

УДК 621.391:519.2

АЛГОРИТМ ЗАПОЛНЕНИЯ ПРОПУСКОВ В ЭМПИРИЧЕСКИХ ТАБЛИЦАХ (АЛГОРИТМ "ZET")

Н.Г.Загоруйко, В.Н.Ёлкина, В.С.Тимеркаев

Эмпирические данные обычно представляются в виде прямоугольной матрицы, в которой, например, строки соотнесены с исследуемыми объектами, а столбцы - с их характеристиками или приборами, измеряющими эти характеристики.

На пересечении строки и столбца записывается результат взаимодействия данного объекта с данным прибором. На практике иногда встречаются таблицы, часть элементов которых по той или иной причине оказываются незаполненными.

В работе [1] приведен пример таблиц, в которой все строки и столбцы имеют хотя бы по одному пропущенному элементу. Несмотря на это, если между значениями имеющихся элементов удастся обнаружить какие-либо закономерные связи, то можно попытаться использовать эти закономерности для угадывания значений и пропущенных элементов.

В описываемом здесь алгоритме заполнения пропусков в эмпирических таблицах (алгоритм "ZET") используется следующая эвристическая гипотеза, широко распространенная в среде распознавателей: объекты, мало отличающиеся друг от друга по n своим свойствам, имеют приблизительно одинаковое значение n -го свойства. Чем больше n и чем меньше различий между объектами по этим n свойствам, тем больше надежд возлагается на эту гипотезу. Понятия "большое" и "малое" отличия устанавливаются произвольно. Если "малыми" считать все отличия, встречающиеся в таблице, то место пропущенного элемента можно заполнить любым из имеющихся значений соответствующего признака. В другом край-

нем случае, когда "мало" отличающимися считаются только совершенно одинаковые объекты, пропущенный элемент естественно заполнять средним значением $(n+1)$ -й характеристики объектов только такой "референтной" группы, в состав которой входят объекты, неразличимые по n характеристикам. В излагаемом алгоритме "референтная" группа объектов обнаруживается в результате применения таксономической процедуры [2-3]. Этой же процедурой может быть найдена и "референтная" группа признаков (т.е. столбцов таблицы), по которой так или иначе можно судить о значении пропущенного элемента.

Существенным моментом при решении задач таксономии является способ определения "отличия". В тех простых случаях, с которыми мы имели дело в описываемых ниже иллюстративных примерах, мерой "отличия" двух объектов считалось расстояние между ними в n -мерном евклидовом пространстве, а мерой "отличия" двух признаков - величина $R = (1 - |z_{ij}|)$, где z_{ij} - коэффициент их взаимной корреляции. В более общем случае, когда признаки измерены в разнородных шкалах, среди которых имеются шкалы наименований и шкалы порядка, выбор метрики должен рассматриваться специально.

Вопрос о том, на сколько групп (таксонов) следует разделить множество объектов (или признаков), будем решать, опираясь на критерий F качества таксономии, используемый в алгоритме "Краб" [4]:

$$F = \frac{h \cdot d}{\rho \cdot \lambda},$$

где d - среднее расстояние между таксонами, ρ - среднее расстояние между точками внутри таксона, λ - мера "однородности структуры" расположения точек внутри таксона, h - мера "одинаковости количества точек" в разных таксонах.

Точная оценка "качества" восстановления пропущенных элементов, вообще говоря, формально неопределима, так как это зависит в первую очередь от того, насколько хорошо закономерность, используемая для восстановления и находящаяся по части таблицы, будет соответствовать закономерности, которую можно было бы найти по полной таблице. Однако некоторую приближенную оценку можно получить, проверив "точность предсказания" известных элементов таблицы. Такая возможность предусмотрена в описываемом здесь алгоритме (и программе) "ZET".

Теперь рассмотрим поэтапно процесс работы алгоритма "ZET".

1. Выбираем в матрице A исходных данных строку x_i , содержащую пробел a_{ij} .

2. Из матрицы A исключим все строки, кроме рассматриваемой строки x_i , содержащие пробелы в j -м столбце. Обозначим полученную матрицу символом A_1 .

3. Строим для матрицы A_1 все не содержащие пробелов подматрицы A_1 размерности $n_l \times m_l$, где n_l - число столбцов, m_l - число строк. В программе принято, что $n_l, m_l \geq 3$. Если не найдем ни одной такой матрицы $A_1^{(l)}$, то на этом этапе элемент a_{ij} не восстанавливается, программа переходит к п.1 - поиску следующего пробела матрицы A . Если матрица A такова, что ни для одного из пропущенных элементов не удастся найти подматрицу $A_1^{(l)}$, то восстановление матрицы A невозможно. В этом случае программа печатает текст "матрица не восстановлена", и работа алгоритма прекращается.

4. Для каждой подматрицы $A_1^{(l)}$ выполняем таксономию строк по алгоритму "Фораль" [2] и по критерию качества F отбираем лучший для всех $A_1^{(l)}$ вариант разбиения, при котором соответствующая часть исследуемой строки x_i попала в таксон, содержащий более одного элемента.

5. Выбираем в матрице A столбец x_j , содержащий пробел a_{ij} .

6. Строим матрицу A_2 , исключая из A все столбцы, кроме столбца x_j , содержащие пробелы в i -й строке.

7. Находим все подматрицы $A_2^{(k)}$ матрицы A_2 , не содержащие пробелов и имеющие размерность не меньшую, чем $n_k \times m_k$.

8. Для всех $A_2^{(k)}$ выполняем таксономию столбцов по алгоритму "Фораль" и отбираем лучший по критерию качества F вариант разбиения, при котором x_j не выделяется в самостоятельный таксон.

