Homework 8

100
excellent!

Jingsai Liang

1,

I use MATLAB to do this problem and following is the result:

```
% data matrix
Obj=[1 18 120 1 0 0;
     0 36 89 0 1 0.5;
     1 20 115 1 1 0;
     1 3 94 1 0 1;
     0 28 110 0 1 1;
     0 44 80 0 1 0.5];
% standalize the intevral value
Obj(:,2)=(Obj(:,2)-mean(Obj(:,2)))/std(Obj(:,2));
Obj(:,3)=(Obj(:,3)-mean(Obj(:,3)))/std(Obj(:,3));
% calculate the distance
dist=zeros(6);
for i=1:6
    for j=i+1:6
        delta=6;
        if Obj(i,4)==0 && Obj(j,4)==0
            delta=delta-1;
        end
        if Obj(i,5)==0 && Obj(j,5)==0
            delta=delta-1;
        end
        dist(i,j)=sum(abs(Obj(i,:)-Obj(j,:)))/delta;
        dist(j,i)=dist(i,j);
    end
end
```

dist =

0	1.1145	0.2419	0.7332	0.8863	1.3006
1.1145	0	0.8725	1.0153	0.4739	0.2234
0.2419	0.8725	0	0.7485	0.6443	1.0587
0.7332	1.0153	0.7485	0	0.9549	1.2015
0.8863	0.4739	0.6443	0.9549	0	0.6972
1.3006	0.2234	1.0587	1.2015	0.6972	0

2,

I use MATLAB to do this problem and following is the result:

```
S2=[]; % belong to 2
S4=[]; % belong to 4
costp=0;
for i=[1 3 5 6]
   if dist(i,2) < dist(i,4)
       S2=[S2 i];
       costp=costp+dist(i,2);
   else
       S4=[S4 i];
       costp=costp+dist(i,4);
   end
end

S2=[]; % belong to 2
S3=[]; % belong to 3
costc=0;
for i=[1 4 5 6]
   if dist(i,2) < dist(i,3)
       S2=[S2 i];
       costc=costc+dist(i,2);
   else
       S3=[S3 i];
       costc=costc+dist(i,3);
   end
end
```

a, Obj5 and Obj6 are assigned to the cluster of Obj2. Obj1 and Obj3 are assigned to the cluster of Obj4. Total cost is 2.1789.

b, After replacement, Obj5 and Obj6 are assigned to the cluster of Obj2. Obj1 and Obj4 are assigned to the cluster of Obj3. Total cost is 1.6876.

The cost is decrease. So we should use Obj3 to replace Obj4.

3,

I use MATLAB to do this problem and following is the result:

```
n=6;
% init
for i=1:n
    node{i}=i;
end
for k=1:n-1
    dmin=100;
    % each merge
    for i=1:n-k
        for j=i+1:n-k+1
            % calculate dist between nodes
            % average distance
            dmean=0;
            for p=1:length(node{i})
                for q=1:length(node{j})
                    dmean=dmean+dist(node{i}(p),node{j}(q));
                end
            end
            dmean=dmean/(length(node{i})*length(node{j}));
            % compare dist among all pairs
            if dmean<dmin
                dmin=dmean;
                nodemin=[i,j];
            end

        end
    end
    % update node
    oldnode=node;
    node={};
    nodemin=sort(nodemin);
    node{1}=[oldnode{nodemin(1)}, oldnode{nodemin(2)}];
    j=2;
    for i=1:n-k+1
        if i ~= nodemin(1) && i~= nodemin(2)
            node{j}=oldnode{i};
            j=j+1;
        end
    end
    disp(strcat('New Level ',num2str(k),': ',num2str(node{1})))
    disp(strcat('distance: ',num2str(dmin)))
end
```

Output:

New Level1:2 6

distance:0.22335

New Level2:1 3

distance:0.24194

New Level3:2 6 5

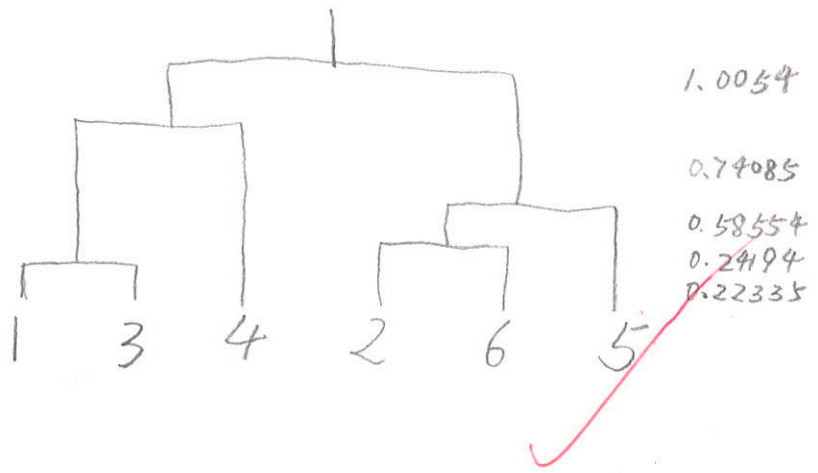
distance:0.58554

New Level4:1 3 4

distance:0.74085

New Level5:1 3 4 2 6 5

distance:1.0054



4.

