traversed.

VISITED: Vector of size N. VISITED(i) = 1 if all the descendants of node i have been visited. Otherwise VISITED(i) = 0.

Algorithm FindCYCLE

Step 1. [INITIALIZE] COUNT(i) ← N+1 (i = 1, ..., N)
FATHER(i) ← 0 (i = 1, ..., N)
VISITED(i) ← 0 (i = 1, ..., N)
NVISIT ← 0

Step 2. [VISIT] If all nodes have been marked as "visited" then STOP. Else pick up any unvisited node W. SEARCH(W)

Step 3. [LOOP] Repeat Step 2.

Procedure SEARCH(W)

[1] COUNT(W) ← NVISIT + 1
NVISIT ← NVISIT + 1

[2] Pick any unvisited vertex Z adjacent to W

[3] If there is no such vertex then mark W VISITED(W) ← 1

[4] Else if COUNT(Z) < COUNT(W) then a cycle has been found.

[5] Else FATHER(Z) ← W
SEARCH(Z)

[6] If VISITED(W) = 0 then go to [2]
Else RETURN.

Annex B
to D Log A Staff Note 83/11
dated December 1983

PROGRAM DESCRIPTIONS

1. The package implementing the hierarchy ranking process consists of several programs which interact as depicted on Figure B1. They are written in the programming language APL.([6] and [7] are complete references on APL). A brief description of the programs is now presented, followed by program listings.

BUILD: Program used to build a table of titles describing each priority and each item to be compared. See Figure B2.

CONSISTENCY: It creates an adjacency matrix corresponding to the directed graph G associated with matrix A. Consistency repeatedly calls SEARCH, until all nodes of G are visited.

CONSTRUCT: Given a vector $(x_1, \ldots, x_k)$, CONSTRUCT creates (if possible) a N x N reciprocal matrix of the form:

$$\begin{bmatrix} 1 & x_1 & x_2 & \cdots & x_{i_1} \\ 1/x_1 & 1 & x_{i_1+1} & x_{i_1+2} \cdots & x_{i_2} \\ 1/x_2 & 1/x_{i_1+1} & 1 & x_{i_2+1} \cdots & x_{i_3} \\ \vdots & & & & \vdots \\ & & & & x_K \\ 1/x_i & \cdots & 1/x_{K-1} & 1/x_K & 1 \end{bmatrix}$$

This is possible if and only if there exists an M such that $\sum_{i=1}^{M} i = K$;

i.e. $K = \frac{M(M+1)}{2}$ for some integer M.

CHECKCYCLE: It determines whether the cycle found by SEARCH is trivial (i.e. consists only of equalities) or non-trivial. In the latter case, CHECKCYCLE prints out the resulting intransitivity.

