SYSTEMS RESEARCH INSTITUTE
POLISH ACADEMY OF SCIENCES

INTERNATIONAL INSTITUTE FOR APPLIED SYSTEMS ANALYSIS

CONTRACTED STUDY AGREEMENT REG / POL / 1

"CONCEPTS AND TOOLS FOR STRATEGIC REGIONAL SOCIO-ECONOMIC CHANGE POLICY"

STUDY REPORT

PART 3

APPENDIX: SOFTWARE AVAILABLE

COORDINATOR, IIASA: A. KOCHETKOV
COORDINATOR, SRI PAS: A. STRASZAK

ZTS / ZPZC / ZTSW 1-36/85

WARSAW 1986

SYSTEMS RESEARCH INSTITUTE

POLISH ACADEMY OF SCIENCES

AND

INTERNATIONAL INSTITUTE FOR APPLIED SYSTEMS ANALYSIS

CONTRACTED STUDY AGREEMENT REG/POL/1

"CONCEPTS AND TOOLS FOR STRATEGIC REGIONAL

SOCIO-ECONOMIC CHANGE POLICY"

STUDY REPORT

Consisting of 3 Parts

PART 3

APPENDIX: SOFTWARE AVAILABLE

COORDINATOR, IIASA : A. KOCHETKOV

COORDINATOR, SRI PAS : A. STRASZAK

ZTS/ZPZC/ZTSW 1-36/85

W A R S A W 1986

I. AGGREGATION OF ORDERINGS - MEASUREMENT OF CONSENSUS DEGREE

by

Lidia Księżopolska, Jan Owsiński and Sławomir Zadrożny

I.1 Introduction

This chapter contains description of software applications related to aggregation of orderings. These applications are based upon three various approaches, referring to, respectively, linear programming formulation, branch-and-bound method, and special heuristics. All of them start from the objective function formulated in its general form by e.g. Kemeny (see e.g. Kemeny (1959), and Kemeny and Snell (1963)).

The problem being solved by these software applications is of high significance for any comprehensive prospective--strategic-analysis, as outlined in Chapter I, Part 1 of the Report. It can be used to model, and therefore bring solutions to, a variety of situations, ranging from collective definition of problem areas to consider, through simple voting, down to aggregation of importance sequences.

It is therefore significant to have these problems properly modelled and adequately solved. In view of the way in which they are going to be applied, the modelling and solution methodes should in addition be simple and operational, most desirably microcomputer implementable. Thus, not only the way in which questions are stated would be proper and answers obtained adequate, but also these questions and responses could be exchanged in an interactive way during an analytic session.

I.2 Problem formulation

Assume that one is dealing with n distinct alternative items (problems, candidates, policies, technologies,...), denoted $a_1,\ldots,a_n$, $a_i \in A = \{a_1,\ldots,a_n\}$, and that a number, m, of judges (experts, voters, analysts,...) are expressing an opinion as to the sequence (importance, quality, appearance in time,...)

of these items. Thus, each judge formulates an ordering R^k, $k=1,\ldots,m$ of items a_i. The problem is to find the aggregate ordering, $\hat{R}$, which would represent in the "best", most appropriate way all the m opinions expressed through R^k.

Having a measure of such "goodness" (i.e. a distance of opinions) and the "optimum", $\hat{R}$, or at least suboptimum, R^*, aggregate opinion, makes it possible to assess the degree of judges' consensus, ranging from 0, when the maximum distance occurs, to 1, when all experts give identical R^k. Further analyses become also feasible, facilitating the course of an interactive analytic session.

The three methods, whose software is introduced in this Chapter, all do refer to the formalism said to have originated with Condorcet (1785), but better known from such references as Kemeny (1959) and Kemeny and Snell (1963). It is also interesting to note that similar formulations as regards the objective function representing the "goodness" of aggregation vis à vis individual R^k come from the cluster analysis domain, see e.g. Owsiński (1986).

The differences in approaches lie first in the more precise formulation of the objective function and in the methods of extremization.

Now, the respective software shall be presented, starting from the one based upon the geometrical considerations, main reference being that by Kuźmin and Ovchinnikov (1975,1976), through the simple but powerful LP formulation due to Marcotorchino and Michaud (1979), and finally to an efficient heuristic procedure, by Owsiński and Zadrożny (1986), based upon the objective function equivalent to that proposed by Marcotorchino and Michaud.

I.3 Method no. 1: the "EXPERT" program

The purpose of the program called "EXPERT" is to determine the group opinion on the basis of the experts' individual opinions given in the form of orderings.

The program has been written in FORTRAN for microcomputer COMPAN-8.

Input data to the program

Input data to the program "EXPERT" are entered in two ways: either from the console keyboard or from a disc file.

At the beginning of the program the following information from the console keyboard are to be entered:

- number of alternative variants (items) (from 1 to 15)
- number of experts (judges) (from 1 to 20)
- the initial options i.e.:
  the entry of expert's opinions
  1 - from the keyboard
  2 - from a disc file
  reports from the program will go out
  1 - to the display
  2 - to the printer
- number of group opinions if the solution

for a given set of orderings is not unique (from 1 to 1000).

The expert's opinions are entered from the keyboard in the form of orderings. Each ordering given by an expert is entered in one line with commas separating individual values. If the data given by an expert are not all entered in on line, the program will request the missing values for the ordering.

When all the orderings are entered, the program processes each of them into preference matrix (described in the next section). The preference matrices are stored on a disc.

Such a disc file is used as an input file to the program if the initial option 2 (entry from a disc file) has been chosen.

Description of the method applied in the program

According to the geometric approach all the experts' individual orderings should be presented in the form of binary preference relations R^k (k=1,...,m).