9. При восстановлении каждого пропущенного значения a_{ij} сравниваем величину критерия F , полученного в п.4 и п.8 для строки x_i и столбца x_j . При $F_{x_i} \geq F_{x_j}$ в матрице A на место пробела a_{ij} ставим среднее значение j -го признака, вычисленное по строкам, попавшим в один таксон с x_i . При $F_{x_j} > F_{x_i}$ заполняем пробел a_{ij} по значениям столбцов матрицы A , попавших в один таксон с x_j следующим образом:

$$a_{ij} = \frac{\sum_{k=1}^p a_{i,k} \cdot |r_{j,k}|}{\sum_{k=1}^p |r_{j,k}|}; j \neq k,$$

где i - номер строки; j - номер столбца, на пересечении которых находятся a_{ij} ; $|r_{j,k}|$ - модуль коэффициента корреляции между j -м и k -м столбцами; p - количество столбцов, попавших в один таксон с x_j ; $a_{i,k}$ - величина, определяемая следующим образом:

$$a_{i,k} = \bar{a}_{i,k} + (\bar{a}_{i,k} - a_{i,k}) \cdot (1 - r_{j,k}),$$

где $\bar{a}_{i,k}$ - среднее значение элементов таксона в k -м столбце.

10. Для восстановления каждого из пропущенных элементов выполняем процедуры пп. I-9 по исходной матрице A , без учета уже заполненных пробелов.

II. Проверяем, насколько подтверждаются найденные закономерности на всей заполненной матрице B . Для этого, поочередно "вычеркивая" из матрицы B каждый из M восстановленных на предыдущем этапе элементов b_{α} , заново вычисляем тем же методом значение $b_{\alpha}^{(1)}$ по матрице B . Получаем новую матрицу B_1 . Подсчитываем для матриц B и B_1 значение $\beta = \frac{2}{M} \sum_{\alpha=1}^M \frac{b_{\alpha} - b_{\alpha}^{(1)}}{b_{\alpha} + b_{\alpha}^{(1)}}$ и сравниваем его с заданной константой $\xi > 0$.

При $\beta \leq \xi$ итерационный процесс прекращается. При $\beta > \xi$ заменяем матрицу B матрицей B_1 и повторяем п. II.

Поскольку вопрос о сходимости и скорости сходимости формально не решен, в алгоритме предусмотрен выход из цикла по количеству выполненных итераций.

Алгоритм исследовался на ряде конкретных примеров, взятых из различных областей науки и техники.

Первый пример представлял собой таблицу, в которой были собраны данные о телевизорах 27 марок. Среди признаков были такие, как размеры телевизора, чувствительность, разрешающая способность, выходная мощность звукового канала, наличие системы автоматического регулирования и т.д. - всего 21 признак. Из этой таблицы искусственно было изъято 67 элементов так, чтобы пропущенные элементы распределялись равномерно по таблице.

На восстановление таблицы было затрачено около 20 минут машинного времени на ЭВМ "БЭСМ-6", и на третьей итерации получены значения пропущенных элементов, отличающихся от истинных не более чем на 0,01%.

Второй пример был представлен таблицей, размером 29x13, в которой находились данные о бумаге разных стран. Среди признаков были такие, как масса бумаги в кв.м на грамм, объем массы в граммах на см³, белизна - %, гладкость по Беку (шкала порядка), зольность - %, состав по волокну - % и т.д. В этой таблице не хватало 136 элементов. Для проверки правильности работы алгоритма изъято еще 20 элементов, так что в результате требовалось восстановить половину таблицы. На восстановление таблицы затрачено 50 минут машинного времени ЭВМ "БЭСМ-6". На 6-й итерации работы алгоритма искусственно изъяты 20 элементов восстановились практически точно, значения остальных 136 восстановленных элементов, по мнению экспертов, также оказались весьма правдоподобными.

Последний пример характеризовал массив экспериментальных данных из области океанологии. Этот массив содержал результаты замеров значений скорости звука в воде на 16-ти стандартных горизонтах, произведенных в Атлантическом океане на разных географических широтах. Некоторые из замеров были пропущены, часть замеров искусственно изъята для проверки алгоритма. В матрице размерностью 20x16 требовалось восстановить 25 элементов.

Результаты восстановления оказались также вполне удовлетворительными. Время работы программы в этом случае было равно 10 минутам ЭВМ "БЭСМ-6". Для сходимости алгоритма понадобилось 5 итераций.

Инструкция к программе

Подпрограмма, реализующая алгоритм "ЖЕТ", составлена на языке ФОРТРАН для ЭВМ "БЭСМ-6" (см. приложение I). Обращение к подпрограмме осуществляется с помощью оператора CALL JET, 73 (XU1, XU2, XU3, XU4, L1S, L1D, KK1, KK2, KK3, KK4, KK5, L1R, L1F, XU4, KK6, KER)

XU1 - массив исходных данных размерности (KK1, KK2);
XU2 - массив рабочих ячеек размерности (7, KK3);
XU3 - массив рабочих ячеек размерности (KK1, KK3);
XU4 - одномерный массив рабочих ячеек, равный L1R;

$LIR = KK1 * KK3$ (перед обращением к подпрограмме массивы $XU3$ и $XU2$ должны быть объявлены эквивалентными оператором *EQUIVALENCE*);

LIS - массив рабочих ячеек размерности $(7, KK3)$;

LID - массив рабочих ячеек размерности $(3, KK3)$;

$KK1$ - количество элементов в строке исходной матрицы;

$KK2$ - количество строк исходной матрицы;

$KK3$ - максимальное из двух чисел $\{KK1, KK2\}$;

$KK4$ - *const* для вычисления значения шага ΔR по алгоритму "Фораль";

$KK5$ - *const* ; если $KK5 = 0$, то на печать выдаются результаты работы внутренних подпрограмм, в противном случае - печатаются только исходная и исправленная матрицы

$LIR = const, LIR = KK1 * KK2$.

В процессе работы подпрограмма "XETL 73" использует подпрограммы "TAXON", "TEMIR", "MATRIX", "CORREL", "RROT", и "MID".