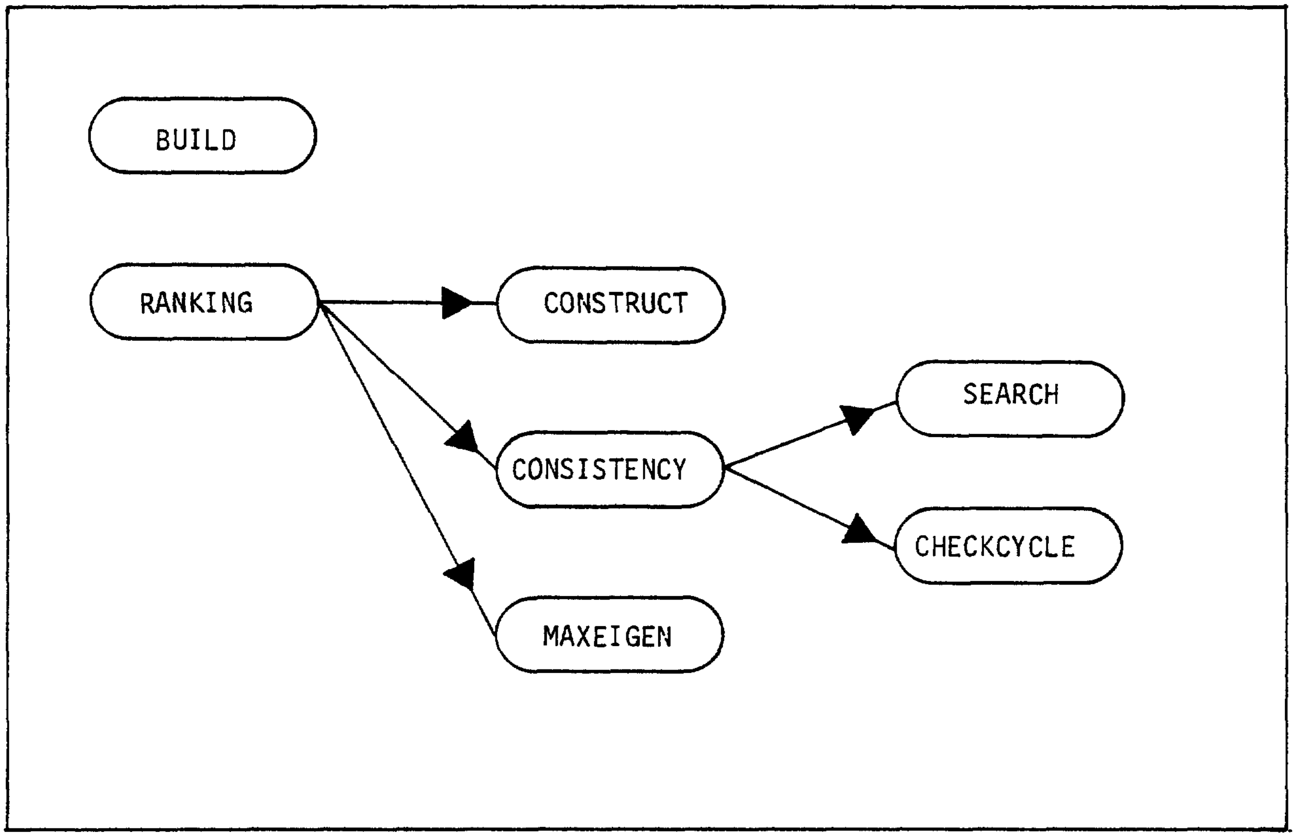

Figure B1 - Program Interactions

```
BUILD
NUMBER OF PRIORITIES ?
□:
      5
ENTER NAME OF PRIORITY 1
BUDGET
ENTER NAME OF PRIORITY 2
PERFORMANCES
ENTER NAME OF PRIORITY 3
COMFORT
ENTER NAME OF PRIORITY 4
SECURITY
ENTER NAME OF PRIORITY 5
BODY
NUMBER OF ITEMS TO BE COMPARED ?
□:
      5
ENTER NAME OF ITEM 1
HONDA
ENTER NAME OF ITEM 2
CHEVETTE
ENTER NAME OF ITEM 3
BMW320I
ENTER NAME OF ITEM 4
GRAND LE MANS
ENTER NAME OF ITEM 5
THUNDERBIRD
```

Figure B2 - Construction of a Table of Titles Using BUILD

MAXEIGEN: Computes the maximum eigenvalue and its associated eigenvector, as well as the consistency ratio.

SEARCH: This recursive program is essentially a depth-first search of G (see Reference 5). SEARCH calls CHECKCYCLE as soon as it finds a cycle.

RANKING: MAIN program calls CONSTRUCT, CONSISTENCY and MAXEIGEN. It has some capabilities to check for incorrect input. RANKING creates a matrix called RANKMAT whose columns are the eigenvectors of the ranking matrices. A final ranking is produced by multiplying RANKMAT by the eigenvector corresponding to the priority matrix.