Elements of the matrix R^k, given by the expert k, are:

$$r_{ij}^{k} = \begin{cases} 1 & \text{if the variant } a_i \text{ is preferred to the variant } a_j \text{ in the opinion the k-th expert} \\ 0 & \text{otherwise} \end{cases}$$

The group opinion is determined in the following way:

On the basis of the set of individual orderings $R^M = \{R^1,\ldots,R^m\}$ a new set Z of all possible orderings of elements of the set A is constructed; Z is a convex set.

Group opinions are included in the so-called core of that convex set Z. The core contains all the orderings "lying" between two "extreme" orderings; one of them shows the highest consensus of the experts, the second - the lowest one.

The application of the geometric approach can result in a solution different from any of the orderings given by experts. The group opinion should satisfy the following conditions:

- the "between" condition which implies that an ordering of variants accepted as the group-proper one "lies" among all the orderings given by experts.

This condition can be expressed as follows:

$$\bigcap_{k=1}^{m} R^k \subseteq R^* \subseteq \bigcap_{k=1}^{m} R^k$$

where:

R^* is the group opinion presented in the form of the matrix preference relation.

- the Pareto condition; it implies that if all the experts prefer variant a_i to variant a_j, then in the group opinion the variant a_i is preferred to a_j too.
- the transitivity condition;

This condition can be expressed as follows:

if $a_i R^* a_j$ and $a_j R^* a_l$ then $a_i R^* a_l$

in which we read " $a_i R^* a_j$" as "according to the preference relation R^* the element a_i is preferred to the element a_j".

An algorithm used to determine the group opinion based on the geometric approach was completed with the majority rule given by Condorcet, i.e. if most of the experts prefer the variant a_i to a_j then in the group opinion a_i is preferred to a_j too.
If the core of the convex set Z determined on the basis of individual orderings R^k (k=1,...,m) includes more than one group opinion, then the solution for a given set A is not unique.

In such cases the best group opinion should satisfy the following condition:

$$\sum_{k=1}^{m} d\,(R^*, R^k) = \min_{R \in Z} \sum_{k=1}^{m} d\,(R, R^k)$$

where:

$d(R,R^k)$ is a distance between the ordering R, belonging to the set Z, and the ordering R^k given by the k-th expert.

The distance between two orderings can be expressed as follows, Kemeny and Snell (1963):

$$d(R,R^k) = \frac{1}{2} \sum_{i,j}^{m} |r_{ij} - r_{ij}^{k}|$$

Out of all the opinions belonging to the core of the convex set Z, the group opinion satisfying the condition of the least distance is the Kemeny median of the set R (of individual orderings).

Output data from the program

The reports from the program will go out either to the screen (option 1) or to the printer (option2) in the following sequence:

- the experts' orderings in the form: number of the variant/the ordered sequence of variant,
- the group opinions generated by the program,
- the distance between the group opinion obtained first and all the experts' individual opinions,
- the distances between all the orderings given by experts.

The program "EXPERT" occupies 24kB. The 10 kB area on a disc to store the preference matrices is to be reserved.

I.4 Method no. 2 : LP formulation

In this method the software data are formulated in a similar way, as in method no. 1 except that the number of equivalent solutions is not given.

The question is treated via the method of Marcotorchino and Michaud (1979), i.e. as an LP problem

$$\max_{x_{ij}} \sum_{i,j} (d_{ij}x_{ij} + s_{ij}(1-x_{ij})) \qquad (I.1)$$

subject to

$$x_{ij} + x_{ji} = 1 \qquad \forall\, i,j \qquad (I.2)$$

$$x_{ij} + x_{jk} - x_{ik} \leq 1 \quad \forall\, i,j,k \qquad (I.3)$$

being a continuous representation of a binary problem, in which $x_{ij} = 1$ and $x_{ji} = 0$ when issue (item) i precedes issue (item) j in the aggregate opinion, while $x_{ij} = 0$ and $x_{ji} = 1$ in the opposite case. Magnitudes of d_{ij} and s_{ij} indicate, respectively, the number of judges who placed i before j and of those who placed j before i, i.e.

$$d_{ij} = \sum_{k=1}^{m} x_{ij}^k \;, \quad s_{ij} = \sum_{k=1}^{m} x_{ji}^k \qquad (I.4)$$

where m is the number of judges and $x_{ij}^k = x_{ji}^k = 0$ in case of a tie, e.g. when a judge k does not refer neither to i nor to j in his ordering R^k.

This LP problem was slightly reformulated, in order to get rid of (I.2) and to accomodate a weighting coefficient r, into

$$\max_{x_{ij}} \sum_{i<j} (rd_{ij}\, x_{ij} + (1-r)d_{ji}(1-x_{ij})) \qquad (I.5)$$

$$\max_{x_{ij}} \sum_{i<j} (r(d_{ij} + d_{ji})\, x_{ij} - d_{ji}x_{ij})$$

$$x_{ij} \leq 1 \quad \forall i < j \qquad (I.6)$$

and (I.3) as before. The new problem (I.5), (I.6), (I.3) yields

the same solution as (I.1, I.2, I.3) for r=0.5. Now note that (I.5, I.6, I.3) can be easily parametrized with regard to r, $r \in [0,1]$. Assume starting at r = 1. Since (I.5) takes then the form of

$$\max_{x_{ij}} \sum_{i<j} d_{ij} x_{ij} \qquad (I.7)$$

maximum for r=1 is reached for $x_{ij} = 1, \forall i < j$. At the other end of parametrization, i.e. at r=0, there is

$$\max_{x_{ij}} \sum_{i<j} - d_{ji} x_{ij} \qquad (I.8)$$

which means an optimum at $x_{ij} = 0, \forall i < j$. Thus, on the way from r=1 to 0 an initially chosen "lexicographical" order is reversed, and at r=0.5 the proper optimum or optima are found. At the discrete values of r order is changed. The values of r indicates the relative aggregate preference of the shifted entities. When r falls below 0.5 the counter-preference shifts are made. Thus, not only the optimal aggregate opinion could be determined, but also its stability and structure.
Note that for r=0.5 one obtains the actual optima $\hat{R}$.

