Ep.6.7
482

ISSN 0136-3549
0320-3409

TALLINNA
POLÜTEHNILISE INSTITUUDI
TOIMETISED
482

ТРУДЫ ТАЛЛИНСКОГО
ПОЛИТЕХНИЧЕСКОГО
ИНСТИТУТА

TPI '80

ПОСТРОЕНИЕ ТРАНСЛЯТОРОВ, ОБРАБОТКА ДАННЫХ, ВОПРОСЫ ПРОГРАММИРОВАНИЯ

Ep.6.7

482

TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

TPI '80

УДК 681.3.06

ПОСТРОЕНИЕ ТРАНСЛЯТОРОВ, ОБРАБОТКА ДАННЫХ, ВОПРОСЫ ПРОГРАММИРОВАНИЯ

Труды экономического факультета XL

Таллин – 1980

ТАЛЛИНСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ
Труды ТПИ № 482
ПОСТРОЕНИЕ ТРАНСЛЯТОРОВ, ОБРАБОТКА ДАННЫХ,
ВОПРОСЫ ПРОГРАММИРОВАНИЯ
Труды экономического факультета XL
Редактор И. Амитан. Техн. редактор В. Ранник
Сборник утвержден коллегией Трудов ТПИ 2 окт. 1979 г.
Подписано к печати 18 авг. 1980 г. Бумага 60x90/16
Печ. л. 9,75+0,5 приложение. Уч.-изд. л. 8,8
Тираж 300. МВ-07907. Ротапринт ТПИ
Таллин, ул. Коскла, 2/9. Зак.№ 447
Цена 1 руб. 30 коп.

© Таллинский политехнический институт, 1980

№ 482

TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 681.3.01

А.О. Вооглайд, Д.Б. Лийб

ДРЕВОВИДНАЯ ПЕРЕМЕННАЯ В ОБРАБОТКЕ ДАННЫХ

Введение

В последнее время все больше внимания уделяется использованию идеологии структурного программирования в реальных системах программирования.

В структурном программировании различают два принципиально разных пути рассмотрения структур. Условно их можно называть последовательным и библиотечным образом мышления [3].

Библиотечный образ мышления с точки зрения программирования наиболее общий способ рассмотрения структур и он дает достаточно гибкие методы для проектирования, реализации и вида изменения всей системы.

Такие методы в программировании требуют более мощных средств по сравнению с классическими языками программирования.

Одним из таких средств в данной статье предлагается древовидная переменная. Саму идею о древовидной переменной подали молодые ученые из ФРГ [1]. Они предложили при программировании употреблять синтексически-ориентированные методы в языках высокого уровня. Здесь предлагается использовать нормальную форму Бэкуса-Наура с расширениями Лукаса (БНФ) не только для определения языков программирования, но и включить их в язык с тем, чтобы дать программистам возможность динамично использовать синтаксически-ориентированные преобразования.

Опишем неформально некоторые области использования древовидной переменной.

Пример I. Древовидную переменную можно использовать при создании универсального генератора ввода/вывода управляемого данными. Такой генератор во время сканирования проводит грубый анализ данных [7], в ходе последнего определяют тип структуры данных. Используя результаты грубого анализа, далее генерируют динамический БНФ [2], которые определяют соответствующую контекстно-свободную грамматику (КСГ). В соответствии с описанием БНФ и генерируется анализатор. Учитывая, что сама система позволяет динамически менять анализаторы во время анализа исходного текста, можно создать такой класс КСГ, который описывает структуру возможных данных, причем данные сами определяют, на каком этапе работы системы и на каком языке их будут обрабатывать.

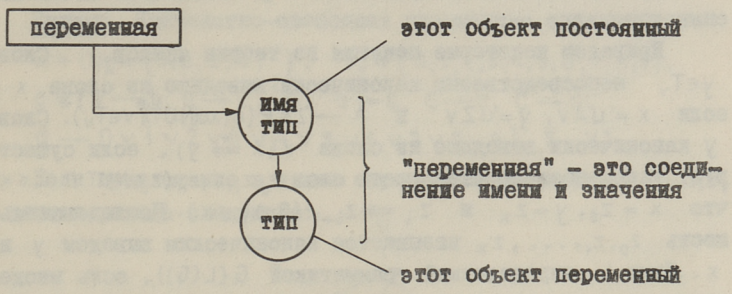
Пример 2. Древовидную переменную можно использовать при организации управления сложными структурами типа прикладных программ (ППП). При организации управления ППП возможны два пути. Во-первых, для каждого класса заданий на каком-то классическом языке программирования всегда пишется новая главная программа. Это типичное последовательное мышление. Во-вторых, для каждого класса заданий описывается его макет, и система сама выбирает алгоритм решения, если это возможно [4]. Это типичное библиотечное мышление.

На практике часто возникает необходимость для каждого класса заданий описывать макет и главную программу. Так, неразумно каждый раз для всего класса задач автоматически находить те части алгоритма решения, которые нам известны заранее.

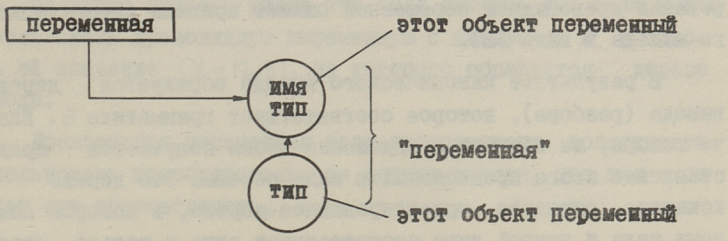
Используя древовидную переменную можно управление описывать с помощью БНФ, и в случае надобности в процессе вычисления создавать новые древовидные переменные. Это значит, что мы определяем управление конечным автоматом. Управление в машинной памяти представляется деревом с помеченными вершинами. В каждой вершине дерева может быть выполняемый модуль и математическое условие или адрес поля декларации новой древовидной переменной. Декларация постоянно выполняется во время трансляции. Другими словами, данные и программы сами генерируют себе пути прохода, по которым они реализуют нужный алгоритм.

Древовидная переменная

В качестве переменной неформально можно рассматривать пару объектов, состоящую из экземпляра значения и именующего его имени:



Древовидная переменная тоже состоит из двух объектов, но отличие в том, что при декларировании классической переменной ее тип определяет тип значения, а в случае декларирования древовидной переменной, наоборот, значение определяет ее тип:



Для описания древовидной переменной применяются правила в виде БНФ. БНФ использовалась для описания АЛГОЛ-60 и АЛГОЛ-68 [8]. Точнее говоря, эти правила задаются как грамматика в каком-то классе формальных языков. Формальный язык состоит из бесконечного множества предложений или слов, которые можно построить из алфавита терминальных символов, и язык может быть определен конечным путем так называемой порождающей грамматикой. Далее, при определении древовидной переменной будем пользоваться контекстно-свободной грамматикой. Определим ее следующим образом:

КСГ – это упорядоченная четверка $G=(N_N,T_V,P,S)$, где

I) N_N – конечное множество нетерминальных символов или нетерминалов;

2) T_v – не пересекающееся с N_N конечное множество нетерминальных символов или T_v терминалов;

3) P – конечное множество правил в виде БНФ;

4) S – выделенный символ из N_N, называемый начальным символом.

Приведем некоторые понятия из теории языков. Слово $y \in T_v$ непосредственно канонически выводимо из слова x, если $x = uAv$, $y = uZv$ и $A \to Z \in P (u, x \in (NUT)^*, v \in T_v)^*$. Слово y канонически выводимо из слова $x (x \overset{*}{\Rightarrow} y)$, если существует такая последовательность слов $z_0, z_1, \ldots, z_K (z_i \in NUT\ 0 \leqslant i \leqslant K)$, что $x = z_0$, $y = z_K$ и $z_i \Rightarrow z_{i+1}$ $(0 \leqslant i \leqslant K)$. Последовательность $z_0, z_1, \ldots, z_K$ называется каноническим выводом y из x. Язык L, определяемый грамматикой $G(L(G))$, есть множество слов, выводимых из S и состоит только из терминалов $L(G) = \{x : S \overset{*}{\Rightarrow} x;\ x \in T_v^*\}$.

Канонический вывод $z_0, z_1, \ldots, z_K$ можно представить в виде $p_1, p_2, \ldots, p_K$, где $p_i \in P$ – правило, при помощи которого z_i непосредственно выводимо из z_{i-1}. При декларировании древовидной переменной опишем правила канонического вывода в виде БНФ.

В результате канонического вывода образуется дерево вывода (разбора), которое соответствует грамматике G. Иначе говоря, из каждого предложения языка получается представление этого предложения в виде дерева. Это дерево – конечное, корневое, ориентированное дерево, в котором каждому узлу и каждой дуге соответствует одна и только одна метка, причем

1) метка корня есть нетерминал и обязательно начальный символ S грамматики G;

2) узлы, не являющиеся листьями, помечены нетерминалами N_N грамматики G;

3) узлы, являющиеся листьями, помечены терминалами T_v грамматики G.

Как уже выше было сказано, переменная состоит из значения, ассоциированного с именем, которым он именуется. Для определения древовидной переменной нам надо уточнить, что является его значением и что его именем.

Древовидная переменная определена конкретной контекстно-свободной грамматикой G и ее именем служит начальный символ S грамматики G , а значением - цепочка, которая создается конкатенацией слева направо листьев дерева вывода.

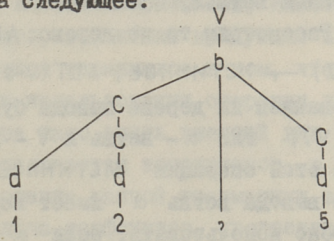
Пример. Контекстно-свободная грамматика следующая:

$G = (N_N = \{b, c, d, V', ''\}, \quad T_V = \{0,1,2,3,4,5,6,7,8,9\},$

$p = \{V \to b,\ b \to C',''C,\ \ b \to C,\ C \to d,\ C \to dC,$

$d \to 0 \vee 1 \vee 2 \vee 3 \vee 4 \vee 5 \vee 6 \vee 7 \vee 8 \vee 9\},\ S = V).$

Если предложение в языке определено грамматикой G-12,5, то дерево вывода следующее:



Из этого примера видно, что если описать КСГ G, то можно использовать древовидную переменную с именем V и присвоить ей значение (V=12,5), из которого образуется дерево вывода.