Подпрограмма "TAXON" служит для выделения таксонов с помощью гиперсфер по алгоритму "Фораль".

Подпрограмма "TEMIR" производит построение кратчайшего незамкнутого пути на элементах каждого таксона и соединяет таксоны кратчайшим незамкнутым путем.

В качестве меры расстояния между таксонами выбрано расстояние между множествами в метрическом пространстве $d_{ij} = \inf \alpha(q_i, q_j)$, где q_i и q_j - произвольные точки из i -го и j -го таксонов. Подпрограмма "TEMIR" производит также оценку разбоя полученного подпрограммой "TAXON" по критерию F .

Подпрограмма "MATRIX" служит для печати матрицы.

Подпрограмма "CORREL" производит вычисление модуля коэффициента корреляции $\{\alpha = |r_{k,j}|\}$ между признаками x_k и x_j .

Подпрограмма "RROT" производит вычисление евклидова расстояния между объектами.

Подпрограмма "MID" определяет центр таксона.

Обращение к подпрограмме "TAXON" производится оператором $CALL TAXON(L1, L2, L3, L4, L5, L6, L7, L8, Z1, Z2, Z3, L9)$,

где

$L1$ - размерность признакового пространства;

$L2$ - количество реализаций;

$L3$ - минимальное необходимое количество таксонов;

$L4$ - максимальное количество таксонов;

$L5$ - *const* для вычисления шага по радиусу R ;

$L6$ - максимальное из чисел $(L1, L2)$;

$L7$ - *const* , если $L7 = 0$, то печатаются результаты подпрограммы "TAXON";

$L8$ - рабочая матрица размерности $(7, L6)$;

$Z1$ - матрица исходных данных размерности $(L1, L2)$;

$Z2$ - рабочая матрица размерности $(7, L6)$;

$Z3$ - *const* , после работы подпрограммы хранит максимальный критерий качества;

$L9$ - *const* хранит количество лучших по F критерию таксонов.

Подпрограмма "TEMIR" вызывается оператором

$CALL TEMIR(L1, L2, L3, L4, L5, L6, Z1, Z2, Z3)$,

где

$L1$ - размерность признакового пространства;

$L2$ - количество реализаций;

$L3$ - количество таксонов, полученных подпрограммой "TAXON";

$L4$ - максимальное из чисел $(L1, L2)$;

$L5$ - *const*, если $L5 = 0$, то происходит печать результатов подпрограммы.

$L8$ - рабочая матрица размерности $(7, L4)$ (перед работой программы в 4-м столбце матрицы должен находиться список принадлежности элементов таксону (например, 1, 1, 2, 5, 6, 6, 1. означает: 1 элемент - 1 таксону, 2 элемент - 1 таксону, 3 элемент - 2 таксону, 4 элемент 5 таксону и т.д.); в 5-м столбце матрицы должен находиться список количества элементов в таксонах, например, 3, 4, 5, 1 означает: в 1-м таксоне - 3 элемента, во 2-м таксоне - 4 элемента и т.д.);

$Z1$ - матрица исходных данных, размерности $(L1, L2)$;

$Z2$ - рабочая матрица, размерности $(7, L4)$;

$Z3$ - *const* , в $Z3$ после работы подпрограммы хранится критерий качества.

Подпрограмма "CORREL" вызывается оператором

$CALL CORREL(L1, L2, L3, X, X2)$,

где

$L1$ - количество элементов в строке подматрицы;

$L2$ - количество строк подматрицы;

Знаком (-) выделены типичные для своего таксона реализации. Например,

I	2	3	4	5	6
I	I	-I	-2	3	-3
-	-	-	-	-	-
9	I	2	8		

означает: I реализации в I-м таксоне; 2 реализации в I-м таксоне; 3-я реализация - типичная - в I-м таксоне; 4-я реализация типична во 2-м таксоне. Непосредственно за таблицей выведена строка количества элементов в таксонах по порядку следования: 9 - число элементов в I-м таксоне; I - число элементов во 2-м таксоне; 2 - число элементов в 3-м таксоне и т.д.

3. Печатается матрица с заполненными пробелами.

Л и т е р а т у р а

1. ЗАГОРУЙКО Н.Г. Общие свойства задач распознавания образов. -В кн.: Вычислительные системы. Вып. 44. Новосибирск, 1972, с. 3-12.

2. ЕЛКИНА В.Н., ЗАГОРУЙКО Н.Г. Об алфавите объектов распознавания. -В кн.: Вычислительные системы. Вып. 22. Новосибирск, 1966, с. 59-76.

3. ЗАГОРУЙКО Н.Г. Методы распознавания и их применение. М., Изд-во "Сов.Радио", 1972.

4. ЕЛКИНА В.Н., ЗАГОРУЙКО Н.Г. Количественные критерии качества таксономии и их использование в процессе принятия решений. -В кн.: Вычислительные системы. Вып. 36. Новосибирск, 1969, с. 29-46.

5. ЗАГОРУЙКО Н.Г. Классификация задач распознавания образов. -В кн.: Вычислительные системы. Вып. 22. Новосибирск, 1966, с. 3-19.

Поступила в ред.-изд.отд.