I.5 Method no. 3 : effective heuristics

The third method used is derived from the approach developed by one of these authors, see e.g. Owsiński (1984). This method starts from the objective function equivalent to the one as in (I.5), although it does not refer explicity to variables x_{ij}.

The procedure is, roughly, as follows:

1° t, step number, = 1, $r^t = 1$, $O^t = \{1,2,3,\ldots,n\}$, where n: total number of items ordered.

2° t = t + 1.

3° calculate, for all k,l, k < l:

$$r^t_{kl} = \frac{\sum_{i \in O^{t-1}_{l-}(k,l-1)} d_{li} - \sum_{i \in O^{t-1}_{l+}(k,l-1)} d_{li}}{\sum_{i \in O^{t-1}_{l-}(k,l-1)} (d_{li} + d_{il}) - \sum_{i \in O^{t-1}_{l+}(k,l-1)} (d_{li} + d_{il})} \qquad (I.9)$$

and

$$r_{lk}^{t} = \frac{\sum_{i \in O_{k+}^{t-1}(k+1,l)} d_{ik} - \sum_{i \in O_{k-}^{t-1}(k+1,l)} d_{ik}}{\sum_{i \in O_{k+}^{t-1}(k+1,l)} (d_{ik} + d_{ki}) - \sum_{i \in O_{k-}^{t-1}(k+1,l)} (d_{ik} + d_{ki})} \qquad (I.10)$$

where, e.g.

$$O_{k+}^{t-1}(k+1,l) = \{i \mid k+1 >_{O^{t-1}} i >_{O^{t-1}} l, i > k\} \qquad (I.11)$$

and $>_{O^{t-1}}$ denotes weak preference according to O^{t-1}, while r_{kl}^{t} and r_{lk}^{t} correspond, respectively, to the following operations:

r_{kl}^{t} : ... k i_1 i_2 ... ... l ... → ...l k i_1 i_2 ...

r_{lk}^{t} : ... k ... i_2 i_1 l ... → ...i_2 i_1 l k ...

4° find $\max_{k,l} \{ r_{kl}^{t}, r_{lk}^{t} \} = r^{t}$, and thereby k*l*

5° perform operation corresponding to r_{k*l*}^{t} and thereby form O^{t},

6° if O^{t} is a reverse of O^{1} or $r^{t} = 0$, go to 7°, otherwise return to 2°,

7° stop.

The procedure outlined obviously suboptimizes (I.5), in that it makes order changes belonging to a predefined narrow class, e.g. the ones indicated above (one item shifted ahead of the other or one item dropped behind another). What it can, however, loose on optimality, it compensates in simplicity and numerical efficiency. One can see that parametrization in (I.5) was in fact a step in the direction of this procedure. The series of r^{t} can again be used for assessing the strength of aggregate preferences.

This method, of course, requires the same data as the method no.2. Both of them were programmed in FORTRAN for a minicomputer SM-4, PDP-compatible, but the method no.3 can, of course, be implemented and run on any sort of personal computer.

The Appendix, which follows the references contains the listings of methods considered.

I.6 References

Condorcet, J.A., marquis de (1785): Essai sur l'application de l'analyse a la probalité des decisions rendues a la pluralité des voix. Ac. Roy. des Sciences, Paris.

Kuźmin V.B. and S.V. Ovchinnikov (1975,1976): Geometry of the space of preference relations (in Russian). Automatika i Telemekhanika no.12/1975, no. 1/1976.

Kemeny, J.G.(1959): Mathematics without numbers. Daedalus, vol. 88, pp. 571-591.

Kemeny J.G. and J.L. Snell (1963): Mathematical models in the social sciences. New York.

Marcotorchino, F. and P. Michaud (1979): Optimisation en analyse ordinale des donnies. Masson, Paris.

Owsiński, J.W. (1984): On a quasi-objective global clustering method. In: Data Analysis and Informaties, III. E.Diday et al., eds. North Holland, Amsterdam.

Owsiński, J.W. (1986): Optimisation in clustering: an approacl and other approaches. Control and Cybernetics. To appear.

Owsiński, J.W. and S. Zadrożny (1986): Structuring a regional problem. Aggregation and clustering in orderings. Applic Stochastic Models and Data Analysis. To appear.

APPENDIX A.