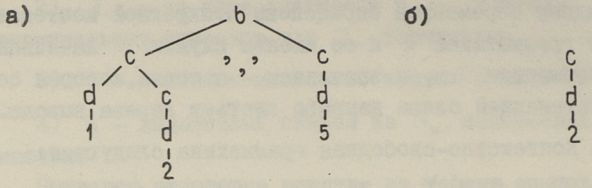
Древовидная переменная дает программисту возможность использовать преобразования на дереве вывода, так как он может его просматривать.

Для рассмотрения преобразований на дереве вывода надо декларировать понятие ветви.

Пусть K произвольная внутренняя вершина на дереве B с корнем V.

Если мы проходим по дереву вывода последовательно с корня V и запоминаем все вершины дерева до вершины K , тогда мы получаем ветвь с дерева B. Такая ветвь определяет одно поддерево из дерева B , которое, в свою очередь, тоже дерево с корнем K. Например, в предыдущем примере ветви могут быть:

а) V·b, б) V·b·c·c. Эти ветви определяют поддеревья:



Рассмотрим преобразования ALT, IN, IS на дереве вывода. Преобразование на дереве вывода ALT(α,y), где α – ветвь и y – цепочка. Эта логическая операция на дереве вывода получает значение ИСТИННОЕ тогда, когда под корнем, определенным ветвью, конкатенация вершин слева направо дает y. Рассмотрим то же дерево: ALT(V,b)→ИСТИННОЕ, ALT(V·b·c·c·d,2) → ИСТИННОЕ, ALT(V·b·c·d,1) → ЛОЖНОЕ. Вторым преобразованием на дереве вывода будет логическая операция IN(α,V), где α – ветвь и V – вершина дерева вывода. Значение этой операции ИСТИННОЕ, если отображенная на дереве вывода ветвь α имеет корень с именем V. Эту операцию удобно использовать, если α дается с формальными параметрами.

Значение логической операции IS(α,V) будет ИСТИННОЕ тогда, когда для цепочки α найдется дерево вывода с корнем V.

В следующем подпункте статьи рассмотрим подробнее сущность операции и область их использования на дереве вывода, а также реализацию древовидной переменной.

Реализация

В вычислительном центре Таллинского политехнического института в настоящее время разрабатывается новая версия системы построения трансляторов (СПТ) для ЭВМ ЕС с использованием операционной системы ОС.

СПТ дает средства для составления трансляторов [9]. Данная система позволяет генерировать транслятор с синтаксически управляемой схемой трансляций. Эта система использована для реализации древовидной переменной. Работа с древовидной переменной производится в двух частях.

Сперва декларируется древовидная переменная. Для этого грамматика переменной пишется на метаязыке [5]. Это зна-

чит, что определяют терминалы языка и правила вывода в виде $\alpha \rightarrow \beta$. При рекурсии можно использовать звездочку Клини.

Написанная декларация древовидной переменной будет входным текстом для анализатора, который осуществляет лексический и синтаксический анализы. Выводом анализатора служит дерево разбора. Из дерева разбора используя программы семантической части метатранслятора составляются таблицы грамматики. Работу программ облегчают вспомогательные структуры на дереве разбора. Выводом метатранслятора становятся таблицы терминалов, классы разделителей, семантики терминалов, семантики продукций и сами продукции. Особенность таблицы терминалов в том, что она содержит и нетерминалы. При использовании операции для преобразований на дереве разбора надо знать внешний вид нетерминалов. Далее в работу подключается конструктор, в задачу которых входит преобразование данной грамматики в грамматику предшествования, выдача конфликтов предшествования и проверка, является ли полученная грамматика предшествования редуцируемой $C(1,1)$ – ограниченным каноническим контекстом. Выводом конструктора являются дополнительно к вышеназванным таблицам еще и таблицы левой и правой частей продукций, матрица предшествования.

После такой работы транслятора можно программировать программы, используя уже декларированную древовидную переменную.

Приведем содержательный пример: пусть генератору ввода данных задано читать команду языка в виде V=READ( ), где V – древовидная переменная. По этой команде читается следующая константа и ее значение присваивается древовидной переменной V.

В нашей системе с такой командой обращаются к анализатору. Анализатор производит разбор дерева и переменная V получает значение – относительный адрес этой вершины дерева разбора, в котором расположен корень.

Зная относительный адрес корня, нетерминалы и правила разбора дерева для данной древовидной переменной V, можно в программах пользователя производить операции на дереве разбора.

В данной системе реализованы две операции – IN и ALT. Эти операции реализованы на языке программирования ПЛ/I в виде процедур функций:

IN(U,α), ALT(U,α),

где U – адрес текущей вершины дерева разбора;

α – ветвь в виде цепочки, $a_1, a_2, \ldots, a_n$

где $a_1, a_2, \ldots, a_n$ имена (нетерминалов) вершин, для разделения которых ставятся между ними точки.

Ветвь можно представить в виде константы или декларировать цепочки с переменной длиной.

Например: DCL A CHAR(100)VARYING;

Для реализации этих операций еще необходимы программы для определения значения ветви, для присваивания переменной значения поддерева, для определения значения цепочки и/или константы и программа для определения нетерминала из описания ветви. Программа для определения ветви работает как функция. Значением этой программы является адрес вершины дерева разбора, определенной описанием ветви. Обращаться к программе можно следующим образом:

HARU (U,'$a_1 \cdot a_2 \cdot \ldots \cdot a_n$').

Программа определения ветви обращается к программе

ALTERN(α,NM),

которая разделяет точками одиночные имена в описании ветви, где

$NM - a_1 \vee a_2 \vee \ldots \vee a_n$.

При присваивании переменной значения поддерева используется программа-функция:

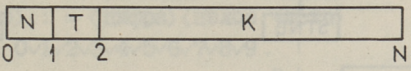
OMIST(U,H), где H=HARU(U,α).

Эта программа просматривает определенное ветвью поддерево.

Для определения терминала в вершине дерева указан индекс ХЕШ – таблицы, где он находится. Для определения значения константы или цепочки в вершине дерева указан его

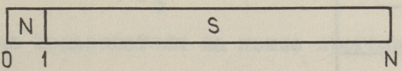
относительный адрес. Константы и цепочки располагаются в отведенной для дерева области памяти.

Структура константы в памяти следующая:



где K – константа в виде цепочки;
T – тип константы (с фиксированной длиной,
вещественная,
двоичная,
десятеричная);
N – длина константы в байтах.

Структура цепочки:



S – цепочка;
N – длина цепочки.

Для присваивания переменной значения цепочки или константы используется программа

STNG(TT, STRING, LL, TR),

где TT – константа, имеющая значение 0 или I.

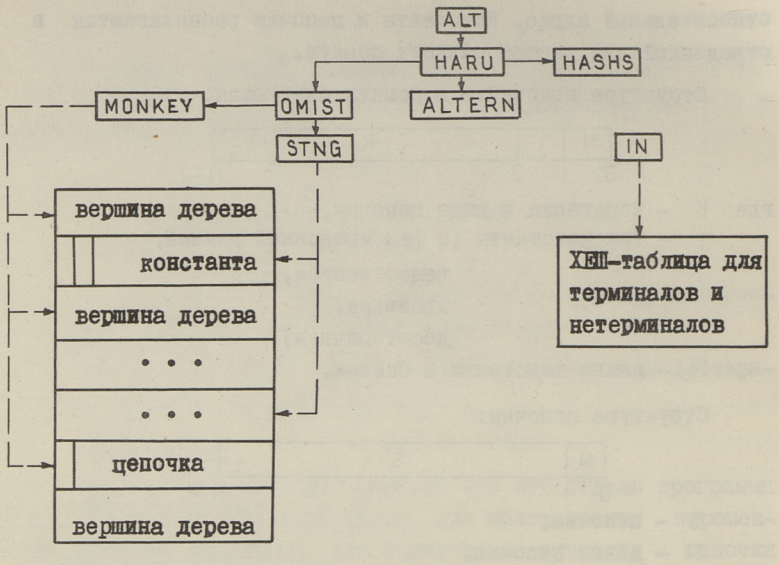
Если значение 0, тогда в листе дерева есть ссылка на константу, в противном случае – на цепочку.

STRING – переменная, которой присваивается это значение;
LL – номер вершины дерева;
TR – начальный адрес дерева.

Блок-схема, характеризующая управление данными и использование программ, приведена на рисунке на странице I2.

Помимо вышеописанных программ на схеме имеются еще две программы.

Программа MONKEY при обращении к дереву разбора выдает значение следующей за текущей вершины. Программа HASHS выдает индекс ХЕШ-таблицы нетерминала, который определен ветвью. Программы MONKEY, HASHS, STNG, ALTERN написаны на АССЕМБЛЕРЕ.



В заключение приведем одну задачу, которую можно легко решить, используя древовидную переменную. Задача генератора ввода данных вводить вещественные и целые числа. Эти числа разделены между собой пробелами и находятся в случайном порядке. Нам надо определить сумму целых чисел и сумму вещественных.