5 февраля 1974 года

Программа "ZET"

```

SUBROUTINE ZET 73(XY1,XY2,XY3,XYZ,LIS,LID,KK1,KK2,KK3,KK4,
1KK5,LIR,LIF,XY4,KK6,KKR)
DIMENSION XY1(KK1,KK2),XY2(7,KK3),XY3(KK1,KK3),XYZ(LIR),
1LIS(7,KK3),LID(3,KK3),LIF(2,KK6),XY4(KK6)
START=8.517
KARTA=0
820 CONTINUE
CALL MATRIX(KK1,KK2,KK3,0,XY1,XY2,LIS)
DO 99 I=1,KK1
PE1=0
PE2=8.517
DO 98 J=1,KK2
IF(XY1(I,J)-PE2) 97,98,98
PE2=XY1(I,J)
97 CONTINUE
DO 96 J=1,KK2
IF(XY1(I,J)-PE1) 96,96,95
IF(XY1(I,J)-START) 94,96,96
94 PE1=XY1(I,J)
96 CONTINUE
IF(PE1-PE2) 93,92,93
92 DO 91 J=1,KK2
IF(XY1(I,J)-START) 88,91,88
88 XY1(I,J)=1
91 CONTINUE
GO TO 90
93 DO 89 J=1,KK2
IF(XY1(I,J)-START) 87,89,89
87 XY1(I,J)=(XY1(I,J)-PE2)/(PE1-PE2)
89 CONTINUE
90 XY2(6,I)=PE1
XY2(7,I)=PE2
99 CONTINUE GO TO 800
1 KAP=0
DO 10 KT2=1,KK2
DO 10 KT1=1,KK1
IF(XY1(KT1,KT2)-START) 10,11,10
11 KA1=KT1
KC1=KT2
KB1=0
KB2=0
KAB1=0
KAB2=0
PE1=0
PE2=0
KAC=0
DO 9 J=1,KK2
IF(XY1(KA1,J)-START) 12,9,12
12 KB1=KB1+1
9 CONTINUE
IF(KB1-1) 79,75,13
13 DO 8 I=1,KK1
IF(XY1(I,KC1)-START) 14,8,14

```

```

14 KB2=KB2+1
8 CONTINUE
IF(KB2-1) 75,75,15
15 DO 7 KT3=1,4
KAC1=0
KAC2=0
KAC3=0
IF(KT3-1) 17,16,17
17 IF(KT3-2) 19,18,19
19 IF(KT3-3) 21,20,21
16 DO 22 I=1,KK1
IF(XY1(I,KC1)-START) 23,24,23
24 KAC1=KAC1+1
LIS(1,KAC1)=I
GO TO 22
23 KAC2=KAC2+1
LID(1,KAC2)=I
22 CONTINUE
DO 25 J=1,KK2
DO 26 I=1,KK1
IF(XY1(I,J)-START) 26,5,26
5 IF(I-KA1) 27,25,27
27 DO 28 KT4=1,KAC1
IF(I-LIS(1,KT4)) 28,26,28
28 CONTINUE
GO TO 25
26 CONTINUE
KAC3=KAC3+1
LID(2,KAC3)=J
25 CONTINUE
GO TO 50
18 DO 29 J=1,KK2
IF(XY1(KA1,J)-START) 80,81,80
81 KAC1=KAC1+1
LIS(1,KAC1)=J
GO TO 29
80 KAC3=KAC3+1
LID(2,KAC3)=J
29 CONTINUE
DO 30 I=1,KK1
DO 31 J=1,KK2
IF(XY1(I,J)-START) 31,82,31
82 IF(J-KC1) 83,30,83
83 DO 84 KT4=1,KAC1
IF(J-LIS(1,KT4)) 84,31,84
84 CONTINUE
GO TO 30
31 CONTINUE
KAC2=KAC2+1
LID(1,KAC2)=I
30 CONTINUE
50 IF(KAC2-1) 7,32,32
32 IF(KAC3-3) 7,33,33
33 KAB1=KAC2
KAB2=KAC3+1
KAB3=MAX0(KAB1,KAB2)

```

```

KT4=1
KAB2=KAB2+1
LID(2,KAB2)=KC1
DO 34 J=1,KAB2
KB1=LID(2,J)
DO 34 I=1,KAB1
KB1=LID(I,I)
XYZ(KT4)=XY1(KB1,KB2)
34 KT4=KT4+1
GO TO 60
20 DO 37 J=1,KK2
IF(XY1(KA1,J)-START) 38,39,38
39 KAC1=KAC1+1
LIS(1,KAC1)=J
GO TO 37
38 KAC3=KAC3+1
LID(2,KAC3)=J
37 CONTINUE
DO 40 I=1,KK1
DO 41 J=1,KK2
IF(XY1(I,J)-START) 41,35,41
35 IF(J-KC1) 42,40,42
42 DO 43 KT4=1,KAC1
IF(J-LIS(1,KT4)) 43,41,43
43 CONTINUE
GO TO 40
41 CONTINUE
KAC2=KAC2+1
LID(1,KAC2)=I
40 CONTINUE
GO TO 79
21 DO 44 I=1,KK1
IF(XY1(I,KC1)-START) 85,86,85
86 KAC1=KAC1+1
LIS(1,KAC1)=I
GO TO 44
85 KAC2=KAC2+1
LID(1,KAC2)=I
44 CONTINUE
DO 45 J=1,KK2
DO 46 I=1,KK1
IF(XY1(I,J)-START) 46,36,46
36 IF(I-KA1) 77,45,77
77 DO 78 KT4=1,KAC1
IF(I-LIS(1,KT4)) 78,46,78
78 CONTINUE
GO TO 45
46 CONTINUE
KAC3=KAC3+1
LID(2,KAC3)=J
45 CONTINUE
79 IF(KAC3-1) 7,47,47
47 IF(KAC2-3) 7,48,48
48 KAB1=KAC2+1
KAB2=KAC3
KAB3=MAX0(KAB1,KAB2)