Computer printout of software for method no. 1

```
      program expert
      integer oe(15,15,20),a(15,15),b(15,15),c(15,15),d(225),
     *e(225),f(225)
      open(1,file='lpt1')
      open(6,file='macierz.dat')
      write(*,2006)
1     write(*,1506)
      read(*,'(i1)')ii
      if(ii.eq.1)then
           ii=0
      else
      if(ii.eq.2)then
          ii=6
      else
          go to 1
      endif
      endif
2     write(*,1507)
      read(*,'(i1)')io
      if(io.eq.1)then
           io=0
      else
      if(io.eq.2)then
          io=1
      else
          go to 2
      endif
      endif
      if(ii.eq.6)go to 4
      write(*,1502)
      read(*,2000)i1
      write(*,1504)
      read(*,2000)k1
4     write(*,1503)
      read(*,'(i4)')ird
      do 3 i=1,15
      do 3 j=1,15
3     a(i,j)=0
      write(io,1500)
      if(ii.eq.0)go to 5
      read(6,'(2i2)')i1,k1
      read(6,'(i1)')(((oe(i,j,k),i=1,i1),j=1,i1),k=1,k1)
      write(io,1508)i1,k1
      go to 11
5     continue
      do 6 k=1,k1
      do 7 i=1,i1
      do 7 j=1,i1
7     oe(i,j,k)=1
      read(*,2003)(a(1,i),i=1,i1)
      do 8 i=1,i1-1
      do 8 j=i+1,i1
      if(a(1,i).eq.a(1,j))go to 8
      if(a(1,i).gt.a(1,j))go to 9
      oe(i,j,k)=0
```

```
      go to 8
9     oe(j,i,k)=0

8     continue
6     continue
      write(6,'(2i2)')i1,k1
      do 10 k=1,k1
      do 10 j=1,i1
      do 10 i=1,i1
10    write(6,'(i1)')oe(i,j,k)
11    continue
      ie=0
      mn=100
      do 12 k=1,k1
      do 13 i=1,i1
      do 13 j=1,i1
13    c(i,j)=1-oe(i,j,k)
      call druk(a,c,i1,io,ie)
12    continue
      ie=0
      k2=k1/2
      if(k2*2.eq.k1)go to 14
      write(io,2005)((a(k,i),i=1,15),k=2,4,2)
      write(io,2005)((a(k,i),i=1,15),k=1,4,2)
14    continue
      do 16 i=1,i1
      do 16 j=1,i1
      a(i,j)=0
16    c(i,j)=1
      do 17 k=1,k1
      do 17 j=1,i1
      do 17 i=1,i1
      oe(i,j,k)=1-oe(i,j,k)
      if(oe(i,j,k).eq.0)go to 17
      a(i,j)=1
17    continue
      do 18 i=1,i1
      do 18 j=1,i1
      l=0
      k2=k1/2
      if(2*k2.eq.k1)go to 19
      k2=k2+1
19    continue
      do 20 k=1,k1
20    l=l+oe(i,j,k)
      if(l.ge.k2)go to 21
      b(i,j)=0
      go to 18
21    b(i,j)=1
18    continue
      do 22 k=1,k1
      do 23 j=1,i1
      do 23 i=1,i1
      if((a(i,j).eq.1).and.(oe(i,j,k).eq.0))go to 24
      go to 23
24    oe(i,j,k)=1
```

```
      do 25 l=1,i1
      do 25 m=1,i1
      do 25 n=1,i1
      if((oe(l,n,k)*oe(n,m,k).eq.1).and.(oe(l,m,k).eq.0))go to 2
25    continue
      go to 23
26    oe(i,j,k)=0

23    continue
22    continue
      write(io,'(//)')
      do 27 k=1,k1
      do 27 i=1,i1
      do 27 j=1,i1
      if(oe(i,j,k).eq.0)go to 28
      go to 27
28    c(i,j)=0
27    continue
      do 29 i=1,i1
      do 29 j=1,i1
      if(b(i,j).eq.1)go to 36
      go to 29
36    c(i,j)=1
29    continue
30    continue
      do 51 i=1,i1
      do 51 j=1,i1
      do 51 l=1,i1
      if((c(i,l)*c(l,j).eq.1).and.(c(i,j).eq.0))go to 52
51    continue
      go to 40
52    continue
      do 53 i=1,i1
      do 53 j=1,i1
      if(b(i,j).eq.1)go to 54
53    continue
      go to 40
54    c(i,j)=0
      b(i,j)=0
      go to 30
40    continue
      l=0
      m=0
      do 56 i=1,i1
      do 56 j=1,i1
      if((a(i,j)-c(i,j)).eq.1)go to 55
      go to 56
55    m=m+1
      l=i*mn+j
      d(m)=l
56    continue
103   continue
      write(io,1505)
      do 96 i=1,i1
      do 96 j=1,i1
96    b(i,j)=c(i,j)
```

```
      do 117 i=1,i1
      do 117 j=1,i1
117   a(i,j)=0
      call druk(a,c,i1,io,ie)
      if(m.eq.0)go to 155
      if(ird.eq.1)go to 155
      do 57 i=1,m
57    e(i)=d(i)
      k2=-1
68    k2=k2+1
      if(k2.eq.m)go to 77
      do 58 i=1,m

      f(i)=e(i)
58    d(i)=e(i)
      m1=m
      k3=k2
      m2=m
67    continue
      do 59 i=1,m1-k2
      do 60 k=i,i+k2
      ia=d(k)/mn
      ib=d(k)-ia*mn
      if(k2.eq.0)go to 61
      if(i.gt.1)go to 60
      if(k.gt.i)go to 62
      do 63 l1=1,i1
      do 63 l2=1,i1
63    c(l1,l2)=b(l1,l2)
62    c(ia,ib)=1
60    continue
61    c(ia,ib)=1
      do 64 i2=1,i1
      do 64 i3=1,i1
      do 64 i4=1,i1