Древовидная переменная для определения чисел описывается на метаязыке по следующим правилам:

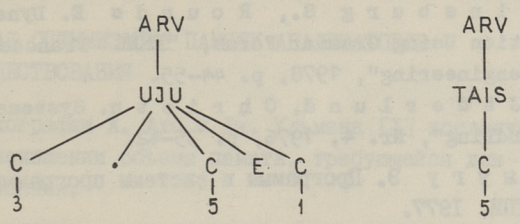
```
RULES
R1 ARV ⇒ TAIS
R2 ARV ⇒ UJU
R3 UJU ⇒ C·CEC
R4 TAIS ⇒ C
TERMINALS
C
·
E
ARV
TAIS
UJU
```

Данные можно описать по следующему:

⟨вещественное число⟩:: = ⟨целое число⟩·⟨целое число⟩ E⟨
⟨целое число⟩

⟨целое число⟩:: = ⟨цифра⟩⟨целое число⟩
⟨цифра⟩:: = 0∨1∨2∨3∨4∨5∨6∨7∨8∨9 .

Деревья для чисел 35 ($3{,}5\cdot10^{1}$) и 5 следующие:



Программа пользователя на языке программирования следующая:

```
ARV: PROC OPTIONS (MAIN);
     DCL (SUMUJU, UJU) DEC FLOAT(6) INIT (0);
     DCL (SUMTAIS, TAIS) DEC FIXED(4) INIT (0);
     DCL U BIN FIXED (15);
     ON ENDFILE GOTO LOP;
NEXT: U=READ(U);
     IF ALT(U,'UJU') THEN DO;
       GET STRING (OMIST(U,'UJU')) LIST (UJU);
       SUMUJU = SUMUJU + UJU; END;
     ELSE DO;
       GET STRING (OMIST(U,'TAIS')) LIST(TAIS);
       SUMTAIS = SUMTAIS + TAIS; END;
     GOTO NEXT;
LOP: END ARY;
/*
// GO. SYSSCAN DD*
% ARV
15·2E10 12·3E4 15  20  13·5E 3
/*
//
```

Литература

I. Maurer H., Stucky W. Ein Vorschlag für die Verwendung syntax-orientierter Methoden in höberen Programmiersprachen, Angewandte Informatik, Nr. 5, p. 189–195.

2. Ginsburg S., Rounds E. Dynamic Syntax Specification Using Grammar Forms, "IEEE Transactions on software engineering", 1978, p. 44–55.

3. Jäderlund, Christn. Systematik, "Databehandling", Nr. 4, 1976, p. 35–42.

4. Тыугу Э. Программы и системы программирования. Таллин, ТПИ, 1977.

5. Вадер А.Р., Вооглайд А.О. Описание метаязыка системы построения трансляторов. – Тр. Таллинск. политехн. ин-та, 1978, № 439, с. 25.

6. Томбак М.О., Вооглайд А.О. Система построения эффективных многопроходных трансляторов с LR (к)-семантикой. Программирование, № 5. М., 1976.

7. Kabbaz, N.A. Control Sets of Linear Grammars. Information and Control, Nr. 25, ;1974, pp. 206–221.

8. Линдси Ч., Мюйлен С. ван дер. Неформальное введение в Алгол 68. М., Мир, 1973. 408 с.

9. Ершов А.П., Покровский С.Б. Об унификации алгоритмических языков. Проблемы кибернетики, № 32, М., Наука, 1977, с. 95–118.

A. Vooglaid, D. Liib

Total Optimization of the Memory of Precedence Analyzers

Summary

This article describes an extension of classical programming languages enabling the use of dynamic tree-structured variables in structured programming. The tree-structured variable described by BNF-language with a set of standard operations in it is implemented in the framework of compiler writing system of Tallinn Polytechnic Institute.

№ 482
TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 681.3.01

А.О. Вооглайд, Д.Б. Лийб

ПОЛНАЯ ОПТИМИЗАЦИЯ ПАМЯТИ АНАЛИЗАТОРОВ ПРЕДШЕСТВОВАНИЯ

В монографии А. Ахо и Дж. Ульмана [1] исследуется вопрос об уменьшении объема памяти, требующейся для матриц предшествования.

Показывается, что в определенных случаях, матрицу предшествования размерами $n \times k$ удается заменить двумя векторами длины n и k соответственно. В общем случае, если исходить только из матрицы предшествования, данный вопрос неразрешим.

Для практических работ в целях оптимизации синтаксических анализаторов выгодно, чтобы для каждого анализатора, работающего на матрицах предшествования (алгоритмы разбора типа: "перенос-свертка", "слабое предшествование", "простое предшествование", "смешанная стратегия предшествования"), находились соответствующие вектсры длиной n и k.

Чтобы добиться этого, надо вопрос ставить координально по-другому: исходя из контекстно-свободного языка найти для него векторы предшествования.

Данный подход к исследуемому вопросу является более общим и дает полное решение проблемы оптимизации памяти, требующейся для матриц предшествования. Только использование техники семантического равенства распознавателей [2] позволяет ставить задачу оптимизации таким образом. Дело в том, что данная техника позволяет организовать семантический анализ, полностью независимый от реализации конкретного метода синтаксического анализатора, иначе говоря, пользователю, который проектирует и реализует составляемый транслятор, нужно учитывать только уровень описания синтаксиса и семантики в рамках метаязыка системы построения трансляторов [3].

В анализе данных также исследуются проблемы сжатия разреженных матриц данных. Из этой области позаимствуем алгоритм Белла [4], слегка модифицируя его.

I. Основные понятия

Обозначим через A^* множество всех слов в алфавите A и $A^+ = A^*/\{\lambda\}$, где λ – пустое слово.

Контекстно-свободная грамматика (КСГ) – это упорядоченная четверка $G=(V_N, V_T, P, S)$, где V_T и V_N – конечные алфавиты терминальных и нетерминальных символов, такие, что $V_N \cap V_T = \phi$ P – множество правил подстановки вида $A \to x$, где $A \in V_N$, $x \in (V_N \cup V_T)^*$. Обозначим $V = V_N \cup V_T$. Слово y ($y \in V^*$) непосредственно выводимо из слова x ($x \in V^+$), обозначаем через $x \Rightarrow y$, если $x = z_1 A z_2$, $y = z_1 z z_2$ и $A \to z \in P$ ($z_1, z_2, z \in V^*$, $A \in V_N$). Если $z_2 \in V_T^*$, то слово y непосредственно канонически выводимо из слова x ($x \underset{K}{\Rightarrow} y$). Слово y выводимо из слова x ($x \overset{*}{\Rightarrow} y$), если существует последовательность слов $z_0, z_1, \ldots, z_K$ ($K \geq 0$), такая, что $x = z_0, y = z_K, z_i \Rightarrow z_{i+1}$ ($0 \leq i < K$). В дальнейшем будем использовать только канонический разбор и λ-свободную КСГ ($A \to \lambda \notin P$).

Пусть $G=(V_N, V_T, P, S)$ – некоторая КС-грамматика. Язык, определяемый грамматикой G – это множество слов $\mathcal{L}(G)$: $\mathcal{L}(G) = \{x / S \overset{*}{\Rightarrow} x \& x \in V_T^*\}$. Если $S \overset{*}{\Rightarrow} x$ ($x \in V^+$), то x называется канонической сентенциальной формой (КС-форма). Если $S \overset{*}{\Rightarrow} z_1 A z_2 \Rightarrow z_1 z z_2$ ($z_1, z \in V^*$, $z_2 \in V_T^*$, $A \to z \in P$), то z называется основой КС-формы $z_1 z z_2$.

Для описания некоторого метода синтаксического анализа для класса КС-грамматик $\mathcal{K}$ достаточно дать алгоритм, который по любой грамматике $G \in \mathcal{K}$ и любой КС-форме x находит в G элемент, непосредственно следующий за x в каноническом анализе КС-формы, т.е. алгоритм, реализующий шаг анализа. Разобьем шаг анализа на два действия [5]:

I. Детектирование. Найти основу z КС-формы x.

2. Редуцирование. Найти правило $A \to z$, такое, что $A \to z \in P$, $x_1 A x_2 \Rightarrow x_1 z x_2 \equiv x$, $x_1 \in V^*$, $x_2 \in V_T^*$.

Универсальным методом детектирования для всех КС-языков является метод предшествования [6], базирующийся на

следующих бинарных отношениях $(V \times V)$:

$$\beta = \{(A,\mathfrak{X}) \mid A \to \mathfrak{X}x \in P, x \in V^*\},$$

$$\rho = \{(\mathfrak{X},A) \mid A \to x\mathfrak{X} \in P, x \in V^*\},$$

$$\alpha = \{(\mathfrak{X},\mathfrak{Y}) \mid A \to x\mathfrak{X}y\mathfrak{y} \in P, x,y \in V^*\},$$

$$\lessdot = \alpha\beta^+,$$

$$\doteq = \alpha,$$

$$\gtrdot = (\rho^+\alpha\beta^*) \cap (V \times V_T).$$

Отношения $\lessdot, \doteq, \gtrdot$ называются отношениями предшествования и в матрице предшествования кодируем их соответственно -1, 0, $+1$. Следовательно, матрицу с элементами -1, 0, $+1$ или "пусто" будем называть матрицей предшествования.

Если отношения предшествования в КС-грамматике G попарно непересекающиеся и G сама λ-свободная грамматика, то G называется грамматикой предшествования.

Известно, что по любой λ-свободной КС-грамматике G можно построить грамматику предшествования G', такую, что $\mathcal{L}(G) = \mathcal{L}(G')$ [8].

Пусть M матрица предшествования размера $n \times k$. Будем говорить, что пара (f, g) векторов с целочисленными компонентами представляет M, если

$f = (f_1,\ldots,f_k)$, $g = (g_1,\ldots,g_k)$ и

$f_i < g_j$, если $M_{ij} = -1$,

$f_i = g_j$, если $M_{ij} = 0$,

$f_i > g_j$, если $M_{ij} = +1$.

Векторы f и g называем функциями предшествования.

2. Вычисление функций предшествования

Вычисление функций предшествования для любого λ-свободного КС-языка сначала проиллюстрируем на примере.