```

```

LID(1,KAB1)=KA1
DO 49 I=1,KAB1
KB1=LID(1,I)
DO 51 J=1,KAB2
KB2=LID(2,J)
51 XY2(1,J)=XY1(KB1,KB2)
DO 52 KAC2=1,KAB1
KB1=LID(1,KAC2)
DO 53 KAC3=1,KAB2
KB2=LID(2,KAC3)
53 XY2(2,KAC3)=XY1(KB1,KB2)
CALL CORREL(KAB1,KAB2,KAB3,PZ,XY2)
XYZ(KT4)=PZ
52 KT4=KT4+1
49 CONTINUE
DO 54 J=1,KAB1
IF(XY3(J,KAB1)) 55,56,56
55 LID(3,J)=-1
GO TO 54
56 LID(3,J)=1
54 CONTINUE
KT4=KT4-1
DO 76 J=1,KT4
76 XYZ(J)=1-ABS(XYZ(J))
KAC1=KAB1
KAC2=KAB1
KAC3=3
KAC4=KAB1-1
KAC5=KK4
KAC6=KAB3
KAC7=KK5
KAC9=0
PA=0
CALL TAXON(KAC1,KAC2,KAC3,KAC4,KAC5,KAC6,KAC7,LIS,XY3,XY2,
1PA,KAC9)
KB3=IABS(LIS(4,KAB1))
IF(1-LIS(5,KB3)) 57,7,7
57 IF(PA-PE1) 7,7,58
58 KAC=KAC+1
PE1=PA
KT4=0
DO 59 I=1,KAB1
IF(KB3-IABS(LIS(4,I))) 59,61,59
61 KT4=KT4+1
LIS(1,KT4)=I
59 CONTINUE
KB1=KT4-1
DO 62 I=1,KB1
KATC=LIS(1,I)
XY2(5,I)=(1-XY3(KATC,KAB1))*LID(3,KATC)
XY2(4,I)=0
KAK=LID(1,KATC)
DO 63 J=1,KAB2
KB2=LID(2,J)
63 XY2(4,I)=XY2(4,I)+XY1(KAK,KB2)
62 XY2(4,I)=(XY2(4,I)+XY1(KAK,KC1))/(KAB2+1)

```

```

PE3=0
PE4=0
DO 64 I=1,KB1
KATC=LIS(1,I)
KAK=LID(1,KATC)
64 XY2(4,I)=XY1(KAK,KC1)+(XY2(4,I)-XY1(KAK,KC1))*(1-XY2(5,I))
DO 65 I=1,KB1
PE3=PE3+ABS(XY2(5,I))
65 PE4=PE4+XY2(4,I)*ABS(XY2(5,I))
IF (PE3) 66,67,66
66 PE3=PE4/PE3
GO TO 68
67 PE3=PE4
68 PE2=PE3
GO TO 7
60 KAC1=KAB1
KAC2=KAB2
KAC3=3
KAC4=KAB2-1
KAC5=KK4
KAC6=KAB3
821 CONTINUE
KAC7=KK5
KAC9=0
PA=0
CALL TAXON(KAC1,KAC2,KAC3,KAC4,KAC5,KAC6,KAC7,LIS,XY3,XY2,
1PA,KAC9)
KB3=IABS(LIS(4,KAB2))
IF(1-LIS(5,KB3)) 69,7,7
69 IF(PA-PE1) 7,7,70
70 KAC=KAC+1
PE1=PA
KT4=0
DO 71 J=1,KAB2
IF(KB3-IABS(LIS(4,J))) 71,72,71
72 KT4=KT4+1
LIS(1,KT4)=J
71 CONTINUE
KB1=KT4-1
PE3=0
DO 73 J=1,KB1
KATC=LIS(1,J)
KAK=LID(2,KATC)
73 PE3=PE3+XY1(KA1,KAK)
PE3=PE3/KB1
PE2=PE3
7 CONTINUE
IF(KAC) 74,75,74
74 XY4(KOCT)=PE2
IF(KOLIA) 206,206,506
506 XY1(KA1,KC1)=PE2
206 PE3=PE2*(XY2(6,KA1)-XY2(7,KA1))+XY2(7,KA1)
PRINT 611,KOCT,KA1,KC1,PE3
75 KOCT=KOCT+1
10 CONTINUE
IF(KOCT-KK6) 601,601,602
601 IF(KOLIA) 600,602,600

```

```

600 IF(KKLT-KK6) 505,602,602
602 KAB1=0
KOLIA=7
PRINT 610
DO 604 J=1, KK6
IF(XY4-START) 605,604,604
605 XY6=XY6+XY4(J)
KAC1=LIF(1,J)
KAC2= LIF(2,J)
XY1(KAC1,KAC2)=XY4(J)
KAB1=J
XY4(J)=XY4(J)*(XY2(6,KAC1)-XY2(7,KAC1))+XY2(7,KAC1)
PE3=XY4(J)
PRINT 611, KAB1,KAC1,KAC2,PE3
604 CONTINUE
GO TO 810
800 XY5=0
KAP=1
KOCT=1
XY6=0
KOLIA=0
DO 502 J=1, KK2
DO 502 I=1, KK1
IF(XY1(I,J)-START) 502,503,502
503 LIF(1,KAP)=I
LIF(2,KAP)=J
KAP=KAP+1
502 CONTINUE
DO 501 J=1, KK6
501 XY4(J)=START
GO TO 1
500 KOCT=1
XY6=0
KELT=0
505 KELT=KELT+1
I=LIF(1,KELT)
J=LIF(2,KELT)
XY1(I,J)=START
GO TO 1
810 IF(XY6-XY5) 606,608,606
606 KARTA=KARTA+1
XY5=XY6
IF(KARTA-KK8) 500,500,608
610 FORMAT(20X, HOMEP, 10X, HOMEP CTOMENIA, 10X, HOMEP
1CTPOKH, 10X, BOCTAH BEMEHT /)
611 FORMAT(21X, I4, 19X, I4, 18X, I4, 15X, E10.3)
608 DO 202 I=1, KK1
PE3=XY2(6,I)-XY2(7,I)
DO 202 J=1, KK2
IF(XY1(I,J)-START) 204,202,202
204 XY(I,J)=XY1(I,J)*PE3+XY2(7,I)
202 CONTINUE
CALL MATRIX(KK1, KK2, 1, XY1, XY2, LIS)
RETURN
END

```

```

SUBROUTINE RAND(TK, TL, TM, JN)
I=JN*5**5
K=2**16
P=1
JN=I-INT(P/K)*K
Q=JN
TM=(Q/K)*(TL-TK)+TK
P=JN+1
RETURN
END