PROGRAM LISTINGS

```
      ∇ Z←MAXEIGEN A;I
   X←|X÷+/X←(×/[2] A)*÷FIRST←1↑ρA
   I←(⍳FIRST)∘.=⍳FIRST
  LE:→((+/((A+.×X)-(L←(X+.×A+.×X)÷X+.×X)×X)*2)<0.0001)ρEND
   X←X÷+/X←|X⌹(L×I)-A
   →LE
  END:'EIGENVECTOR: ',⍕Z←X
   'MAX EIGENVALUE ',⍕L
   'CONSISTENCY RATIO: ',⍕(CI←(L-N)÷N-1)÷INDEX[N]
      ∇

      ∇ RANKING;N;M;I;TEMP;TP
  AGAIN:'INPUT PRIORITY ENTRIES'
   CONSTRUCT ⎕
   →(INPUTOK=1)ρSUITE1
   'INCORRECT NUMBER OF ENTRIES'
   →AGAIN
  SUITE1:N←1↑ρA
   TYPE←1
   CONSISTENCY
   →(CONSISTENT=0)ρSKIP
   'DO YOU WANT TO MAKE CHANGES TO PRIORITY MATRIX ?(YES/NO)'
   →('Y'=1↑⍞)ρAGAIN
  SKIP:PRIORVECTOR←MAXEIGEN A
   I←1
  MORE: 3 45 ρ(90ρ' '),'INPUT ENTRIES CORRESPONDING TO ',ITEM[1;I;]
   CONSTRUCT ⎕
   →(INPUTOK=1)ρSUITE2
   'INCORRECT NUMBER OF ENTRIES'
   →MORE
  SUITE2:M←1↑ρA
   TYPE←2
   CONSISTENCY
   →(CONSISTENT=0)ρBOUCLE
   'DO YOU WANT TO CHANGE ENTRIES ? (YES/NO)'
   →('Y'=1↑⍞)ρMORE
  BOUCLE:→(I≠1)ρNEXT
   RANKMAT←(M,1)ρMAXEIGEN A
   →LOOP
  NEXT:RANKMAT←RANKMAT,[2](M,1)ρMAXEIGEN A
  LOOP:→((I←I+1)≤N)ρMORE
   5 13 ρ(26ρ' '),'FINAL RANKING',13ρ'-'
   TEMP←RANKMAT+.×PRIORVECTOR
   ITEM[2;TP;],[2]((M,1)ρ' '),[2]⍕(M,1)ρTEMP[TP←⍒TEMP]
      ∇

      ∇ SEARCH W;Z;POS
   COUNT[W]←NVISIT←NVISIT+1
   POS←0
  NEXT:POS←POS+1
   Z←ADJACENCY[W;POS]
   →(Z=0)ρFIN
   →((VISITED[Z]=1)∨W=Z)ρNEXT
   →(COUNT[Z]≠COUNT[W])ρSUITE
   W CHECKCYCLE Z
   →NEXT
  SUITE:FATHER[Z]←W
   SEARCH Z
   →NEXT
  FIN:VISITED[W]←1
      ∇
```

```
∇ BUILD;N;I;M
'NUMBER OF PRIORITIES ?'
ITEM←ρN←⎕
I←1
MORE1:'ENTER NAME OF PRIORITY ',⍕I
ITEM←ITEM,6↑⍞,6ρ' '
→((I←I+1)≤N)ρMORE1
'NUMBER OF ITEMS TO BE COMPARED ?'
I←1
→(N≥M←⎕)ρMORE2
ITEM←ITEM,(6×M−N)ρ' '
MORE2:'ENTER NAME OF ITEM ',⍕I
ITEM←ITEM,6↑⍞,6ρ' '
→((I←I+1)≤M)ρMORE2
ITEM←(2,(M⌈N),6)ρITEM
∇
```

```
∇ Z CHECKCYCLE W;N
⍝FIND IF CYCLE IS CAUSED BY INCONSISTENCY
STACK←⍳0
CYCLE←A[W;Z]≠1
LOOP:STACK←STACK,Z
CYCLE←CYCLE,A[FATHER[Z];Z]≠1
Z←FATHER[Z]
→(Z≠W)ρLOOP
STACK←STACK,W
→((+/CYCLE)=0)ρEND
2 23 ρ'YOU ARE INTRANSITIVE...',23ρ' '
CONSISTENT←1
OUTPUT←ITEM[TYPE;STACK[1];]
N←ρCYCLE
I←1
AGAIN:OUTPUT←OUTPUT,(1↑(CYCLE[I],1)/'<='),ITEM[TYPE;STACK[I+1];]
→((I←I+1)≤N)ρAGAIN
⎕←OUTPUT←OUTPUT,' => ',ITEM[TYPE;W;],'<',ITEM[TYPE;W;]
END:' '
∇
```

```
∇ CONSISTENCY;N;I
N←1↑ρA
I←1
ADJACENCY←(N,N+1)ρ0
AGAIN:TEMP←(A[I;]≥1)/⍳N
ADJACENCY[I;⍳ρTEMP]←TEMP
→((I←I+1)≤N)ρAGAIN
CONSISTENT←NVISIT←0
COUNT←NρN+1
VISITED←FATHER←Nρ0
BOUCLE:→((+/VISITED)=N)ρEXIT
SEARCH VISITED⍳0
→BOUCLE
EXIT:
∇
```

```
∇ CONSTRUCT X;I;N;⎕CT
⎕CT←1E¯10
N←1+(¯1+(1+8×ρX)*0.5)÷2
→(INPUTCK←(+/N=⍳10)=1)×OK
OK:A←(⍳N)∘.=⍳N←⌊0.5+N
POS←¯1−I←1
NEXT:A[I;I+⍳N−I]←TP←X[(⍳N−I)−POS]
POS←POS+N−I
A[I+⍳N−I;I]←÷TP
→((I←I+1)<N)ρNEXT
∇
```

REFERENCES

1. T.L. Saaty, The Analytic Hierarchy Process, McGraw-Hill, New York, 1980.

2. G.W. Stewart, Introduction to Matrix Computations, Academic Press, New York, 1973.

3. J.H. Wilkinson, The Algebraic Eigenvalue Problem, Oxford University Press (Clarendon), London and New York, 1965.

4. B.T. Smith et al., Matrix Eigensystem Routines - EISPACK Guide, 2nd ed., (Lecture Notes in Computer Science) Vol. 6, Springer-Verlag, New York, 1976.

5. Alfred V. Aho, John E. Hopcroft and Jeffrey D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, Massachusetts, 1974.

6. Raymond P. Polivka and Sandra Pakin, APL: The Language and its Usage, Prentice-Hall, Englewood Cliffs, New Jersey, 1975.

7. Leonard Gilman and Allen J. Rose, APL An Interactive Approach, 2nd ed., John Wiley & Sons, New York, 1976.

8. R. Bijoor, Experiment and Analysis to Assess a Preference Ordering Scheme - The Analytical Hierarchy Process, ORAE Project Report No. 232, November 1983.

#91789