Пусть даны язык L и грамматика предшествования для него $G = (\{S,B\}, \{a,b\}, P, S)$, $L = \{aa, ab, ba, bb\}$,

P: $S \to aa$
$S \to ab$
$S \to ba$
$S \to Bb$
$B \to b$

	a	b	S	B
a	$\doteq$	$\doteq$		
b	$\doteq$	$\gtrdot$		
S				
B		$\doteq$		

M:

	1	2	3	4
1	0	0		
2	0	+1		
3				
4		0		

$$\begin{cases} f_1 = g_1 \\ f_1 = g_2 \\ f_2 = g_1 \\ f_2 > g_2 \\ f_4 = g_2 \end{cases} \Longrightarrow f_2 > g_2 \;\&\; f_2 = g_2 \qquad (I)$$

На основании системы уравнений (I) нетрудно убедиться, что для данной матрицы M не могут существовать функции предшествования f и g.

Сделав стратификацию направо [2] в грамматике G через продукцию $S \to aa$, получаем новую грамматику G′:

$$G' = (\{S,B,A\},\{a,b\},P',S)$$

P: $S \to aA$, $S \to ab$, $S \to ba$, $S \to Bb$, $B \to b$, $A \to a$

	a	b	S	B	A
a	$\lessdot$	$\doteq$			
b	$\doteq$	$\gtrdot$			
S					
B		$\doteq$			
A	$\doteq$				

M′:

	1	2	3	4	5
1	-1	0			
2	0	+1			
3					
4		0			
5	0				

Причем $\mathfrak{L}(G) = \mathfrak{L}(G')$ и функции предшествования следующие

$$f = (0,1,0,0,0),\quad g = (1,0,0,0,0).$$

В качестве другого примера приведем язык $L_1 = \{aa, ab, ba, ac, bc\}$.

$$G'' = (\{S, A, B\}, \{a,b,c\}, P'', S)$$

P″: $S \to ab$, $S \to aA$, $S \to Ac$, $S \to bB$, $B \to c$, $A \to a$

M″:

	1	2	3	4	5	6
1	-1	0	+1		0	
2	0		-1			0
3						
4						
5		0				
6						

$$\begin{cases} f_1 < g_1 \\ f_1 = g_2 \\ f_1 > g_3 \\ f_2 = g_1 \\ f_2 < g_3 \end{cases} \Rightarrow g_2 > g_3 \;\&\; g_2 < g_3 . \qquad (2)$$

Аналогично из системы уравнений (2) следует, что для данной матрицы M'' не могут существовать функции предшествования.

Сделав стратификацию налево через продукцию $S \to aA$, получаем новую грамматику $G''' = (\{S, A, A_1, B\}, \{a, b, c\}, P''', S)$

P''':

		M''':	1	2	3	4	5	6	7
$S \to ab$	1		+1	0	+1				
$S \to A_1A$	2		0		−1			0	
$S \to Ac$	3								
$S \to bB$	4								
$A \to a$	5			0					
$A_1 \to a$	6								
$B \to c$	7						0		

$f = (2,0,0,0,0,0,0)$, $g = (0,2,1,0,0,0,0)$.

Причем $\mathfrak{L}(G'') = \mathfrak{L}(G''')$.

Эти два примера характеризуют два разных типа конфликтов, при которых невозможно построить для матрицы предшествования функции предшествования. Для определения данных ситуаций поточнее приведем часть из алгоритма 7.1 [1].

Алгоритм Ч. Вычисления графа линеаризации.

Вход. Матрица M размера $n \times n$ с элементами −1, 0, +1 и "пусто".

Выход. Ориентированный граф, содержащий не более $n \times n$ вершин и называемый графом линеаризации для матрицы M.

Метод. (1) Пометим $2n$ вершин буквами $F_1, \ldots, F_n, G_1, \ldots, G_n$. В дальнейшем граф будет преобразовываться, причем в каждый момент будут существовать некоторая вершина $\hat{F}_i$, представляющая F_i, и некоторая вершина $\hat{G}_j$, представляющая G_j.
Вначале $\hat{F}_i = F_i$ и $\hat{G}_j = G_j (1 \leq i, j \leq n)$. Затем для всех i и $j (1 \leq i, j \leq n)$ выполним шаги (2) и (3).

(2) Если $M_{ij}=0$, построим новую вершину N, сливая $\hat{F}_i$ и $\hat{G}_j$. Теперь N представляет все те вершины, которые ранее представлялись вершинами $\hat{F}_i$ и $\hat{G}_j$.

(3) а) Если $M_{ij}=+1$, проведем дугу из $\hat{F}_i$ в $\hat{G}_j$.
б) Если $M_{ij}=-1$, проведем дугу из $\hat{G}_j$ в $\hat{F}_i$.

В теореме 7.I [I] доказывается, что матрица предшествования имеет функции предшествования тогда и только тогда, когда граф линеаризации ациклический.

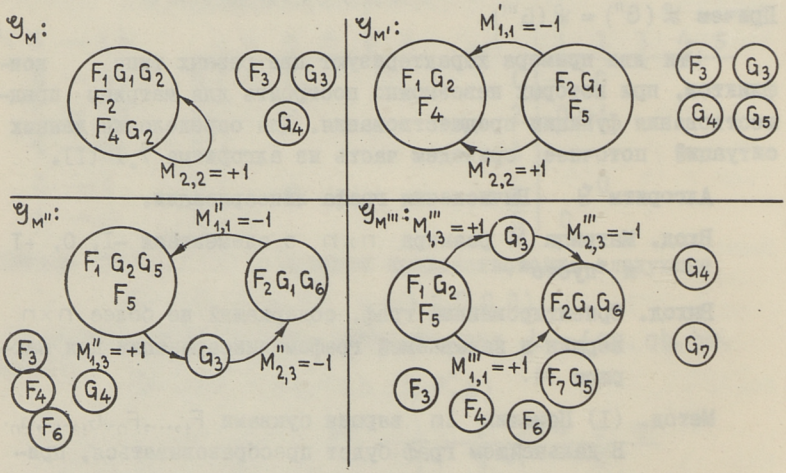
Данный граф может быть циклическим тогда, когда он имеет некоторую вершину N, имеющую путь, замыкающийся на себе таким образом, что:

а) все дуги представляют одинаковые элементы матрицы M в смысле алгоритма $\mathcal{Y}$;

б) существуют по крайней мере две разных дуги, представляющие разные элементы матрицы M.

Это и есть разные типы конфликтных ситуаций при построении функций предшествования.

Соответствующие графы для матриц M, M', M'' и M''' следующие.



Назовем данные конфликты конфликтами линеаризации типа L2 и L1.

L2 – все дуги в цикле представляют одинаковые элементы матрицы M.

L1 – в цикле найдутся по крайней мере две дуги, представляющие разные элементы матрицы M.

Пусть $\mathcal{G}=(\mathcal{T},\mathcal{K})$ – граф линеаризации, где $\mathcal{T}\subset\mathcal{N}$, $\mathcal{K}\subset$ $\subset\mathcal{N}\times\mathcal{N}$ $\mathcal{N}=\{1,2,3,\ldots\}$. Отметим информацию, содержащуюся в вершине $q_i\in\mathcal{T}$, следующим образом:

$$q_i \top {}^{i_1,\ldots,i_{Kq_i};\ j_1,\ldots,j_{mq_i}}_{q_1,\ldots,q_{lq_i}},$$

где q_i – номер вершины;

$i_1,\ldots,i_{Kq_i}$ – индексы компонент F_i, которые представляет вершина q_i;

$j_1,\ldots,j_{mq_i}$ – индексы компонент G_j, которые представляет вершина q_i;

$q_1,\ldots,q_{lq_i}$ – номера вершин, являющихся прямыми потоками вершины q_i,

и информацию, содержащуюся в дуге $(i,j)={}^{i,j}K_{\alpha}^{i',j'}$, где $\alpha=M_{i',j'}$. Если $\alpha+1$, то дуга проведена из $\hat{F}_i$ в $\hat{G}_j$, а если $\alpha=-1$, то дуга проведена из $\hat{G}_j$ в $\hat{F}_i$.

Прежде описания алгоритма полной оптимизации памяти матрицы предшествования приведем алгоритмы, являющиеся ее составными компонентами.

Алгоритм M & P. Преобразование предшествования.

Вход. КС-грамматика, λ-свободная $G=(\mathcal{V}_N,\mathcal{V}_T,P,S)$.

Выход. Грамматика предшествования G', такая, что $\mathcal{L}(G)=\mathcal{L}(G')$ и матрица предшествования для грамматики G'.

Метод. Алгоритм преобразования из статьи [8].

Алгоритм P1. Замена отношения $\lessdot$ или $\doteq$ на $\gtrdot$.

Вход. Грамматика предшествования $G=(\mathcal{V}_N,\mathcal{V}_T,P,S)$ и символы $\mathcal{X}\in\mathcal{V}$ и $\mathcal{Y}\in\mathcal{V}_T$.

Выход. Грамматика предшествования G', где $\mathcal{X}\gtrdot\mathcal{Y}$ и $\mathcal{L}(G)=$ $=\mathcal{L}(G')$.

Метод. Преобразование конфликтов типов P1 из статьи[8].

Алгоритм P2. Замена отношения $\doteq$ на $\lessdot$.

Вход. Грамматика предшествования $G=(\mathcal{V}_N,\mathcal{V}_T,P,S)$ и символы $\mathcal{X},\mathcal{Y}\in\mathcal{V}$.

Выход. Грамматика предшествования G', где $\mathfrak{X} \lessdot \mathcal{Y}$ и $\mathfrak{L}(G) = \mathfrak{L}(G')$.

Метод. Преобразование конфликтов типа P2 из статьи [8].

Алгоритм $\mathcal{Y}2$. Проверка цикличности графа линеаризации и выделения цикла, состоящего из вершин, которые соединены с дугами, представляющими одинаковые элементы матрицы M.

Вход. Граф линеаризации $\mathcal{Y} = (\mathcal{T}, \mathcal{K})$.

