ZASTOSOWANIA MATEMATYKI
APPLICATIONES MATHEMATICAE
XVI, 4 (1980)

ALGORITHM 69

M. KAROŃSKI and Z. PALKA (Poznań)

TWO DISTANCES BETWEEN HYPERGRAPHS

1. Procedure declaration. The procedure *hypergraphmetrics* calculates two distances between hypergraphs proposed in [2]. Suppose that $H_1 = (X, \mathscr{E}_1)$ and $H_2 = (X, \mathscr{E}_2)$ are two hypergraphs, both having the same set of vertices and different sets of edges $\mathscr{E}_1 = (E_{1i}; i \in I_1)$ and $\mathscr{E}_2 = (E_{2j}; j \in I_2)$. Then a distance between H_1 and H_2, as defined in [2], is

$$\varrho_1(H_1, H_2) = \frac{1}{2}\left[\max_{i\in I_1}\min_{j\in I_2}\sigma_\mu(E_{1i}, E_{2j}) + \max_{j\in I_2}\min_{i\in I_1}\sigma_\mu(E_{1i}, E_{2j})\right]. \tag{1}$$

We also considered the hypergraphs generated by arborescences with the same set of pendant vertices X. Let an arborescence A be represented by a hypergraph $H_A = (X, \mathscr{E}_A)$, where the class of edges $\mathscr{E}_A$ is defined in the following way.

Each non-pendant vertex x in the arborescence A generates $d^+(x) - 1$ identical edges in $\mathscr{E}_A$, where $d^+(x)$ denotes the outer demi-degree of the vertex x. Such an edge consists of those elements of the set X which are pendant vertices of the subarborescence generated by the vertex x. The distance between $H_{A_1} = (X, \mathscr{E}_{A_1})$ and $H_{A_2} = (X, \mathscr{E}_{A_2})$, as defined in [2], is

$$\varrho_2(H_{A_1}, H_{A_2}) = \frac{1}{n-1}\min_{p\in P}\sum_{i=1}^{n-1}\sigma_\mu(E^i_{A_1}, E^{p_i}_{A_2}), \tag{2}$$

where p_i is the i-th element of the permutation p of the first $n-1$ integers, P is the set of all such permutations, and $E^i_{A_1} \in \mathscr{E}_{A_1}, E^{p_i}_{A_2} \in \mathscr{E}_{A_2}$ $(i = 1, 2, \ldots, n-1)$. In (1) and (2), $\sigma_\mu(\cdot, \cdot)$ denotes the Marczewski-Steinhaus distance between two sets (see [2] or [4]).

In the procedure *hypergraphmetrics* we use the representation of a hypergraph $H = (X, \mathscr{E})$ by the vector h. Let $N = \{1, 2, \ldots, n\}$, where $n = |X|$, and let l: $X \mapsto N$ be a one-to-one labeling of the vertex set X. Suppose that the hypergraph H has r edges $E_1, E_2, \ldots, E_r$. An edge of the hypergraph H is represented by a sequence of labels of its elements ended by -1, and all such r sequences (in an arbitrary order) form the repre-

sentation vector h. If the labeling of the elements of an edge consists of a subsequence of more than two consecutive integers, say $i, i+1, \ldots, j-1, j$, then this subsequence can be replaced by a triple $i, 0, j$. For example, let

$$X = \{x_1, x_2, \ldots, x_7\}$$

and

$$E_1 = \{x_1, x_2, x_5\}, \quad E_2 = \{x_1, x_3, x_4, x_5, x_6, x_7\}, \quad E_3 = \{x_4\}.$$

Suppose that $x_i \leftrightarrow i$ for $i = 1, 2, \ldots, 7$; then the hypergraph $H = (X, \mathscr{E})$ is represented by

$$h = [1, 2, 5, -1, 1, 3, 0, 7, -1, 4, -1].$$

Data:

n1, *n2* — sizes of the arrays *h1*, *h2*, respectively;

h1[*1* : *n1*], *h2*[*1* : *n2*] — arrays representing the first and the second hypergraphs, respectively;

fac — integer scaling factor, needed for the representation of real distances as integers without loss of information; it is preferable to set *fac* = 1000000;

first — Boolean variable: if *first* equals **true**, then the procedure calculates the distance (1), else the distance (2);

inf — maximal allowed number of type **integer** in the computer;

assign — the procedure finding $\min \sum_{i=1}^{n} d_{ip_i}$, where $(p_1, p_2, \ldots, p_n)$ is the permutation of the numbers $(1, 2, \ldots, n)$; the procedure heading should be of the form

procedure *assign*(*n*, *d*, *p*, *total*, *inf*);
value *n*, *inf*; **integer** *n*, *total*, *inf*;
integer array *d*, *p*;

where

n — size of the matrix *d*;

d[*1* : *n*, *1* : *n*] — matrix of costs;

p[*1* : *n*] — permutation of indices which minimizes the cost function;

total — minimal value of the cost function;

inf — maximal allowed number of type **integer** in the computer.

