ZASTOSOWANIA MATEMATYKI
APPLICATIONES MATHEMATICAE
XI, 4(1970)

ALGORITHM 9

A. JANICKI (Wrocław)

SOLUTION OF THE TIME TRANSPORTATION PROBLEM

1. Procedure declaration.

procedure *TTProblem*$(m, n, T, A, B, V, W, tX, d)$;
value m, n;
integer m, n, tX, d;
integer array T, A, B, V, W;
comment Procedure *TTProblem* finds an optimal solution of the $m\times n$ time transportation problem, i.e. finds an $m\times n$ matrix $X = (x_{ij})$ which minimizes the function

$$t(X) = \max_{(i,j)\in\Theta(X)} t_{ij}, \quad \Theta(X) = \{(i,j): x_{ij} > 0,\ 1\leqslant i\leqslant m,\ 1\leqslant j\leqslant n\}$$

defined in the set of $m\times n$ matrices satisfying for all $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$ the following conditions

$$x_{ij}\geqslant 0,$$
$$\sum_{j=1}^{n} x_{ij} = a_i, \quad \sum_{i=1}^{m} x_{ij} = b_j,$$

where
$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j, \quad a_i > 0, \ b_j > 0.$$

Data:
m, n — dimensions of the time transportation problem, (m number of origins, n number of destinations),
$A[1\colon m]$ — vector $(a_1, a_2, \ldots, a_m)$, (amounts of goods available at the origins),
$B[1\colon n]$ — vector $(b_1, b_2, \ldots, b_n)$, (amounts of goods wanted at the destinations),
$T[1\colon m, 1\colon n]$ — matrix $T = (t_{ij})$, (transportation times).
Results:
The final result, an $m\times n$ matrix $X = (x_{ij})$, contains at most $m+n-1$ elements $x_{ij}\neq 0$. Procedure *TTProblem* uses this fact and finds only $m+n-1$ elements x_{ij} of the matrix X.
$A, V[1\colon m]$ — if $A[i]\colon= j$ then $V[i]\colon= x_{ij}$, $i = 1, \ldots, d-1, d+1, \ldots, m$,

d – number which shows that $A[d]$, $V[d]$ must be left out,
B, $W[1{:}\,n]$ – if $B[j] := i$ then $W[j] := x_{ij}$, $j = 1, 2, \ldots, n$,
tX – the value $t(X)$.

Example:
5×6 time transportation problem

1	2	3	4	5	6	8
2	3	4	5	6	7	5
3	9	6	8	9	9	5
4	9	5	9	9	9	7
5	6	7	7	6	5	6
3	6	4	7	3	8	

where the numbers a_i, b_j are on the right and below the matrix $T = (t_{ij})$ respectively,

has as an optimal solution 5×6 matrix $X = (x_{ij})$

$$\begin{matrix} 0 & \mathbf{3} & 0 & 0 & \boldsymbol{3} & \boldsymbol{2} \\ 0 & \boldsymbol{3} & 0 & \mathbf{2} & 0 & 0 \\ \boldsymbol{0} & 0 & 0 & \boldsymbol{5} & 0 & 0 \\ \mathbf{3} & 0 & \boldsymbol{4} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{6} \end{matrix}$$

Procedure *TTProblem* gives only $10 = 5+6-1$ elements of the matrix X denoted by $\mathbf{x}$ and $\boldsymbol{x}$ for (k, l) of the form $(B[j], j)$ and $(i, A[i])$, respectively. Thus,

$B[j]$	j	$W[j]$	i	$A[i]$	$V[i]$
3	1	0	1	2	3
2	2	3	2	4	2
4	3	4	4	1	3
3	4	5	5	6	6
1	5	3			
1	6	2	$(d = 3)$,		