Выход. "Граф ациклический" или множество вершин $\mathcal{O}$, составляющих один цикл.

$$\mathcal{O} = \{ i_q j_q \in \mathcal{T} \mid {}^{i_q j_q}K^{i'_q j'_q}_{\alpha_q},\ q=1,\dots,n \,\&\, \alpha_1 = \alpha_2 = \dots = \alpha_n (j_m = i_{m+1}) \,\&\, (i_1 j_1) = (i_K j_K) \}. \quad (1 < m < K)$$

Метод. (I) Проверить по алгоритму из [1] на с. I2 является ли $\mathcal{Y}$ циклическим.

(2) Если граф $\mathcal{Y}$ ациклический, выдать сообщение "граф ациклический".

(3) Если граф $\mathcal{Y}$ содержит циклы, то по алгоритму из статьи [9] выделить все циклы.

(4) Выбрать из циклов любой такой цикл, дуги которого представляют одинаковые элементы матрицы M. Если таких нет, положить $A = \phi$.

(5) Составить массив A из вершин данного цикла.

Алгоритм $\mathcal{Y}1$. Проверка цикличности графа линеаризации и выделения цикла, состоящего из дуг, представляющих разные элементы матрицы M.

Вход. Граф линеаризации $\mathcal{Y} = (\mathcal{T}, \mathcal{K})$.

Выход. "Граф ациклический" или множество дуг $\mathfrak{L}$, составляющих один цикл.

Метод. Аналогичен алгоритму $\mathcal{Y}2$.

Алгоритм L2. Устранение конфликта линеаризации типа L2.

Вход. Грамматика предшествования $G = (V_N, V_T, P, S)$ граф линеаризации $\mathcal{Y} = (\mathcal{T}, \mathcal{K})$ и множество вершин $\mathcal{O} = \{q / q \in \mathcal{T}\}$, таких, что

$i_q j_q K \overset{i'_q, j'_q}{\alpha_q}$, $q = 1, \ldots, n$ & $\alpha_1 = \alpha_2 = \ldots = \alpha_n$ $(j_m = i_{m+1})$ & $(i_1 j_1) = (i_k j_k)$ $(1 < m < k)$

и матрица M.

Выход. Грамматика предшествования $G' = (V'_N, V_T, P', S)$ такая, что $\mathcal{L}(G) = \mathcal{L}(G')$.

Метод. Идея метода состоит в том, что при помощи алгоритма P2 некоторое отношение $\doteq$, которое участвовало в создании одной из вершин множества $\mathcal{A}$, заменяется на отношение $\lessdot$. При этом в графе линеаризации, построенном на новой грамматике G', соответственная вершина распадается на две.

Как правило, при конструировании вершин множества $\mathcal{A}$ участвует не одно отношение $\doteq$. Чтобы сделать правильный выбор, какое отношение $\doteq$ заменить на отношение $\lessdot$, для матрицы M надо условиться о выборе меры $\overline{P2}$, показывающей степень размытости отношений $\doteq$ в матрице M.

Пусть $\overline{P2} = \{$сумма длин между всеми отношениями $\doteq$ в матрице M$\}$. Под длиной между двумя отношениями предшествования $M_{i,j} = 0$ и $M_{i',j'} = 0$ подразумеваем величину $\sqrt{(i-i')^2 + (j-j')^2}$

(I) выбираем отношение $M_{i,j} = 0$, которое участвовало при конструировании вершин множества $\mathcal{A}$ таким образом, что

(2) используя алгоритм P2(i,j), мера $\overline{P2}$ новой матрицы M' была бы максимальной.

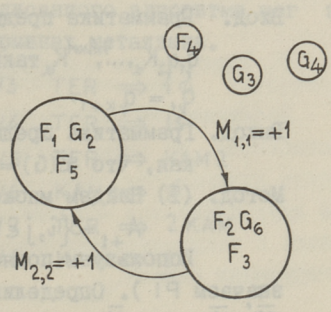
Приведем два примера, чтобы проиллюстрировать функцию меры $\overline{P2}$.

Пусть дана грамматика $G = (\{S, A, B\}, \{a, b\}, P, S)$

P: $S \to Aa$
$S \to ab$
$S \to Bb$
$S \to ba$
$A \to a$
$B \to b$

	a	b	A	B
a	$\gtrdot$	=		
b	=	$\gtrdot$		
A	=			
B				

	1	2	3	4
1	+1	0		
2	0	+1		
3	0			
4		0		



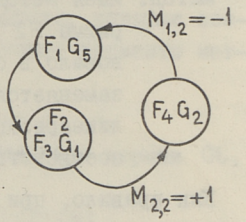
Здесь отношения $M_{1,2}=0$ и $M_{2,1}=0$ участвовали при конструировании вершин множества $\mathcal{U}$ и при конфликте линеаризации L2. Для преобразования графа можно использовать алгоритм L2 на месте (I,2) или на месте (2,I).

P 2 (I,2):

P': $S \to Aa$
$S \to aB_1$
$S \to Bb$
$S \to ba$
$A \to a$
$B \to b$
$B_1 \to b$

M'

	a	b	A	B	B_1
a	$\gtrdot$	$\lessdot$			$=$
b	$\doteq$	$\gtrdot$			
A	$\doteq$				
B		$\doteq$			
B_1					

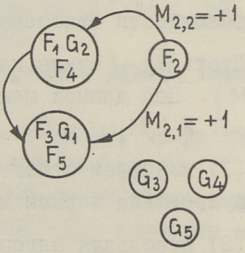


Из графа линеаризации видно, что он циклический P2(2,1):

P'': $S \to Aa$
$S \to ab$
$S \to Bb$
$S \to B_1a$
$A \to a$
$B \to b$
$B_1 \to b$

M''

	a	b	A	B	B_1
a	$\gtrdot$	$=$			
b	$\gtrdot$	$\gtrdot$			
A	$=$				
B		$=$			
B_1	$=$				



Из графа линеаризации видно, что он ациклический.

Соотношения меры $\overline{P2}$ матриц M' и M'' следующие

$$\overline{P2}(1,2) < \overline{P2}(2,1).$$

Алгоритм L1. Устранение конфликта линеаризации типа L1.

Вход. Грамматика предшествования $G=(V_N, V_T, P, S)$, дуги $q_1q_2K_\alpha, \ldots, q_{2n}q_{2n+1}K_\alpha$ такие, что $q_i=q_{i+1}$, $i=2n$, $1\leqslant n<\left[\frac{n}{2}\right]$, $q_1=q_{K+1}$.

Выход. Грамматика предшествования $G'=(V_N', V_T', P', S')$ такая, что $\mathfrak{L}(G)=\mathfrak{L}(G')$.

Метод. (I) Найдем множество A_{+1}

$$A_{+1}=\{i,j\in\mathcal{J}\mid i,j\ K_\alpha^{i',j'},\ \alpha=+1\}.$$

Используем понятие меры преобразования P1 (обозначаем $\overline{P1}$). Определим значение меры числом, равным $\overline{V_N'}-\overline{V_N}$, где $\overline{V}$ – число элементов алфавита V.

$$\overline{P1}(x,y)=\bar{V}'-V_N.$$

Во множестве A_{+1} найдем такие (i',j'), что $\overline{P1}(i',j')$ была бы минимальной.

(2) Используем алгоритм P1 на месте $P1(i',j')$.

Алгоритм 1. Полная линеаризация анализатора предшествования.

Вход. КС-язык L такой, что $\lambda \in L$.

Выход. Два целочисленных вектора $f=(f_1,\dots,f_n)$ и $g=(g_1,\dots,g_n)$, которые представляют матрицу предшествования M языка L.

Метод. (I) Поскольку L-КС-язык, найдем КС-грамматику $G=(V_N,V_T,P,S)$ такую, что $L=\mathcal{L}(G)$.

(2) Применим алгоритм M&P к данной грамматике G.

(3) Применим алгоритм $\mathcal{G}$ к полученной матрице предшествования.

(4) Используем алгоритм $\mathcal{G}2$. Если получим "граф ациклический", перейдем к шагу (I0). Если получим $\mathcal{OL}=\phi$, перейдем к шагу (7).

(5) Используем алгоритм L2.

(6) Перейдем к шагу (4).

(7) Используем алгоритм $\mathcal{G}1$. Если получаем "граф ациклический", перейдем к шагу (I0).

(8) Используем алгоритм L1.

(9) Перейдем к шагу (7).

(I0) Положим значение f_i равным длине самого длинного пути, начинающегося в $\bar{F}_i$, а g_j – длине самого длинного пути, начинающегося в $\bar{G}_j$.

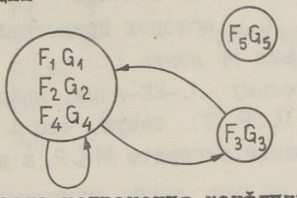
Проиллюстрируем работу вышеописанного алгоритма шаг за шагом. Пусть дан язык $\mathcal{L}(G)$ в терминах метаязыка.

```
GRAMNAME NUMBER
TERMINALS
ϕ
1
2
RULES
W1 TER ⇒ ϕ ϕ
W2 TER ⇒ ϕ 1
W3 TER ⇒ 1 ϕ
W4 TER ⇒ 1 1
W5 TER ⇒ KAM 2
W6 KAM ⇒ ϕ
W7 TER ⇒ 2 KAM
W8 KAM ⇒ 1
```

Полученная грамматика предшествования:

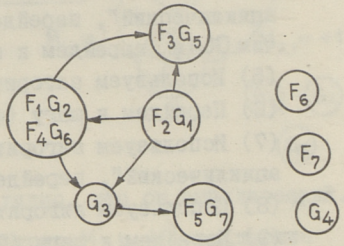
TER $\Rightarrow \phi\phi$	TER $\Rightarrow$ KAM 2
TER $\Rightarrow \phi 1$	KAM $\Rightarrow \phi$
TER $\Rightarrow 1\phi$	TER $\Rightarrow$ 2KAM
TER $\Rightarrow$ MAK1	KAM $\Rightarrow$ 1
MAK $\Rightarrow$ 1	

Ее граф линеаризации



Грамматика после устранения конфликтов линеаризации типа L1 и L2 и соответствующий граф линеаризации.