The procedure *assign* is published in [3].

Result:

d — distance between hypergraphs.

```
procedure hypergraphmetrics(n1,n2,h1,h2,fac,first,inf,d,
 assign);
 value n1,n2,fac,inf;
 integer n1,n2,fac,inf;
 Boolean first;
 real d;
 integer array h1,h2;
 procedure assign;
 begin
  integer i,j,k,k1,k2,l,l1,l2,t,s,s1,s2;
  procedure size(h,j,k,l);
   value j;
   integer j,k,l;
   integer array h;
   begin
    integer hi,i;
    k:=l:=0;
    i:=1;
next:
    hi:=h[i];
    if hi=0
     then
     begin
      k:=k+h[i+1]-h[i-1];
      i:=i+1
     end hi=0
     else
      if hi≠-1
       then k:=k+1
       else l:=l+1;
```

```
  i:=i+1;
  if i≤j
   then go to next
 end size;
size(h1,n1,k1,l1);
size(h2,n2,k2,l2);
begin
 integer array a1[1:k1],a2[1:k2],b1[1:l1],b2[1:l2],c[1:l1,
 1:l2],e[1:l1];
 procedure maxmin(c,t1,t2,w,max);
  value t1,t2,w;
  Boolean w;
  integer t1,t2,max;
  integer array c;
  begin
   integer cij,i,j,min;
   max:=0;
   for i:=1 step 1 until t1 do
    begin
     min:=if w then c[i,1] else c[1,i];
     for j:=2 step 1 until t2 do
      begin
       cij:=if w then c[i,j] else c[j,i];
       if cij<min
        then min:=cij
      end j;
     if min>max
      then max:=min
    end i
  end maxmin;
```

```
    procedure tables(m,a,b,c);
     value m;
     integer m;
     integer array a,b,c;
     begin
      integer i,j,k,p,q,s,t;
      s:=t:=1;
      for i:=1 step 1 until m do
       begin
        k:=0;
et:     a[t]:=p:=c[s];
        k:=k+1;
        s:=s+1;
        t:=t+1;
        if c[s]=0
         then
         begin
          q:=c[s+1];
          s:=s+2;
          for j:=p+1 step 1 until q do
           begin
            a[t]:=j;
            t:=t+1
           end j;
          k:=k+q-p
         end c[s]=0;
        if c[s]≠-1
         then go to et;
        b[i]:=k;
        s:=s+1
```

```
      end i
    end tables;
   tables(l1,a1,b1,h1);
   tables(l2,a2,b2,h2);
   k1:=0;
   for i:=1 step 1 until l1 do
    begin
     s1:=k1+1;
     k1:=k1+b1[i];
     k2:=0;
     for j:=1 step 1 until l2 do
      begin
       t:=0;
       s2:=k2+1;
       k2:=k2+b2[j];
       for l:=s1 step 1 until k1 do
        begin
         s:=a1[l];
         for k:=s2 step 1 until k2 do
          if s=a2[k]
           then
           begin
            t:=t+2;
            go to end
           end k,s=a2[k];
end:     end l;
       s:=b1[i]+b2[j];
       c[i,j]:=(s-t)/(s-t÷2)×fac
      end j
    end i;
```

```
    if first
     then
     begin
      maxmin(c,i1,i2,true,s1);
      maxmin(c,i2,i1,false,s2);
      d:=(s1+s2)/(fac+fac)
     end first
     else
     begin
      assign(i1,c,e,s,inf);
      d:=s/(i1×fac)
     end ¬first
   end block
end hypergraphmetrics
```

Remark. It should be noticed that if we want to compare two hypergraphs using the distance (2), then these hypergraphs ought to have the same number of edges (i.e. the same number of -1 should be in both arrays *h1* and *h2*).