$tX := 8$, $(t(X) = t_{34} = 8)$;

```
begin
    integer i, j, k, l, p, q, r, s, u;
    begin
        comment This block calculates an initial solution by
            the minimum row method and locates the result into
            arrays A, V, B, W in the same way as procedure
            TTProblem does it with an optimal solution, (d: = m);
        for i: = 1 step 1 until n do W[i]: = -1;
        i: = 1;
        k: = m+n-1;
        for l: = 1 step 1 until k do
            begin
                p: = -1;
```

```
                    for j: = 1 step 1 until n do
                        begin
                            u: = T[i, j];
                            if W[j] < 0 ∧ (p < 0 ∨ u < q)
                                then begin
                                        r: = j;
                                        q: = u;
                                        p: = 1
                                     end W[j] < 0 ∧ (p < 0 ∨ u < q)
                        end j;
                    if A[i] < B[r]
                        then begin
                            V[i]: = A[i];
                            B[r]: = B[r]-A[i];
                            A[i]: = r;
                            i: = i+1
                         end A[i] < B[r]
                        else begin
                            W[r]: = B[r];
                            A[i]: = A[i]-B[r];
                            B[r]: = i
                         end A[i] ≥ B[r]
                end l;
            d: = m
        end of block which calculates an initial solution;
    A[m]: = 1;
    V[m]: = 0;
F1: i: = -1;
F2: for j: = 1 step 1 until n do
        begin
            r: = B[j];
            s: = T[r, j];
            if W[j] > 0 ∧ (i < 0 ∨ tX < s)
                then begin
                        tX: = s;
                        k: = r;
                        l: = j;
                        i: = 1
                     end W[j] > 0 ∧ (i < 0 ∨ tX < s)
        end j;
    for j: = 1 step 1 until d-1, d+1 step 1 until m do
        begin
            r: = A[j];
```

```
                s: = T[j, r];
                if V[j] > 0 ∧ (i < 0 ∨ tX < s)
                    then begin
                            tX: = s;
                            k: = j;
                            l: = r;
                            i: = 1
                        end V[j] > 0 ∧ (i < 0 ∨ tX < s)
            end j;
        i: = k;
        j: = A[k];
        q: = V[k];
        for s: = B[j] while i ≠ d do
            begin
                r: = W[j];
                B[j]: = i;
                W[j]: = q;
                i: = s;
                q: = r;
                p: = A[i];
                r: = V[i];
                A[i]: = j;
                V[i]: = q;
                j: = p;
                q: = r
            end s;
        d: = k;
        A[k]: = 0;
        B[l]: = -B[l];
E1:     q: = 0;
        for i: = 1 step 1 until m do
            begin
                r: = A[i];
                if if r ⩽ 0 then false else B[r] < 0
                    then begin
                            A[i]: = -r;
                            q: = 1
                        end
            end i;
        if q ≠ 0
            then begin
                    q: = 0;
                    for j: = 1 step 1 until n do
```

```
                    begin
                         r: = B[j];
                         if if r < 0 then false else A[r] < 0
                              then begin
                                        B[j]: = -r;
                                        q: = 1
                                   end
                    end j;
               if q = 1
                    then go to E1
          end q ≠ 0;
r: = tX;
for j: = 1 step 1 until n do
     if B[j] < 0
          then begin
                    for i: = 1 step 1 until m do
                         begin
                              u: = T[i, j];
                              if A[i] ≥ 0 ∧ u < r
                                   then begin
                                             r: = u;
                                             p: = i;
                                             q: = j
                                        end A[i] ≥ 0 ∧ u < r
                         end i;
                    B[j]: = -B[j]
               end B[j] < 0, j;
 for i: = 1 step 1 until m do A[i]: = abs(A[i]);
 if r ≠ tX
      then begin
                r: = k;
                A[k]: = 1;
                u: = W[l];
                j: = q;
                for i: = B[j] while j ≠ l do
                     begin
                          if W[j] < u
                               then begin
                                         u: = W[j];
                                         r: = B[j]
                                    end W[j] < u;
                          j: = A[i]
                end i;
```

```
i: = p;
for j: = A[i] while i ≠ k do
    begin
        if V[i] < u
            then begin
                    u: = V[i];
                    r: = i
                 end V[i] < u;
        i: = B[j]
    end j;
j: = q;
for i: = B[j] while j ≠ l do
    begin
        W[j]: = W[j]-u;
        j: = A[i];
        V[i]: = V[i]+u
    end i;
i: = p;
for j: = A[i] while i ≠ k do
    begin
        V[i]: = V[i]-u;
        i: = B[j];
        W[j]: = W[j]+u
    end j;
W[l]: = W[l]-u;
d: = r;
i: = 0;
for s: = W[q] while i ≠ k do
    begin
    i: = B[q];
        B[q]: = p;
        W[q]: = u;
        p: = i;
        u: = s;
        j: = A[p];
        s: = V[p];
        A[p]: = q;
        V[p]: = u;
        q: = j;
        u: = s
    end s;
    i: = 1;
    go to if k = r
```

then *F1*
else *F2*

end $r \neq tX$

end *TTProblem*

2. Method used. The procedure *TTProblem* finds an optimal solution of the time transportation problem by the method described in [1]. The procedure *TTProblem* is a simple and precise realization of that method and for any time transportation problem gives an exact result. An initial solution is calculated by the minimum row method.

3. Certification. The example given in the commentary to the procedure *TTProblem* has been solved and correct results obtained, the computer being Odra 1204.

Reference

[1] A. Janicki, *Remarks on the time transportation problem*, Zastosow. Matem. 11 (1970), 493-502.