a)

C_1		L_1		C_2		L_2		C_3		L_3
A	4	A	4	AB	4	AB	4	ABC	2	ABC	2
B	4	B	4	AC	2	AC	2	ABD	3	ABD	3
C	2	C	2	AD	3	AD	3	ABE	2	ABE	2
D	3	D	3	AE	2	AE	2	ACE	2	ACE	2
E	2	E	2	BC	2	BC	2	BCE	2	BCE	2
K	1			BD	3	BD	3				
				BE	2	BE	2				
				CD	1	CE	2				
				CE	2						
				DE	1						

C_4	L_4
→ ABCE 2	→ ABCE 2

b) pick ABCE

A → BCE	2/4 = 50% ×	BCE → A	2/2 = 100% ✓	AB → CE	2/4 = 50% ×			
B → ACE	2/4 = 50% ×	ACE → B	2/2 = 100% ✓	AC → BE	2/2 = 100% ✓			
C → ABE	2/2 = 100% ✓	ABE → C	2/2 = 100% ✓	AE → BC	2/2 = 100% ✓			
E → ABC	2/2 = 100% ✓	ABC → E	2/2 = 100% ✓	BC → AE	2/2 = 100% ✓			
				BE → AC	2/2 = 100% ✓			
				CE → AB	2/2 = 100% ✓			

pick ABC

A → BC	2/4 = 50% ×	BC → A	2/2 = 100% ✓
B → AC	2/4 = 50% ×	AC → B	2/2 = 100% ✓
C → AB	2/2 = 100% ✓	AB → C	2/4 = 50% ×

pick ABD

A → BD	3/4 = 75% ✓	AB → D	3/4 = 75% ✓
B → AD	3/4 = 75% ✓	AD → B	3/3 = 100% ✓
D → AB	3/3 = 100% ✓	BD → A	3/3 = 100% ✓

pick ABE

A → BE	2/4 = 50% ×	AB → E	2/4 = 50% ×
B → AE	2/4 = 50% ×	AE → B	2/2 = 100% ✓
E → AB	2/2 = 100% ✓	BE → A	2/2 = 100% ✓

Pick ACE	A→CE	2/4 = 50% ×	AC→E	2/2 = 100% ✓
	C→AE	2/2 = 100% ✓	AE→C	2/2 = 100% ✓
	E→AC	2/2 = 100% ✓	CE→A	2/2 = 100% ✓
Pick BCE	B→CE	2/4 = 50% ×	BC→E	2/2 = 100% ✓
	C→BE	2/2 = 100% ✓	CE→B	2/2 = 100% ✓
	E→BC	2/2 = 100% ✓	BE→C	2/2 = 100% ✓
Pick AB	A→B	4/4 = 100% ✓	B→A	4/4 = 100% ✓
Pick AC	A→C	2/4 = 50% ×	C→A	2/2 = 100% ✓
pick AD	A→D	3/4 = 75% ✓	D→A	3/3 = 100% ✓
pick AE	A→E	2/4 = 50% ×	E→A	2/2 = 100% ✓
pick BC	B→C	2/4 = 50% ×	C→B	2/2 = 100% ✓
pick BD	B→D	3/4 = 75% ✓	D→B	3/3 = 100% ✓
pick BE	B→E	2/4 = 50% ×	E→B	2/2 = 100% ✓
pick CE	C→E	2/2 = 100% ✓	E→C	2/2 = 100% ✓

5,

(a) Given $S' \subset S$.

If S in one transaction T, we denote as $S \subset T$. So we have $S' \subset S \subset T$, which means S' in this transaction too.

So $Support(S') \geq Support(S)$

(b) Given $S' \subset S$.

From (a), we know $Support(S') \geq Support(S)$

So $\frac{Support(S')}{Support(l)} \geq \frac{Support(S)}{Support(l)}$

so $P(S' \rightarrow l - S') \geq P(S \rightarrow l - S)$

(c) Find all candidates in a transaction t:

At first, the condition is that all items in any sets are ordered. Beginning from the root, we hash every item in t. Then we continue to next level. If we reach one node by hashing item i, we hash every item after i in the t. Until we reach the leaf, we count the itemset in this leaf, which is a subset in the transaction t.