TER $\Rightarrow \phi 1$
TER $\Rightarrow 1\phi$
TER $\Rightarrow$ 2KAM
KAM $\Rightarrow \phi$
KAM $\Rightarrow$ 1
TER $\Rightarrow \phi$ PAM
PAM $\Rightarrow \phi$
TER $\Rightarrow$ KAM SAM
SAM $\Rightarrow$ 2
TER $\Rightarrow$ MAK1
MAK $\Rightarrow$ 1



Наконец векторы f и g следующие

$$f = (1, 2, \phi, 1, \phi, \phi, \phi);$$
$$g = (2, 2, 1, \phi, \phi, 2, \phi).$$

3. Модификация алгоритма Белла

Использование векторов, представляющих матрицу предшествования, во время синтаксического анализа приводит к утрате возможности обнаружения ошибки в моменте концентрации двух элементов V. Ошибка выявится при попытке редуцировать с несуществующим правилом. Происходит задержка в обнаружении ошибки.

Если учитывать алгоритм из статьи [7], то подобная ситуация создается умышленно при попытке исправлять ошибку автоматически. Вероятно здесь кроются интересные возможности исправления ошибок на базе векторов предшествования.

Но в отдельных случаях важно обнаружить ошибки как можно раньше и при возможности исправлять автоматически. Тогда можно в матрицу предшествования в некоторых пустых местах поместить информацию для облегчения обнаружения ошибок, а полученную матрицу данных сжать при помощи алгоритма Белла [4].

Приведем для пояснения алгоритма Белла только механизм воспроизведения данных:

$$M_{i,j} = \begin{cases} v(i), & f(i) < g(j) \\ w(j), & f(i) > g(j) \\ K(v(i), w(j)), & f(i) = g(j). \end{cases}$$

Данный алгоритм нуждается в два раза большей памяти, чем алгоритм линеаризации, но элементами начальной матрицы могут быть произвольные числа.

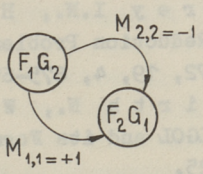
Но, увы, алгоритм Белла "хуже" в смысле свойства сжатия информации по сравнению с алгоритмом линеаризации.

Например, матрицу

M:

	1	2
1	+1	0
2	0	−1

$\mathcal{G}_M$



нельзя сжать при помощи алгоритма Белла, а функции предшествования следующие:

$$f = (1, 0) \quad g = (0, 1).$$

Поэтому советуем использовать модифицированный алгоритм Белла, где механизм воспроизведения данных следующий

$$M_{i,j} = \begin{cases} v(i), & f(i) < g(j) \\ w(j), & f(i) > g(j) \\ c_1, & L_1(i,j) = \text{'истинно'} \\ c_2, & L_2(i,j) = \text{'истинно'} \\ \cdots\cdots & \\ c_K, & L_K(i,j) = \text{'истинно'} \end{cases}$$

где L_i – логические выражения,
c_i – константы $(1 \leq i < K)$.

Обычно на практике достаточно, если $L_K \equiv i = j \pm K$, где $K = 0,1,\ldots,n-1$.

В заключение отметим, что алгоритм полной оптимизации внедрен в ВЦ ТПИ.

Также запрограммированы алгоритмы Белла, за что авторы признательны тов. К. Аллику.

Литература

I. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции. Том 2. Компиляция. Мир, 1978. 485 с.

2. Вооглайд А. Семантическое равенство распознавателей, работающих на грамматике LR(K) и грамматике предшествования с (I,I) ограниченным каноническим контекстом. – Тр. Таллинск. политехн. ин-та, 1976, № 411, с. 39-56.

3. Вадер А., Вооглайд А. Описание метаязыка системы построения трансляторов. – Тр. Таллинск. политехн. ин-та, 1978, № 439, с. 57-81.

4. Bell J. A compression method for compiler precedence tables. IFIP, 1974, pp. 359-363.

5. Grey I.N., Harrison M.A. On the Covering and Reduction Problems for Context-Free Grammars. I.ACM, 1972, 19, 4, 675-698.

6. Wirth N., Weber H. EOLER – a Generalization of ALGOL and its Formal Definition. Pt.I.Comm.ACM, 1966, 9, 1, 13-25.

7. Graham S.L., Rhodes S.P. Practical Syntactic Error Recovery. Comm. ACM, 1975, 18, 11, 639-650.

8. McAfee I., Presser L. An Algorithm for the Design of Simple Precedence Grammars. I.ACM, 1972, 19, 3, 385-395.

9. Tarjan R. Depth First Search and Linear Graph Algorithms. Siam J. Comput., Vol 1, N 2, June 1972.

A. Vooglaid, D. Liib

Tree-structured Variable in Data Processing

Summary

In this article the problem of syntax analyses optimizing is discussed. A method to compress precedence matrix into two precedence vectors is presented.

№ 482

TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 51:801

М.В. Лепп

СХЕМА ТРАНСЛЯЦИИ, РЕАЛИЗУЕМАЯ ПРИ ПОМОЩИ СПТ ТПИ

Введение

В течение ряда лет многие системные программисты интересуются проблемами автоматизации производства трансляторов.

Не будет преувеличением сказать, что каждая прикладная система построения трансляторов (СПТ), созданная в целях автоматизации, является вспомогательным средством в процессе составления трансляторов. Но выдаваемый ею транслятор может оказаться неэффективным в глазах пользователя СПТ. Это объясняется тем, что СПТ, как правило, генерируют трансляторы с унифицированными свойствами, ограничивая тем самым сферу применения на практике.

В данной статье альтернативой унифицирующей СПТ предлагается настраиваемая СПТ, которая направлена на производство трансляторов с различными качествами для проблемно-ориентированных языков. Один из способов достижения настраиваемости системы состоит в том, что пользователю позволяют определять схему трансляции создаваемого транслятора и управление его выполнением. При этом фиксируется язык, позволяющий удобно описывать схему трансляции, сохраняя простоту и ясность, свойственную графическому представлению структур.

I. Проблема настройки СПТ

Не вызывает сомнения, что системного программиста, занятого составлением транслятора при помощи СПТ, непременно заинтересует настройка системы на различные свойства транслятора. Настраивающая СПТ дает системному программисту некоторую свободу действий. Например, он получит возможность

выбора между трансляторами с альтернативными свойствами – в случае необходимости предпочесть неоптимизирующий транслятор оптимизирующему или скоростной с большим объемом памяти более медленному с меньшим объемом памяти.

Исходной точкой для решения проблемы настраиваемости служит структурный подход к проектированию и программированию сложных вычислительных процессов. Более подробнее об этом в работах [I, 2].

В основу подхода положен принцип постепенного разложения вычислительного процесса на подпроцессы до некоторого элементарного уровня. Процесс разложения можно прекратить в определенный момент и очередной полученный подпроцесс принять за элементарный, заинтересовавшись самими результатами и абстрагируясь от того, как они получены. В итоге получается некая иерархическая структура процесса, которая должна быть отражена в структуре программы [I]. Структура процесса определяет управление процессом.

В классических работах по процессам трансляции выделяются следующие подпроцессы или действия: лексический анализ, синтаксический анализ, семантический анализ, генерация кода. Такое разложение служит скорее методологической, чем реализованной цели.

В интересах эффективности создаваемого транслятора не всегда целесообразно придерживаться классической схемы разложения. Это дает некоторое разнообразие структур создаваемых трансляторов. Последнее немаловажно с точки зрения настраиваемости СПТ. Тем более, что свойства транслятора во многом зависят от его структуры. Кроме того, свойства создаваемого транслятора зависят и от методов, по которым реализуются соответствующие действия трансляции.

Один из способов добиться настройки СПТ – это сделать процесс трансляции управляемым со стороны пользователя. Другими словами позволить пользователю самостоятельно определять структуру генерируемого транслятора, в котором в качестве действий можно использовать как реализованные самой системой, так и пользователем. Имея в виду сказанное, при создании СПТ следует строго придерживаться принципа модульного программирования, при этом действия четко разграничи-

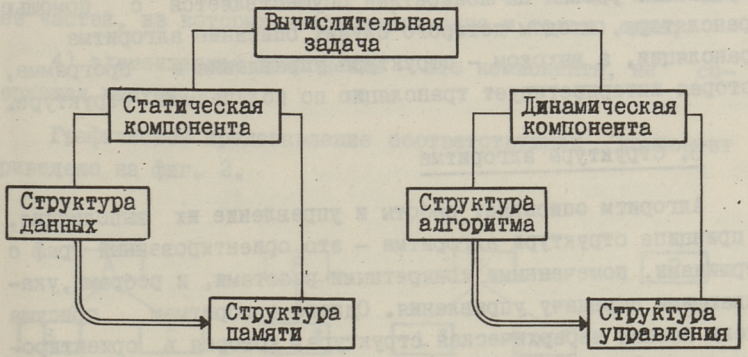
ваются от управления. Сохраняя стандартную схему обмена информацией, в СПТ должно быть несколько вариантов реализации одних и тех же действий, с тем чтобы пользователь мог бы выбрать удобную для него реализацию.

2. Трансляция как вычислительная задача

В этом пункте конкретизируются задачи, стоящие перед разработчиком СПТ. Успех решения поставленных задач определяет степень практичности конкретной СПТ.

В интересах краткости изложения ниже используются понятия и определения, заимствованные из монографии Холла [3]. Автор монографии называет любую задачу, решаемую на ЭВМ, вычислительной, независимо от того, имеются в виду числовые или нечисловые расчеты. Таким образом, трансляцию можно трактовать тоже как вычислительную задачу. Каждая подобная задача рассматривается на двух уровнях – абстрактном и конкретном. На обоих уровнях выделяются две компоненты – статическая и динамическая. Далее структура данных отображается в структуру памяти, а структура алгоритма – в структуру управления.