DEPT. OF NUMERICAL METHODS
UNIVERSITY OF WROCŁAW

Received on 26. 6. 1970

ALGORYTM 9

A. JANICKI (Wrocław)

ROZWIĄZANIE ZAGADNIENIA TRANSPORTOWEGO Z KRYTERIUM CZASU

Streszczenie

Procedura *TTProblem* rozwiązuje zagadnienie transportowe z kryterium czasu, tzn. konstruuje macierz $X = (x_{ij})$ typu $m \times n$, dla której funkcja

$$t(X) = \max_{(i,j)\in\Theta(X)} t_{ij}, \quad \Theta(X) = \{(i,j): x_{ij} > 0,\ 1 \leqslant i \leqslant m,\ 1 \leqslant j \leqslant n\}$$

określona na zbiorze macierzy o wymiarach $m \times n$ spełniających dla $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$ następujące warunki

$$x_{ij} \geqslant 0, \quad \sum_{j=1}^{n} x_{ij} = a_i, \quad \sum_{i=1}^{m} x_{ij} = b_j,$$

przy czym

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j, \quad a_i > 0, \quad b_j > 0,$$

osiąga minimalną wartość.

Dane:

m, n — wymiary zagadnienia transportowego (m ilość dostawców, n ilość odbiorców),
$A[1:m]$ — wektor $(a_1, a_2, \ldots, a_m)$ (ilości towarów oferowane przez dostawców),

$B[1:m]$ — wektor $(b_1, b_2, \dots, b_n)$ (ilości towarów zamówione przez odbiorców),
$T[1:m, 1:n]$ — macierz $T = (t_{ij})$ (macierz jednostkowych czasów transportu).

Wyniki:

Wynik końcowy, macierz $X = (x_{ij})$ typu $m \times n$ zawiera co najwyżej $m+n-1$ elementów $x_{ij} \neq 0$. W procedurze *TTProblem* wykorzystuje się tę własność rozwiązania dla skrócenia czasu wykonywania obliczeń i dlatego w wyniku otrzymuje się tylko $m+n-1$ elementów macierzy X, przedstawionych w następujący sposób:

A, $V[1:m]$ — jeśli $A[i] := j$, to $V[i] := x_{ij}$, $i = 1, \dots, d-1, d+1, \dots, m$,
d — liczba wskazująca, że należy pominąć $A[d]$ i $V[d]$,
B, $W[1:n]$ — jeśli $B[j] := i$, to $W[j] := x_{ij}$, $j = 1, 2, \dots, n$,
tX — wartość funkcji $t(X)$.

Procedura *TTProblem* jest dokładną i stosunkowo prostą realizacją metody opublikowanej w [1].

А. ЯНИЦКИ (Вроцлав)

АЛГОРИТМ 9

РЕШЕНИЕ ТРАНСПОРТНОЙ ЗАДАЧИ С КРИТЕРИЕМ ВРЕМЕНИ

РЕЗЮМЕ

Процедура *TTProblem* решает транспортную задачу с критерием времени или, говоря точно, находит матрицу $X = (x_{ij})$ порядка $m \times n$, для которой функция

$$t(X) = \max_{(i,j)\in\Theta(X)} t_{ij}, \qquad \Theta(X) = \{(i,j): x_{ij} > 0,\ 1 \leqslant i \leqslant m,\ 1 \leqslant j \leqslant n\}$$

определена в совокупности всех матриц порядка $m \times n$ удовлетворяющих для $i = 1, 2, \dots, m$ и $j = 1, 2, \dots, n$ условиям

$$x_{ij} \geqslant 0, \qquad \sum_{j=1}^{n} x_{ij} = a_i, \qquad \sum_{i=1}^{m} x_{ij} = b_j,$$

причем

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j, \qquad a_i > 0, \qquad b_j > 0$$

принимает минимальное значение.

Данные

m, n — порядок транспортной задачи (m количество поставщиков, n количество потребителей),

$A[1:m]$ — вектор $(a_1, a_2, \ldots, a_m)$ (количества товаров предлагаемые поставщиками),

$B[1:n]$ — вектор $(b_1, b_2, \ldots, b_n)$ (количества товаров нужны потребителям),

$T[1:m, 1:n]$ — матрица $T = (t_{ij})$ (t_{ij} обозначает время необходимое для поставки товара от i-того поставщика j-тому потребителю).

Решение:

В матрицы $X = (x_{ij})$ порядка $m \times n$, которая является решением транспортной задачи найденным при помощи процедуры *TTProblem*, не больше чем $m+n-1$ элементов $x_{ij} \neq 0$ и они найдены в следующем виде:

A, $V[1:m]$ — если $A[i] := j$, то $V[i] := x_{ij}$, $i = 1, 2, \ldots, d-1, d+1, \ldots, m$, d — число, которое показывает, что надо отбросить $A[d]$, $V[d]$,

B, $W[1:n]$ — если $B[j] := i$, то $W[j] := x_{ij}$, $j = 1, 2, \ldots, n$, tX — значение функции $t(X)$.

Процедура *TTProblem* это точная и сравнительно простая реализация метода опубликованного в [1].