```

```

SUBROUTINE CORREL(L1, L2, L3, X, X2)
DIMENSION X2(7, L3)
VA=0
VB=0
DO 1 I5=1, L2
VA=VA+X2(1, I5)
VB=VB+X2(2, I5)
1 CONTINUE
VA=VA/L2
VB=VB/L2
VD=0
VC=0
X=0
DO 2 I5=1, L2
VD=VD+(VA-X2(1, I5))**2
VC=VC+(VB-X2(2, I5))**2
2 CONTINUE
VD=VD/(L2-1)
VC=VC/(L2-1)
VD=SQRT(VD)
VC=SQRT(VC)
VB=0
DO 3 I5=1, L2
3 VB=VB+(VA-X2(1, I5))*(VB-X2(2, I5))
X=(VD*VC*(L2-1))
IF(X) 4, 5, 4
4 X=VB/X
5 RETURN
END

```

```

SUBROUTINE MATRIX(L1, L2, L3, L4, X1, X2, L5)
DIMENSION X1(L1, L2), X2(7, L3), L5(7, L3)
IF(L4) 1, 2, 1
2 PRINT 10
GO TO 3
1 PRINT 11
3 I6=1
I7=10
I8=I1/10
I9=I8*10
I11=I1-I9
DO 5 J=1, L1

```

```

5  L5(1,J)=J
   IF(L11) 4,4,6
6  IF(L8) 7,7,4
4  DO 8 J=1,L8
   PRINT 12,(L5(1,I),I=L6,L7)
   DO 9 L12=1,L2
   PRINT 13,L12,(X1(I,L12),I=L6,L7),L12
9  CONTINUE
   L6=L7+1
8  L7=L7+10
   IF(L11) 20,20,7
7  L7=L7-9
   PRINT 12,(L5(1,I),I=L7,L1)
   DO 19 J=1,L2
   PRINT 13,J,(X1(I,J),I=L7,L1),J
19 CONTINUE
20 PRINT 14
   RETURN
10 FORMAT(50X,' МАТРИЦА ИСХОДНЫХ ДАННЫХ'///)
11 FORMAT(50X,' МАТРИЦА ИСПРАВЛЕННЫХ ДАННЫХ'///)
12 FORMAT(//10X,10I10/10X'.....
1.....')
13 FORMAT(2X,I4,2X,2H*,10E10.3,2H*,2X,I4)
14 FORMAT(10X,.....'/10
1X,10I10///)
   END

```

```

SUBROUTINE ROOT1(KF,KQ,PE,PF,KD,KBT)
DIMENSION PE(KF,KQ),PF(7,KBT)
DO 133 JTA=1,KQ
  PE(4,JTA)=0
DO 133 JTB=1,KP
  PF(4,JTA)=PF(4,JTA)+(PF(KD,JTB)-PE(JTB,JTA))*2
133 CONTINUE
DO 134 JTB=1,KQ
  PF(4,JTB)=SQRT(PF(4,JTB))
134 RETURN
END

```

```

SUBROUTINE FEMIR(J1,J2,J3,J4,J5,LA,Y1,Y2,CR)
DIMENSION Y1(J1,J2),Y2(7,J4),LA(7,J4)
JY=1
JW=1
JA=LA(5,1)
JB=1
JC=0
12 IF(1-IABS(LA(4,JB))) 10,11,10
11 JC=JC+1
   LA(6,JC)=JB
10 JB=JB+1
   IF(JB-J2) 12,12,13
13 JB=0
   DO 14 J=1,JA
     JC=LA(6,J)

```

```

DO 15 I=1,J1
Y2(1,I)=Y1(I,JC)
IF(JB) 16,17,16
17 CALL ROOT1(J1,J2,Y1,Y2,1,J4)
DO 18 I=1,J2
  LA(1,I)=JC
18 Y2(2,I)=Y2(4,I)
  JB=1
  GO TO 14
16 CALL ROOT1(J1,J2,Y1,Y2,1,J4)
DO 19 I=1,J2
  IF(Y2(4,I)-Y2(2,I)) 20,19,19
20 Y2(2,I)=Y2(4,I)
  LA(1,I)=JC
19 CONTINUE
14 CONTINUE
DO 21 J=1,JA
  JB=LA(6,J)
21 Y2(2,JB)=8.E17
22 J=1
  PR=8.E16
23 IF(Y2(2,J)-PR) 24,25,25
24 PR=Y2(2,J)
  JS=J
  JP=LA(1,J)
25 J=J+1
  IF(J-J2) 23,23,26
26 JB=IABS(LA(4,JS))
  JA=LA(5,JB)
  JC=0
  JD=1
42 IF(JB-IABS(LA(4,JD))) 40,41,40
41 JC=JC+1
  LA(6,JC)=JD
40 JD=JD+1
  IF(JD-J2) 42,42,43
43 JD=0
  DO 27 J=1,JA
    JC=LA(6,J)
    DO 28 I=1,J1
      Y2(1,I)=Y1(I,JC)
      IF(JD) 29,30,29
30 CALL ROOT1(J1,J2,Y1,Y2,1,J4)
      DO 31 I=1,J2
        Y2(5,I)=Y2(4,I)
        LA(7,I)=JC
31 CONTINUE
      JD=1
      GO TO 27
29 CALL ROOT1(J1,J2,Y1,Y2,1,J4)
DO 32 I=1,J2
  IF(Y2(5,I)-Y2(4,I)) 32,32,33
33 Y2(5,I)=Y2(4,I)
  LA(7,I)=JC
32 CONTINUE
27 CONTINUE
  LA(2,JY)=JP
  LA(3,JY)=JS