Описанный способ подхода к задаче иллюстрирует фиг. I.



Фиг. 1.

Поскольку проблемы управления трансляцией затрагивают больше динамическую компоненту, то изучим ее более подробно. Если речь идет о задаче трансляции, то динамическую компоненту удобно назвать "схемой трансляции".

Теперь, используя полученную терминологию, перейдем к конкретизации задач, возникающих при создании управляемого пользователем транслятора. Разработчику СПТ необходимо определить:

а) средства описания схемы трансляции на абстрактном уровне;

б) представление схемы трансляции на конкретном уровне;

в) автоматическое преобразование схемы трансляции с абстрактного уровня на конкретный.

Так как схема трансляции на абстрактном уровне совпадает со структурой алгоритма пользователя СПТ, то наиболее подходящим средством описания представляется язык высокого уровня, дающий возможность структурного выражения алгоритмов.

Схема трансляции на конкретном уровне это структура управления с условиями и правилами, определяющими порядок и время выполнения действий.

Автоматическое преобразование схемы трансляции с абстрактного уровня на конкретный осуществляется с помощью транслятора, входом которого служит описание алгоритма трансляции, а выходом - структура управления и программа, которая интерпретирует трансляцию по получаемой структуре.

3. Структура алгоритма

Алгоритм описывает работы и управление их выполнения. В принципе структура алгоритма - это ориентированный граф с вершинами, помеченными конкретными работами, и ребрами, указывающими передачу управления. Однако алгоритмам присуща очень важная иерархическая структура, которая в ориентированных графах утрачивается.

Можно согласиться с Холлом, что существуют реализованные средства описания управления, с помощью которых программист может описывать управление произвольной сложности. Но тем не менее они не обладают достаточной гибкостью для выражения алгоритма в форме иерархической структуры.

Иерархическая структура целесообразна по следующим причинам. Она позволяет представлять алгоритм сложной программы более естественным и разумным образом; легко находить и подключать подструктуры, программы которых принадлежат матобеспечению ЭВМ; без особых затруднений тестировать, отлаживать и модифицировать программу. Именно этими свойствами должен обладать аппарат определения алгоритма транслятора как сложной программы.

Далее будем исходить из утверждения, что алгоритм любой реальной задачи можно представить в виде иерархической структуры.

При определении же типа основных компонент структуры будем следовать монографии Джэксона [4]. Применяются компоненты четырех типов:

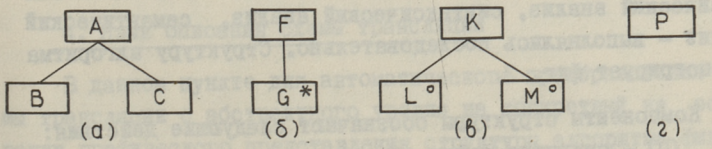
I) последовательность – это компонента, состоящая из одной или более частей, каждая из которых встречается один раз в определенном месте;

2) итерация – это компонента, состоящая из одной части, которая встречается ноль или более раз;

3) выбор – это компонента, состоящая из двух или более частей, из которых встречается одна и только одна;

4) элементарная компонента – это компонента, не содержащая частей.

Графическое представление соответствующих компонент приведено на фиг. 2.



Фиг. 2 (а) – A – последовательность из частей B и C; (б) – F – итерация из части G; (в) – K – выбор из частей L и M; (г) – P – элементарная компонента.

Каждая часть компоненты типа I) – 3) может в свою очередь быть самостоятельной компонентой типа I) – 3) или элементарной компонентой.

Определение структуры алгоритма только при помощи компонент типа I) – 4) оставляет структуру неизменной во время выполнения определенной задачи. Во многих случаях, однако, было бы полезно иметь в каком-то смысле динамически изменяемую структуру алгоритма. Поэтому каждую компоненту удобно рассмотреть как компоненту типа выбор, которую вместе с ее подчиненными при необходимости можно временно изъять из структуры непосредственно в ходе выполнения задачи. Тем самым мы получили средство для описания структуры алгоритма, которая управляется данными. Причем следует подчеркнуть, что действия, описанные алгоритмом, находятся в компонентах.

Пример

Пусть дана задача, где каждую последовательность пар идентификатор (i) – константа (c) требуется преобразовать в последовательность символов из единиц и нулей при следующих условиях:

(а) идентификатор состоит лишь из букв, а константа – из десятичных цифр;

(б) пара ic преобразуется в I, если $|i| = v(c)$, в 0, если $|i| \neq v(c)$, где $|i|$ – обозначает число символов идентификатора i и $v(c)$ – значение константы c.

Преобразование производится в пакетном режиме.

Рассмотрим данную задачу как задачу трансляции. Определим структуру трансляции таким образом, чтобы действия – лексический анализ, синтаксический анализ, семантический анализ – выполнялись последовательно. Структуру алгоритма демонстрирует фиг. 3.

Компоненты структуры обозначают следующие действия:

ОП – открытие файла текстов;
Д – декларирование таблиц идентификаторов (I) и констант (C);
ДН – очистка таблиц I и C;
И – запись идентификатора в таблицу I;
К – запись константы в таблицу C;

СИА – синтаксический анализ и построение разреженного дерева [5], присваивание значения "false" логической переменной L, определяющей наличие синтаксической ошибки;

ДК – доступ к корню разреженного дерева;

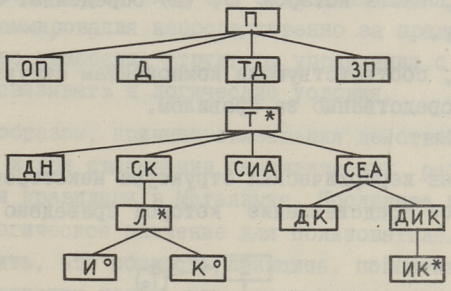
ДИК, СК, ТД – условия итерации;

Л – функция выбора;

СЕА – условие, зависящее от значения L и определяющее изъятие соответствующей подструктуры при наличии синтаксических ошибок;

ЗП – закрытие файла текстов.

Для решения задачи аппаратура трансляции в принципе представляется слишком мощной. Но выбрана она с тем, чтобы продемонстрировать действия трансляции, характерные для большинства трансляторов.



Фиг. 3.

По существу такие действия реализуются самой системой и поэтому их можно считать стандартными. В противоположность стандартным действия, реализуемые пользователем, можно определить как нестандартные действия.

4. Язык описания схемы трансляции

В данном пункте для автоматического преобразования схемы трансляции с абстрактного уровня на конкретный на основании графического представления структуры алгоритма фиксируется некоторый алгоритмический язык.

Анализ графического представления структуры алгоритма по Джэксону показывает, что язык должен содержать такие конструкции, которые отражали бы иерархическую структуру

компонентов четырех типов и позволяли бы действия трансляции приводить в соответствие с этими компонентами. Подобные конструкции можно представить в таком виде, чтобы их синтаксис был аналогичен синтаксису Метаязыка [6] конкретной СПТ, на котором описывается синтаксис объектного.

Тем самым язык можно представить как последовательность правил, каждое из которых имеет один из следующих видов:

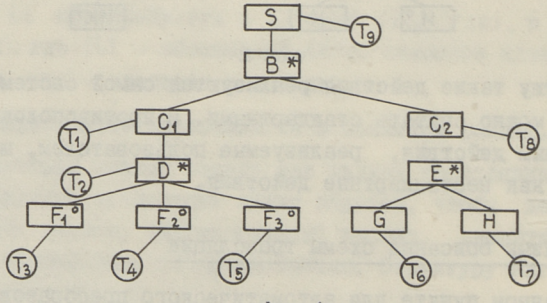
(а) A => BC
(б) K => L
(в) => M
(г) P => (D)*
(д) R =>

Семантика перечисленных конструкций следующая: (а) определяет последовательность A , части которой B и C; (б) и (в) определяют выбор K , части которого L и M; (г) определяет итерацию, часть которой D; (д) определяет элементарную компоненту.

Действия, соответствующие компонентам структуры, записываются непосредственно за правилом.

Пример.

Пусть дана иерархическая структура некоторого алгоритма, графическое представление которой приведено на фиг. 4.



Фиг. 4.

Компонентам структуры соответствуют действия $T_1,\ldots,T_9$, где действия T_1, T_8, T_9 - условия итерации и T_2 - функция выбора.

На языке этот же алгоритм выглядит следующим образом

S => (B)*	T_9	E => GH	
B => C_1C_2		G =>	T_6
C_1 => (D)*	T_1	H =>	T_7
D => F_1	T_2	F_1 =>	T_3
=> F_2		F_2 =>	T_4
=> F_3		F_3 =>	T_5
C_2 => (E)*	T_8		

Как выяснено, действия могут быть двух типов - стандартные и нестандартные. На стандартные действия ссылаются при помощи имени модуля или фазы. Фаза предусмотрена для реализации в транслятора стандартного действия достаточно большого объема. Фаза фиксируется структурой и содержит выбранные по умолчанию модули для выполнения действий в фазе. Пользователь имеет право заменять модули фазы на более подходящие для него. На нестандартные действия ссылаются при помощи имени или они записываются на каком-либо языке программирования непосредственно за правилом. Для динамического изменения структуры управления с её компонентами можно связывать и логические условия.

Таким образом, принцип связывания действий трансляции тот же, что и при связывании семантических работ с синтаксическими правилами в Метаязыке. Последнее имеет большое методологическое значение для пользователя. Кроме того можно добавить, что общность принципа, положенного в основу конструирования только что описанного языка и Метаязыка, дает возможность пользоваться одинаковыми механизмами при их реализации.