      if((c(i2,i4)*c(i4,i3).eq.1).and.(c(i2,i3).eq.0))go to 65
64    continue
      id=id+1
      if(id.le.ird)call druk(a,c,i1,io,ie)
      if(id.eq.ird)go to 77
65    c(ia,ib)=0
59    continue
      if(k2.eq.0)go to 68
73    continue
      m1=m1-1
      if(m1.eq.k2)go to 70
      do 66 i=k3,m1
66    d(i)=d(i+1)
      go to 67
70    continue
      k3=k3-1
      if(k3.eq.0)go to 68
      if(k3.eq.1)go to 71
      do 72 i=1,m2
72    d(i)=f(i)
      m1=m2
```

```
      go to 73
71    continue
      m2=m2-1
      if(m2.eq.k2)go to 74
      do 75 i=1,m2
      f(i)=f(i+1)
75    d(i)=f(i)
      k3=k2
      m1=m2
      go to 67
74    continue
      go to 68
77    continue
      write(io,2005)((a(k,i),i=1,15),k=2,4,2)
      write(io,2005)((a(k,i),i=1,15),k=1,4,2)
      go to 101
155   write(io,2010)(a(2,i),i=1,i1)

      write(io,2010)(a(1,i),i=1,i1)
101   continue
      rewind 6
      read(6,'(2i2)')i1,k1
      do 208 k=1,k1
      do 208 j=1,i1
      do 208 i=1,i1
208   read(6,'(i1)')oe(i,j,k)
      write(io,'(//)')
      write(io,1513)
      do 150 i=1,i1
      a(1,i)=0
      do 150 j=1,i1
150   a(1,i)=a(1,i)+(1-b(i,j))
      do 136 i=1,i1
      do 136 j=1,i1
136   c(i,j)=1
      do 134 i=1,i1-1
      do 134 j=i+1,i1
      if(a(1,i).eq.a(1,j))go to 134
      if(a(1,i).gt.a(1,j))go to 135
      c(i,j)=0
      go to 134
135   c(j,i)=0
134   continue
      do 118 k=1,k1
      do 119 i=1,i1
      do 119 j=1,i1
119   a(i,j)=oe(i,j,k)
      call odleg(a,c,i1,n)
      d(k)=n
118   continue
      l=0
      do 137 k=1,k1
137   l=l+d(k)
      write(io,1515)(k,k=1,k1)
      write(io,1514)(d(k),k=1,k1),l
      write(io,'(//)')
```

```
      write(io,1516)
      write(io,1515)(k,k=1,k1)
      do 124 k=1,k1
      do 126 i=1,i1
      do 126 j=1,i1
126   c(i,j)=oe(i,j,k)
      do 125 l=1,k1
      do 127 i=1,i1
      do 127 j=1,i1
127   a(i,j)=oe(i,j,l)
      call odleg(a,c,i1,n)
      d(l)=n
125   continue
      l1=0
      do 138 j1=1,k1
138   l1=l1+d(j1)
      write(io,1517)k,(d(i),i=1,k1),l1
124   continue
1500  format(1x,'oceny ekspertow (uporzadkowanie/1.punktow)'/)
1501  format(1x,4x,i2,2x,15i2,5x,15i2)
1502  format(1x,'liczba obiektow (1-15)  ')
1503  format(1x,'liczba ocen grupowych  (1-1000)  ')

1504  format(1x,'liczba ekspertow  (1-20)  ')
1505  format(1x,'ocena grupowa')
1506  format(1x,'urz.wejscia : 1.klawiatura, 2.zbior dyskowy')
1507  format(1x,'urz.wyjscia : 1.monitor , 2.drukarka')
1508  format(1x,'l.obiektow=',i2,3x,'l.ekspertow=',i2)
1513  format(1x,'nr.eksperta/odl.od oceny grupowej')
1514  format(1x,4x,20(i3,2x),3x,i4)
1515  format(1x,4x,20(i3,2x),5x,'s')
1516  format(1x,'odl.miedzy uporzadkowaniami ekspertow')
1517  format(1x,i2,2x,20(i3,2x),3x,i4)
2000  format(i2)
2002  format(i1)
2003  format(15(i2,1x))
2005  format(1x,2(15i3,2x))
2006  format(1x,'okreslanie oceny grupowej na podst.ocen ekspertow'
2010  format(1x,15(i3,2x))
2012  format(1x,'koniec przebiegu')
      write(*,2012)
      end
c
c     obliczanie odleglosci m-dzy uporzadkowaniami
c
      subroutine odleg(a,c,i1,n)
      integer a(15,15),c(15,15)
      n=0
      do 10 i=1,i1
      do 10 j=1,i1
      if(a(i,j).eq.c(i,j))go to 10
      n=n+1
10    continue
      return
      end
c
```

```
c       drukowanie oceny grupowej
c
        subroutine druk(a,c,i1,io,ie)
        integer a(15,15),c(15,15)
        ie=ie+1
        n=ie+1
        do 89 i=1,i1
         a(ie,i)=0
        do 88 j=1,i1
88      a(ie,i)=a(ie,i)+(1-c(i,j))
        a(n,i)=i
89      continue
        do 90 i=1,i1-1
        do 91 j=i+1,i1
        if(a(ie,i).lt.a(ie,j))go to 91
        l=a(ie,i)
        m=a(n,i)
        a(ie,i)=a(ie,j)
        a(ie,j)=l
        a(n,i)=a(n,j)
        a(n,j)=m
91      continue
90      continue
        if(n.eq.4)go to 92
        ie=ie+1
        go to 93
92      continue

        write(io,2)((a(k,i),i=1,15),k=2,4,2)
        write(io,2)((a(k,i),i=1,15),k=1,4,2)
        write(io,'(1x/)')
        ie=0
        do 94 i=1,15
        do 94 j=1,15
94      a(i,j)=0
93      continue
2       format(1x,2(15i3,2x))
        return
        end
```