```

```

Y2(3,JY)=PR
JY=JY+1
DO 34 J=1,JA
JD=LA(6,J)
34 Y2(2,JD)=8.E17
J=1
35 IF(Y2(2,J)-Y2(5,J)) 36,36,37
37 IF(Y2(2,J)-8.E17) 38,36,36
38 Y2(2,J)=Y2(5,J)
LA(1,J)=LA(7,J)
36 J=J+1
IF(J-J2) 35,35,39
39 JW=JW+1
IF(JW-J3) 22,44,44
44 IF(J5) 45,46,45
46 PRINT 1
JA=J3-1
PRINT 2,(LA(2,I), I=1,JA)
PRINT 2,(LA(3,I),I=1,JA)
PRINT 3,(Y2(3,I),I=1,JA)
45 DO 50 IKL=1,J3
JC=1
JD=0
JB=0
JW=1
JOK=JY
49 IF(IKL-IABS(LA(4,JC))) 54,48,54
48 JD=JD+1
LA(6,JD)=JC
GO TO 47
54 JB=JB+1
LA(7,JB)=JC
47 JC=JC+1
IF(JC-J2) 49,49,51
51 JA=LA(5,IKL)
JC=LA(6,1)
IF(JA-1) 57,57,56
56 DO 52 I=1,J1
52 Y2(1,I)=Y1(I,JC)
CALL ROOT1(J1,J2,Y1,Y2,1,J4)
DO 53 I=1,J2
Y2(2,I)=Y2(4,I)
53 LA(1,I)=JC
J2(2,JC)=8.E17
DO 55 I=1,JB
JD=LA(7,I)
55 Y2(2,JD)=8.E17
62 J=1
PR=8.E16
63 IF(Y2(2,J)-PR) 64,65,65
64 PR=Y2(2,J)
JS=J
JP=LA(1,J)
65 J=J+1
IF(J-J2) 63,63,66
66 DO 67 I=1,J1
67 Y2(1,I)=Y1(I,JS)

```

```

CALL ROOT1(J1,J2,Y1,Y2,1,J4)
LA(2,JY)=JP
LA(3,JY)=JS
Y2(3,JY)=PR
JY=JY+1
Y2(2,JS)=8.E17
J=1
69 IF(Y2(2,J)-Y2(4,J)) 72,72,70
70 IF(Y2(2,J)-8.E17) 71,72,72
71 Y2(2,J)=Y2(4,J)
LA(1,J)=JS
72 J=J+1
IF(J-J2) 69,69,73
73 JW=JW+1
IF(JW-JA) 62,74,74
74 IF(J5) 50,76,50
76 J=JY-1
PRINT 5,IKL
PRINT 2,(LA(2,I),I=JOK,J)
PRINT 2,(LA(3,I),I=JOK,J)
PRINT 3,(Y2(3,I),I=JOK,J)
GO TO 50
57 IF(J5) 50,58,50
58 PRINT 4,IKL
50 CONTINUE
JA=J2-1
DO 80 J=1,JA
Y4=8.E18
JB=LA(2,J)
JC=LA(3,J)
Y3=Y2(3,J)
LA(2,J)=LA(2,1)
LA(3,J)=LA(3,1)
Y2(3,J)=Y2(3,1)
JD=JA-1
JS=JB
DO 81 II=1,2
DO 82 IK=2,3
DO 82 I=1,JD
IF(JS-LA(IK,I+1)) 82,83,82
83 IF(Y4-Y2(3,I+1)) 82,82,84
84 Y4=Y2(3,I+1)
82 CONTINUE
81 JS=JC
Y2(3,J)=Y3
LA(2,J)=JB
LA(3,J)=JC
IF(Y3) 78,79,78
79 Y2(1,J)=1
GO TO 80
78 Y2(1,J)=Y4/Y3
80 CONTINUE
JA=J3-1
Y3=0
Y5=0
DO 85 J=1,JA
Y3=Y3+Y2(3,J)

```

```

85 Y5=Y5+Y2(1,J)
CONTINUE
Y3=Y3/JA
Y5=Y5/JA
Y4=0
JA=J3
DO 86 J=1,J3
JB=LA(5,J)
PR=0
IF(JB-1) 86,86,87
87 JC=JA+JB-2
DO 88 I=JA,JC
88 PR=PR+Y2(3,I)
Y4=Y4+PR/(JB-1)
JA=JA+JB-1
86 CONTINUE
Y4=Y4/J3
CR=Y3/((1+Y4)*(1+Y5))
IF(J5) 90,91,90
91 PRINT 6,CR
90 RETURN
1 FORMAT(40X,'КРАТЧАЙШАЯ ЦЕПЬ МЕЖДУ ОБРАЗАМИ',5X,'ПЕРВЫЙ
МАССИВ,НОМЕРА НАЧАЛЬНЫХ ВЕРШИН',5X,'ВТОРОЙ МАССИВ,НОМЕРА
2'КОНЕЧНЫХ ВЕРШИН',5X,'ТРЕТИЙ ДЛИНА РЕБЕР')
2 FORMAT(2X,14I8)
3 FORMAT(2X,14E8.2)
4 FORMAT(30X,8H0БРАЗ N=,I4,2X,'СОДЕРЖИТ ОДНУ РЕАЛИЗАЦИЮ)
5 FORMAT(60X,8H0БРАЗ N=,I4)
6 FORMAT(40X,'КРИТЕРИЙ КАЧЕСТВА ТАКСОНОМИИ ',2X,2HF=,E10.3)
END

```

```

SUBROUTINE MID(KT,KU,PC,PD,KP)
DIMENSION PC(KT,KU),PD(7,KP)
DO 131 JTA=1,KT
PD(1,JTA)=0
DO 131 JTB=1,KU
PD(1,JTA)=PD(1,JTA)+PC(JTA,JTB)
131 CONTINUE
DO 132 JTB=1,KT
PD(1,JTB)=PD(1,JTB)/KU
132 RETURN
END