2. Method used. For given hypergraphs $H_1 = (X, \mathscr{E}_1)$ and $H_2 = (X, \mathscr{E}_2)$, where $\mathscr{E}_1 = \{E_{11}, E_{12}, \ldots, E_{1l}\}$, $\mathscr{E}_2 = \{E_{21}, E_{22}, \ldots, E_{2k}\}$, we construct a matrix $[c_{ij}]$ $(i = 1, 2, \ldots, l;\ j = 1, 2, \ldots, k)$ with

$$c_{ij} = \sigma_{\mu_c}(E_{1i}, E_{2j}) = \frac{e_{1i} + e_{2j} - 2d_{ij}}{e_{1i} + e_{2j} - d_{ij}},$$

and $e_{1i} = |E_{1i}|$, $e_{2j} = |E_{2j}|$, $d_{ij} = |E_{1i} \cap E_{2j}|$ (for details see [2]). In the case of calculating the distance ϱ_1, we use formula (1) straightforward. It can be shown that the computation of the distance ϱ_2 is equivalent to the problem of optimal assignment [1]. We compute ϱ_2 from the formula

$$\varrho_2 = \frac{1}{n-1} \min_{p \in P} \sum_{i=1}^{n-1} c_{ip_i}, \tag{3}$$

where $p = (p_1, p_2, \ldots, p_{n-1})$ denotes a permutation from the set P of all permutations of the integers $(1, 2, \ldots, n-1)$ and $n-1 = l = k$. In order to find the value of (3) we use the procedure *assign* from [3].

3. Certification. The procedure has been verified on the Odra 1204 computer for the problems described in [2]. All results were correct.

References

[1] L. Ford and D. Fulkerson, *Flows in networks*, Princeton University Press, Princeton, N. J., 1962.

[2] M. Karoński and Z. Palka, *On Marczewski-Steinhaus type distance between hypergraphs*, Zastosow. Matem. 16 (1977), p. 47-57.

[3] J. Kucharczyk and M. Sysło, *Algorytmy optymalizacji w języku ALGOL 60*, Warszawa 1975.

[4] E. Marczewski and H. Steinhaus, *On a certain distance of sets and the corresponding distance of functions*, Coll. Math. 6 (1958), p. 319-327.

INSTITUTE OF MATHEMATICS
A. MICKIEWICZ UNIVERSITY
60-769 POZNAŃ

Received on 25. 7. 1977

ALGORYTM 69

M. KAROŃSKI i Z. PALKA (Poznań)

DWIE ODLEGŁOŚCI MIĘDZY HIPERGRAFAMI

STRESZCZENIE

Procedura *hypergraphmetrics* oblicza odległość (1) lub (2) między danymi dwoma hipergrafami w zależności od parametru *first*.

Dane:

n1, *n2* — wymiary odpowiednio tablic *h1* i *h2*;

h1[*1* : *n1*], *h2*[*1* : *n2*] — tablice reprezentujące odpowiednio pierwszy i drugi hipergraf;

fac — mnożnik umożliwiający przedstawienie odległości za pomocą liczb całkowitych bez straty informacji; zaleca się przyjąć *fac* = 1000000;

first — zmienna boolowska: jeżeli *first* ≡ **true**, to stosujemy odległość (1), w przeciwnym razie — odległość (2);

inf — największa dopuszczalna w maszynie cyfrowej liczba typu **integer**;

assign — procedura znajdująca $\min \sum_{i=1}^{n} d_{ip_i}$, gdzie $(p_1, p_2, \ldots, p_n)$ jest permutacją liczb $(1, 2, \ldots, n)$; nagłówek procedury powinien być następujący:
procedure *assign*(*n*, *d*, *p*, *total*, *inf*);
value *n*, *inf*; **integer** *n*, *total*, *inf*;
integer array *d*, *p*;
gdzie

n — rozmiar tablicy *d*;

d[*1* : *n*, *1* : *n*] — macierz kosztów;

p[*1* : *n*] — permutacja wskaźników, dająca minimum funkcji celu;

total — minimalna wartość funkcji celu;

inf — największa dopuszczalna w maszynie liczba typu **integer**.

W procedurze *hypergraphmetrics* w przypadku odległości (1) zastosowano metodę bezpośredniego obliczenia jej wartości. W przypadku odległości (2) zauważono, że z obliczeniowego punktu widzenia zagadnienie to jest równoważne problemowi optymalnego przydziału (patrz [1]), i skorzystano z procedury *assign* (patrz [3]). Procedura *hypergraphmetrics* była sprawdzana na maszynie cyfrowej Odra 1204. Obliczono odległości między hipergrafami przedstawionymi w pracy [2].