APPENDIX B.

Computer printout of software for method no. 2

```
      IMPLICIT INTEGER*2(A-Z)
C     ================================================================
C     PROGRAM ---  O R D E R
C
C     FOR GIVEN "M" ORDERINGS OF "N" ITEMS GENERATES LINEAR
C     PROGRAMMING PROBLEM TO AGGREGATE THESE ORDERINGS
C     ACCORDING TO MARLOTORCHINO-MICHAUD MODEL
C     ================================================================
C
C     INPUT  ---   DATA SET "ORD.DAT" CONTAINING GIVEN ORDERINGS
C
C
C     OUTPUT ---   DATA SET "PREMAT.DAT" CONTAINING PRECEDENCE
C                  MATRIX FOR GIVEN ORDERINGS
C
C            ---   DATA SET "MODEL.ORD" CONTAINING GENERATED
C                  LINEAR PROGRAMMING PROBLEM
C
C            ---   DATA SET "ORD.LOG" FOR TRACING PROGRAM EXECUTION
C
C     ================================================================
C
C     BASIC DATA STRUCTURES
C
C
C          PREMAT  --- PRECEDENCE MATRIX COMPUTED IN PROGRAM
C          ISN     --- NUMBER OF ORDERED ITEMS
C          JM      --- NUMBER OF GIVEN ORDERINGS
C
C     ================================================================
C     ================================================================
C
      LOGICAL*1 ABSENT(10)
      REAL*8 NAMES(2)
      INTEGER*2 DIGIT(10)
      REAL PREMAT(2,10,10),P
      DIMENSION ORDER(10)
      DATA DIGIT/'01','02','03','04','05','06','07','08','09','10'/
      DATA NAMES/'PREM','PRANGES'/
      ISN=10
      JM=10
      OPEN
C
      CALL ASSIGN(3,'ORD.DAT')
      CALL ASSIGN(4,'MODEL.ORD')
      CALL ASSIGN(7,'ORD.LOG')
C
      DO 1 K=1,2
      DO 1 I=1,10
      DO 1 J=1,10
1     PREMAT(K,I,J)=0.0
5     ONE=ONE+1
      WRITE(6,120) ONE,ISN
120   FORMAT(' GIVE ORDER NUMBER ',I2,' :  ',I2,'(I2,1X))')
C
C  READING OF ORDERINGS FROM INPUT
C
6     READ(3,100,END=47) ORDER
100   FORMAT(10(I2,1X))
      DO 7 I=1,ISN
      IF(ORDER(I).GE.1.AND.ORDER(1).LE.ISN) GOTO 7
      WRITE(6,101)
101   FORMAT(' ERROR,TRY AGAIN *********')
      GOTO 6
7     CONTINUE
      DO 10 I=1,ISN
```

```
10      ABSENT(I)=.TRUE.        -   20   -
C
C  COMPUTATION OF "PRMAT" ARRAY
C
        DO 15 L=1,ISN-1
C
C  ITEM NUMBER 0 ENCOUNTERED IN ORDERING MEANS
C  NO MORE ITEMS ARE GIVEN IN THIS ORDERING
C
        IF(IORDER(L).EQ.0) GOTO 35
        L1=IORDER(L)
        ABSENT(L1)=.FALSE.
C
C  FOR ALL "I" FOLLOWING "L1" SET PRMAT(I,L1,I)=PRMAT(1,L1,I)+1
C
        DO 20 M=L+1,ISN
        IF(IORDER(M).EQ.0) GOTO 25
        M1=IORDER(M)
        ABSENT(M1)=.FALSE.
        PRMAT(1,L1,M1)=PRMAT(1,L1,M1)+1.0
20      CONTINUE
        GOTO 15
C
C  ITEMS ABSENT IN GIVEN ORDERING TREATED AS FOLLOWING L1
C
25      DO 30 I=1,ISN
30      IF(ABSENT(I)) PRMAT(I,L1,I)=PRMAT(1,L1,I)+1.0
15      CONTINUE
        GOTO 5
C