```

```

SUBROUTINE TAXON(L1,L2,L3,I4,I5,I6,I7,I8,Z1,Z2,Z3,I9)
DIMENSION Z1(L1,I2),Z2(7,I6),I8(7,I6)
IB=0
IC=0
ID=0
II=1
IY=1
IN=1
RA=0
CALL MID(L1,I2,Z1,Z2,I6)
CALL ROOT1(L1,I2,Z1,Z2,1,I6)
RY=0
DO 100 I=1,I2

```

```

IF(Z2(4,I)-MY) 100,100,101
101 RY=Z2(4,I)
100 CONTINUE
IX=L2
RR=RY/I5
RU=RY-RR
DO 125 J=1,I2
125 I8(1,J)=J
GO TO 703
102 RU=RU-RR
DO 50 J=1,I2
50 I8(1,J)=J
IF(RU) 103,103,703
703 IF(I7) 104,701,104
701 PRINT 126,RU,RR
104 I=1
DO 105 J=1,I1
105 Z2(2,J)=Z1(J,I)
106 CALL ROOT1(L1,IX,Z1,Z2,2,I6)
IK=1
I=0
IT=1
I=I+1
107 IF(I-IX) 110,110,111
110 IF(Z2(4,I)-RU) 112,112,113
112 DO 114 J=1,I1
Z2(3,J)=Z1(J,IK)
Z1(J,IK)=Z1(J,I)
Z1(J,I)=Z2(3,J)
114 CONTINUE
IB=I8(1,IK)
I8(1,IK)=I8(1,I)
I3(1,I)=IP
IK=IK+1
GO TO 107
113 IT=IT+1
GO TO 107
111 IK=IK-1
CALL MID(L1,IK,Z1,Z2,I6)
J=0
J=J+1
115 IF(J-I1) 116,116,118
116 IF(Z2(1,J)-Z2(2,J)) 108,115,108
108 DO 109 J=1,I1
109 Z2(2,J)=Z2(1,J)
GO TO 106
118 IE=IE+IK
IY=IY+1
IT=IT-1
CALL ROOT1(L1,IK,Z1,Z2,1,I6)
RISH=8.E17
DO 704 J=1,IK
705 RISH=Z2(4,J)-RISH) 705,704,704
KLIM=I8(1,J)
704 CONTINUE

```

```

DO 124 J=1,IK
IZ=I8(1,J)
I8(4,IZ)=IN
124 CONTINUE
I8(5,IN)=IK
I8(4,KLIM)=-I8(4,KLIM)
IN=IN+1
IF(IT) 120,120,123
123 DO 119 I=1,IT
DO 122 J=1,I1
Z2(3,J)=Z1(J,I)
Z1(J,I)=Z1(J,IY)
Z1(J,IY)=Z2(3,J)
122 CONTINUE
IP=I8(1,I)
I8(1,I)=I8(1,IY)
I8(1,IY)=IP
119 IY=IY-1
IY=IT
IF(I8-I2) 104,120,104
120 IY=I2
IM=IN-1
IN=1
KA=1
586 DO 585 I=KA,I2
IF(I8(1,I)-KA) 585,584,585
584 IOC=I8(1,I)
I8(1,I)=I8(1,KA)
I8(1,KA)=IOC
DO 590 J=1,I1
Z2(3,J)=Z1(J,I)
Z1(J,I)=Z1(J,KA)
Z1(J,KA)=Z2(3,J)
590 CONTINUE
GO TO 591
585 CONTINUE
591 KA=KA+1
IF(KA-I2) 586,586,587
587 RD=0
IF(IM-I3) 605,601,601
601 IF(IM-I4) 600,600,605
600 CALL TSMIR(I1,I2,IM,I6,I7,I8,Z1,Z2,RD)
IF(RD-RA) 605,605,606
606 RA=RD
RC=RU
IC=IM
605 IE=0
IF(ID) 609,607,609
607 IF(IM-I4) 611,610,610
611 IF(I7) 608,401,608
401 PRINT 406
PRINT 404,(I8(4,J),J=1,I2)
PRINT 405
PRINT 404,(I8(5,J),J=1,IM)
PRINT 407

```

```

608 GO TO 102
610 ID=1
RU=RC+RR
GO TO 102
609 PRINT 403,RC,RR,IM,RA
PRINT 406
PRINT 404,(I8(4,J),J=1,I2)
PRINT 405
PRINT 404,(I8(5,J),J=1,IM)
PRINT 405
PRINT 408
IY=I2-1
PRINT 409,(I8(2,J),J=1,IY)
PRINT 409,(I8(3,J),J=1,IY)
PRINT 410,(Z2(3,J),J=1,IY)
L9=IC
Z3=RA
103 RETURN
126 FORMAT(10X,9HPAИYC R=,E10.3, 5X,'ДЕЛТА R=,E10.3)
406 FORMAT(2X,117H1xxx2xxx3xxx4xxx5xxx6xxx7xxx8xxx9xxx0xxx1
1xxx2xxx3xxx4xxx5xxx6xxx7xxx8xxx9xxx0xxx1xxx2xxx3xxx4xxx
25xxx6xxx7xxx8xxx9xxx0)
404 FORMAT(30I4)
405 FORMAT(2X,117H+++++-----
1-----
2-----)
407 FORMAT(40X,32 НКОНЕЦ ОЧЕРЕДНОГО ШАГА ТАКСОНОМИИ/)
403 FORMAT(///3X,ЛУЧШЕЕ РАЗБИЕНИЕ ,3X,9 ПРАДИУС R=,E10.3,9H
1ДЕЛТА R=,E10.3, 3X,ЧИСЛО ТАКСОНОВ N= ,I4,3X,КРИТЕРИЙ F
2= ,E10.3/)
408 FORMAT(30X,'КРАТЧАЙШАЯ СЕТЬ,НОМЕРА СВЯЗАННЫХ ВЕРШИН,
1ДЛИНА РЕБЕР ')
410 FORMAT(2X,14E8.2)
409 FORMAT(2X,14I8)
END

```