C  FOR EVERY PAIR ABSENT ITEMS IN GIVEN ORDERING "I" & "J"
C  SET  PRMAT(2,I,J)=PRMAT(2,I,J)+0.5    &
C  PRMAT(2,J,I)=PRMAT(2,J,I)+0.5
C
35      DO 45 I=1,ISN
        IF(.NOT.ABSENT(I)) GOTO 45
        DO 47 J=1,ISN
40      IF((I.NE.J).AND.ABSENT(J)) PRMAT(2,I,J)=PRMAT(2,I,J)+0.5
45      CONTINUE
        GOTO 5
47      CONTINUE
C
C  PRMAT(2,*,*) TAKES TIES INTO ACCOUNT
C
        DO 50 I=1,ISN
        DO 50 J=1,ISN
50      PRMAT(2,I,J)=PRMAT(1,I,J)+PRMAT(2,I,J)
        DO 55 I=5,7
55      WRITE(7,130) (((PRMAT(J,K,L),L=1,ISN),K=1,ISN),J=1,2)
130     FORMAT(' PRECEDENCE MATRIX: '//10(10(1X,F4.1)/)///
        1' PRECEDENCE MATRIX WITH TIES: '//
        11*(10(1X,F4.1)/))
C
C  LINEAR PROGRAMMING PROBLEM GENERATION
C
        DO 57 K=1,2
        DO 57 I=1,ISN-1
        DO 57 J=I+1,ISN
57      PRMAT(K,J,I)=-PRMAT(K,J,I)
        CALL CLOSE(6)
        CALL CLOSE(7)
        CALL ASSIGN(3,'MODELP.ORD')
        END
        DO 95 K=1,2
        ENDFN=1
        D=0.0
        WRITE(F,400)
400     FORMAT('ORD',T9,'MULTST'/'ROWS'/' N  OBJ'/
        1' E  ORDJ'/' E  OBJP'/' N  OBJP')
```

```
      DO 60 I=1,ISN-2
      DO 60 J=I+1,ISN-1
      DO 60 K=J+1,ISN
60    WRITE(FN,410) DIGIT(I),DIGIT(J),DIGIT(K)
410   FORMAT(' L  C',3A2)
      WRITE(FN,420)
420   FORMAT('COLUMNS')
      DO 70 I=1,ISN-1
      DO 70 J=I+1,ISN
      D=D+PRMAT(M,J,I)
      WRITE(FN,425) DIGIT(I),DIGIT(J),PRMAT(M,I,J)
425   FORMAT(T5,'X',2A2,T15,'OBJ1',T25,F5.1)
      WRITE(FN,427) DIGIT(I),DIGIT(J),PRMAT(M,J,I)
427   FORMAT(T5,'X',2A2,T15,'OBJ2',T25,F5.1)
      IF(I.EQ.1) GOTO 62
      DO 61 K=1,I-1
61    WRITE(FN,430) DIGIT(I),DIGIT(J),DIGIT(K),DIGIT(I),DIGIT(J)
62    IF(I+1.EQ.J) GOTO 64
      DO 63 K=I+1,J-1
63    WRITE(FN,435) DIGIT(I),DIGIT(J),DIGIT(I),DIGIT(K),DIGIT(J)
435   FORMAT(T5,'X',2A2,T15,'C',3A2,T25,'-1.0')
64    IF(J.EQ.ISN) GOTO 70
      DO 65 K=J+1,ISN
65    WRITE(FN,430) DIGIT(I),DIGIT(J),DIGIT(I),DIGIT(J),DIGIT(K)
70    CONTINUE
430   FORMAT(T5,'X',2A2,T15,'C',3A2,T25,'1.0')
      WRITE(FN,436)
436   FORMAT(T5,'Y1',T15,'OBJ',T25,'1.0',T40,'OBJ1',T50,
     1'-1.0'/T5,'Y1',T15,'OBJP',T25,'-1.0'/
     1T5,'Y2',T15,'OBJ',T25,'1.0',T40,'OBJ2',T50,'-1.0'/
     1T5,'Y2',T15,'OBJP',T25,'1.0')
      DO 80 L=1,2
      WRITE(FN,440) NAMES(L)
      IF(L.EQ.2) GOTO 73
      WRITE(FN,445) D
445   FORMAT(T5,'R01',T15,'OBJ2',T25,F6.1)
73    CONTINUE
440   FORMAT(A5)
      DO 80 I=1,ISN-2
      DO 80 J=I+1,ISN-1
      DO 80 K=J+1,ISN
80    WRITE(FN,450) DIGIT(L),DIGIT(I),DIGIT(J),DIGIT(K)
450   FORMAT(T5,'R',A2,T15,'C',3A2,T25,'1.0')
      WRITE(FN,460)
460   FORMAT('BOUNDS')
      DO 90 I=1,ISN-1
      DO 90 J=I+1,ISN
90    WRITE(FN,470) DIGIT(I),DIGIT(J)
470   FORMAT(' UP B1',T15,'X',2A2,T25,'1.0')
      WRITE(FN,480)
480   FORMAT('ENDATA')
95    CONTINUE
      STOP
      END
```

APPENDIX C.

Computer printout of software for method no. 3

```
C  ====================                    ==============================
C     PROGRAM    N O R D
C
C  HAVING "N" ORDERINGS OF "M" ISSUES IN THE FORM OF
C  PRECEDENCE MATRIX THE PROGRAM GENERATES AGGREGATE
C  ORDERING ACCORDING TO THE OWSINSKI-ZADROZNY ALGORITHM
C
C  ==========================================================
C
C
C     INPUT     ---  DATA SET "PREMAT.DAT" CONTAINING PRECEDENCE
C                    MATRIX (SEE COMMENTS IN PROGRAM "ORD")
C
C
C     OUTPUT    ---  SERIES OF AGGREGATE ORDERINGS FOR COMPUTED
C                    VALUES OF "R" COEFFICIENT FROM INTERVAL [0,1]
C                    (SEE DESCRIPTION OF THE ALGORITHM)
C
C  ==========================================================
C
C     BASIC DATA STRUCTURES
C
C
C       PREMAT   ---  PRECEDENCE MATRIX
C       QMAT     ---  AUXILIARY QUOTIENTS MATRIX
C       ORDSEQ   ---  AGGREGATE ORDERING FOR CURRENT VALUE
C                     OF "R". START WITH "NATURAL" ORDERING 1,2,...,M
C  ==========================================================
C  ==========================================================
C
      REAL PREMAT(10,10),QMAT(10,10)
      INTEGER ORDSEQ(10)
      LOGICAL UP
      DATA ORDSEQ/1,2,3,4,5,6,7,8,9,10/
      DATA QMAT/100*-1.0/
C
C
      ISN=10
      CALL ASSIGN(7,'NORD.LOG')
      CALL ASSIGN(1,'PREMAT.DAT')
      READ(1,200) ((PREMAT(I,J),J=1,ISN),I=1,ISN)
200   FORMAT(10(1X,F4.1))
      WRITE(7,211) ((PREMAT(I,J),J=1,ISN),I=1,ISN)
211   FORMAT(10(10(1X,F4.1)/))
      CALL CLOSE(1)
      CALL NORD(ISN,PREMAT,ORDSEQ,QMAT)
      STOP
      END
      SUBROUTINE NORD(ISN,PREMAT,ORDSEQ,QMAT)
      REAL PREMAT(ISN,ISN),QMAT(ISN,ISN)
      INTEGER ORDSEQ(ISN)
      R=1.0
5     F=0.0
      F1=0.0
C
C  COMPUTATION OF CURRENT VALUE OF OBJECTIVE FUNCTION
C
      DO 7 I=1,ISN-1
      DO 7 J=I+1,ISN
      M1=ORDSEQ(I)
      M2=ORDSEQ(J)
      IF(M1.GT.M2) GOTO 6
      F=F+PREMAT(M1,M2)
      F1=F1+R*PREMAT(M1,M2)
      GOTO 7
6     F=F+PREMAT(M1,M2)
```

```
      F1=F1+(1-R)*PREMAT(M1,M2)
7     CONTINUE
      WRITE(6,100) R,F,F1,ORDSEQ
      WRITE(7,100) R,F,F1,ORDSEQ
100   FORMAT(1X,' R= ',F8.3,' ORIG. FUNCT. = ',F8.3
     1,' PAR. FUNCT. = ',F8.3/
     ' ORDER IS : '//19(1X,I2)/)
C
C  COMPUTATION OF CURRENT AUXILIARY QUOTIENTS
C
      CALL COMPQM(ISN,PREMAT,ORDSEQ,QMAT)
      K1=0
      K2=0
      R=0.0
      LEN=0
C
C  SEARCH FOR CANDIDATES TO SEQUENCE CHANGE IN CURRENT ORDERING
C
      DO 10 I=1,ISN-1
      DO 10 J=I+1,ISN
      M1=ORDSEQ(I)
      M2=ORDSEQ(J)
      IF(M1.GT.M2) GOTO 10
      IF((QMAT(M2,M1).LT.R).AND.(QMAT(M1,M2).LT.R)) GOTO 10
      RR=QMAT(M2,M1)
      IF(QMAT(M2,M1).LT.QMAT(M1,M2)) RR=QMAT(M1,M2)
      IF((RR.EQ.R).AND.((J-I).LT.LEN)) GOTO 10
      LEN=J-I
      K1=I
      K2=J
      R=QMAT(M2,M1)
      UP=.FALSE.
      IF(QMAT(M2,M1).GT.QMAT(M1,M2)) GOTO 10
      UP=.TRUE.
      R=QMAT(M1,M2)
10    CONTINUE
      IF(K1.EQ.0) GOTO 30
      IF(R.LT.0.0.OR.R.GT.1.0) GOTO 30
      IF(UP) GOTO 25
C
C  SHIFTING OF SELECTED ITEM
C
      M1=ORDSEQ(K2)
      DO 20 I=1,K2-K1
20    ORDSEQ(K2+1-I)=ORDSEQ(K2-I)
      ORDSEQ(K1)=M1
      GOTO 5
25    M1=ORDSEQ(K1)
      DO 27 I=1,
27    ORDSEQ(K1+I-1)=ORDSEQ(K1+I)
      ORDSEQ(K2)=M1
      GOTO 5
30    WRITE(6,120)
120   FORMAT(' END OF NORD')
      RETURN
      END
      SUBROUTINE COMPQM(ISN,PREMAT,ORDSEQ,QMAT)
C
C
C  SUBROUTINE COMPUTING CURRENT VALUES OF AUXILIARY QUOTIENTS
C  ( "QMAT" ARRAY) FOR (CURRENT) ORDERING ( "ORDSEQ" ARRAY) AND
C  PRECEDENCE MATRIX ( "PREMAT" ARRAY)
C
C
      REAL PREMAT(ISN,ISN),QMAT(ISN,ISN)
      INTEGER ORDSEQ(ISN),COUNT
      DATA COUNT/0/
      DO 10 I=2,ISN
      DO 10 J=1,I-1
```

```
      M1=ORDSEQ(J)
      M2=ORDSEQ(I)
      IF(M1.GT.M2) GOTO 10
      D1=0.0
      D2=0.0
      QMAT(M2,M1)=0.0
      DO 5 L=J,I-1
      M3=ORDSEQ(L)
      IF(M3.GT.M2) GOTO 3
      D1=D1+PREMAT(M2,M3)
      D2=D2+PREMAT(M3,M2)
      GOTO 5
3     D1=D1-PREMAT(M3,M2)
      D2=D2-PREMAT(M2,M3)
5     CONTINUE
      IF((D1+D2).EQ.0.0) GOTO 10
      IF(D1.LT.0.0) GOTO 10
      QMAT(M2,M1)=D1/(D1+D2)
10    CONTINUE
      DO 20 I=1,ISN-1
      DO 20 J=I+1,ISN
      M1=ORDSEQ(I)
      M2=ORDSEQ(J)
      IF(M1.GT.M2) GOTO 20
      D1=0
      D2=0
      QMAT(M1,M2)=0.0
      DO 15 L=I+1,J
      M3=ORDSEQ(L)
      IF(M3.LT.M1) GOTO 14
      D1=D1+PREMAT(M3,M1)
      D2=D2+PREMAT(M1,M3)
      GOTO 15
14    D1=D1-PREMAT(M1,M3)
      D2=D2-PREMAT(M3,M1)
15    CONTINUE
      IF((D1+D2).EQ.0.0) GOTO 20
      IF(D1.LT.0.0) GOTO 20
      QMAT(M1,M2)=D1/(D1+D2)
20    CONTINUE
      COUNT=COUNT+1
      IF(COUNT.LT.5) RETURN
      WRITE(7,100) ((QMAT(I,J),J=1,ISN),I=1,ISN)
100   FORMAT(' QMAT: ',10(T10,10(1X,F6.3)/))
      COUNT=0
      RETURN
      END
```

STUDY REPORT

PART 1: BACKGROUND METHODOLOGIES

AUTHORS: A. STRASZAK
J.W. OWSIŃSKI
A. JAKUBOWSKI
J. KACPRZYK
K. CICHOCKI
M. LEWANDOWSKA
W. WOJCIECHOWSKI
J. STEFAŃSKI
A. ZIÓŁKOWSKI

PART 2: POLISH CASE STUDY REPORT

AUTHORS: J.W. OWSIŃSKI
W. CIECHANOWICZ
J. BABAROWSKI
A. STRASZAK
A. JAKUBOWSKI

PART 3: APPENDIX: SOFTWARE AVAILABLE

AUTHORS: L. KSIĘŻOPOLSKA
S. ZADROŻNY
J.W. OWSIŃSKI
T. ROMANOWICZ
A. ZIÓŁKOWSKI
W. CICHOCKI
C. IWAŃSKI
A. KAŁUSZKO
P. HOLNICKI

ABC — UZP