5. Схема трансляции на конкретном уровне

Напомним, что при переходе на конкретный уровень схема трансляции суть структура управления. Приступим к рассмотрению этой структуры в аспекте реализации.

Структура управления программой описывает способ, по которому, после исполнения одного действия отыскивается следующее, проверяется - нужно ли его выполнять и соответственно выполняется. Так как при иерархическом представлении структуры транслятора неизбежно проявится и структура управления, то естественно, что физическим отражением последней выбирается дерево. В таком дереве будут вершины трех типов - итерации, выбора и элементарного типа. Причем нет

необходимости в вершинах последовательного типа, поскольку порядок выполнения работ устанавливается алгоритмом прохождения дерева.

С вершиной типа выбор связывается функция, осуществляющая выбор. С вершиной типа итерация связывается условие, проверяющее совершение итерации. С вершинами элементарного типа связываются стандартные работы и работы пользователя и логическое выражение, которое выясняет, нужно ли выполнять данную работу в данный момент.

В каждой конкретной задаче трансляции преобразование структуры алгоритма в структуру управления производится автоматически - средствами СПТ. Помимо этого автоматически генерируется программа-интерпретатор, который и осуществляет управление - прохождение дерева и интерпретацию вершин.

Описываемая аппаратура несомненно является удобным для пользователя средством управления транслятором. Также следует отметить, что данную аппаратуру можно успешно использовать при генерировании управляемых пользователем пакетов со сложной структурой прикладных программ.

Литература

I. Дал У., Дейкстра Э., Хоор К. Структурное программирование. М., Мир, 1975.

2. Йодан Э. Структурное проектирование и конструирование программ. М., Мир, 1979.

3. Холл П. Вычислительные структуры. Введение в нечисленное программирование. М., Мир, 1978.

4. Jackson, M.A. Principles of programm design. Academic Press, London - New York - San Francisco, 1975.

5. Вооглайд А.О. Семантическое равенство распознавателей, работающих на грамматике (к) и грамматике предшествования с (I/I) ограниченным каноническим контекстом. - Тр. Таллинск. политехн. ин-та, 1976, № 411, с.39-55.

6. Вадер А.Р., Воеглайд А.О. Описание метаязыка в системе построения трансляторов. - Тр. Таллинск. политехн. ин-та, 1978, № 439, с. 83-92.

M. Lepp

A Translation Schema of the Compiler Writing System of the Tallinn Polytechnic Institute

Summary

In this paper the problem of the compiler writing system tuning according to the needed properties of the generated compiler is treated. A so-called compiling schema as a dynamic component of the compiling task is defined to describe the control structure of the new compiler.

№ 482
TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 518.5:519.2

Л.К. Выханду

О НЕКОТОРЫХ МЕТОДАХ УПОРЯДОЧЕНИЯ ОБЪЕКТОВ И ПРИЗНАКОВ В СИСТЕМЕ ДАННЫХ

I. Введение. Классические методы анализа данных занимаются подтверждением или отклонением выдвинутых исследователем научных гипотез. При этом остается неясным, каким путем были получены научные гипотезы и не учитывается, сколько до их подтверждения уже было использовано из наблюдений информации, чтобы выдвинуть эти же гипотезы.

Кроме того, следует еще отметить, что методика изучения качественных данных слабо развита.

Настоящая статья ставит своей целью разработку и практическую проверку новых методов открытия структуры многомерных данных. При этом развита новая методика нахождения глобального экстремума для т.н. монотонных систем [1,2,3]. Неформально можно сказать, что монотонной системой является такая система, для каждого элемента которой определена функция влияния, такая, что увеличение (уменьшение) элемента приводит к монотонному увеличению (уменьшению) влияний других элементов системы (точные определения даны в статье). Основываясь на этих результатах, в данной статье выводится класс методов изучения структуры многопараметрических систем, которые являются довольно эффективными при статистическом анализе данных.

2. Монотонные системы наблюдений

2.1. Постановка проблемы. В зависимости от способа представления информации о наблюдаемой эмпирической системе обычно различаются типы таблиц данных вида объект–признак, объект–объект и признак–признак. Все эти типы таблиц служат базой для тех или иных моделей извлечения информа-

ции об объектах. В последнее время подобная точка зрения на таблицы данных как на отправную точку построения моделей выдвинута теоретиками школы т.н."исследовательского" анализа данных(Exploratory Data Analysis).Известным представителем этой школы является американский статистик Дж. Тьюки. В СССР наиболее важные работы в этой и сходных областях исходят из ИПУ (Москва) (Э. Браверман, И. Мучник и др.) и из ИЭОПП (Новосибирск) (Б. Миркин).

Итак возникает естественный вопрос о построении технического аппарата, позволяющего создать теорию агрегирования данных на абстрактном уровне с минимальными предположениями об изучаемой системе, без каких-либо статистических (дистрибутивных) предположений.

Оказывается, что для задач вскрытия структуры эмпирической системы и определения естественных агрегатов объектов и признаков, без каких-либо статистических предположений, удобной оказывается теория специальных монотонных систем, созданная И. Муллатом [I].

2.2. Методика применения метода выделения ядер в анализе данных. Предложенный в [2] метод выделения ядра для монотонных систем W предъявляет к исследованию структуры данных следующие требования:

I. Для каждого элемента системы должна быть определена функция π, измеряющая значимость $\pi(w)$ (вес) элемента w во всей системе W.

2. Должны быть заданы правила пересчета значимостей (функции влияния f на веса) элементов системы при внесении изменений в вес данного элемента системы в заданном направлении.

Предъявленные в пп. I, 2 требования оставляют за исследователем довольно большую свободу в выборе как функций влияния, так и выборе правил осуществления влияний в системе. Единственное условие такого выбора состоит в том, что f и π должны быть между собой согласованы таким образом, чтобы, например, после исключения всех элементов w из системы W веса всех элементов $w \in W$ оказались равными нулю.

2.2.I. Построение монотонной системы влияния для объектов. В примере построения монотонной системы для объектов непосредственно на матрице данных используется шкала влияния объектов.

Пусть задана матрица данных X ($i=1,\ldots,N$; $j=1,\ldots,M$) и совершен переход к ее частотному виду Z, т.е. каждое значение x_{ij} заменяется на частоту этого значения у признака j. Мера вариации каждого объекта i определяется как сумма

$$S_i = \sum_{j=1}^{M} z_{ij}^2,$$

а мера вариации всей системы - как сумма

$$S = \sum_{i=1}^{N} S_i.$$

Влияние объекта на систему определяется в виде числа, на которое уменьшается сумма квадратов S, если из системы исключить этот объект. С точки зрения удобства организации вычислительного процесса можно считать, что исключаемый объект переходит в специальный класс неопределенности.

Так как для каждого признака j, каждая частота h_j имеет еще $h_j - 1$ равных себе значений, то сумма квадратов частот для всей системы уменьшается на величину

$$q_j = (h_j - 1)(h_j^2 - (h_j - 1)^2) = 2h_j^2 - 3h_j + 1.$$

Следующим шагом является построение функции, определяющей величину изменений во влияниях других объектов на систему при исключении одного объекта. Исключение объекта k из системы меняет влияние каждого другого объекта i на величину

$$S_i(h_{ij}) - S_i(h_{ij}-1) = \sum_{j=1}^{M} \delta[3h_{ij} - 2h_{ij}^2 - 1 - 3(h_{ij}-1) + 2(h_{ij}-1)^2 + 1] = \sum_{j=1}^{M} \delta(-4h_{ij} + 5),$$

где $\begin{cases} \delta = 1, \text{ если } x_{kj} = x_{ij} \\ \delta = 0, \text{ если } x_{kj} \neq x_{ij}. \end{cases}$

Исследуем теперь вопрос о монотонности полученной формулы, определяющей величину изменения влияния. Если предположить, что $M=1$ и $h_{ij}=1$, то изменение равно числу +1. Если же $M=1$ и $h_{ij} \geq 2$, то изменение имеет отрицательный знак. Следовательно, эта естественная функция не является

монотонной для случая исключения (оказания т.н. $\ominus$-действия по [I] элемента из системы объект-признак. Данный недостаток легко исправить: простое прибавление единицы во все классы гистограммы признака j дает вместо функции влияния $2h_j^2 - 3h_j + 1$ функцию

$$g_j = (h_j - 2)(h_j^2 - (h_j - 1)^2) = (h_j - 2)(2h_j - 1) = \\ = 2h_j^2 - 5h_j + 2.$$

Теперь исключение объекта k из системы меняет влияние каждого другого объекта i на величину

$$S_i(h_{ij}) - S_i(h_{ij} - 1) = \sum_{j=1}^{M} \delta [5h_{ij} - 2h_{ij}^2 - 2 - 5(h_{ij} - 1) + \\ + 2(h_{ij} - 1)^2 + 2] = \sum_{j=1}^{M} \delta(-4h_{ij} + 7).$$

Так как после добавления единицы во все классы гистограммы признака j наименьшая реальная частота $h_j = 2$, то легко убедиться, что изменение влияния монотонное.

2.2.2. Функция влияния для двумерного случая. В предыдущем разделе была выведена функция влияния f, гарантирующая монотонность системы объектов в случае $\ominus$-операции. Эта функция определена на множестве, элементами которого являются объекты. После выделения подгруппы объектов обычно возникает естественный вопрос о признаках, по которым эти объекты выделились в отдельную группу. Аналогичная интерпретационная проблема возникает и во всех методах группировки, использующих матрицу расстояний между объектами. После выделения подгрупп объектов требуется еще интерпретация полученных результатов, так как неясно, какие признаки служат основой группирования.

Поэтому важно развивать также методы, позволяющие провести одновременное разбиение объектов и признаков, облегчая тем самым интерпретацию результатов.

Для определения весовой функции π в системе объект-признак имеется множество вариантов. Следует различать два основные класса весовых функций - аддитивные и мультипликативные. В каждом классе принципиально существует бесконечное количество конкретных весовых функций.

2.2.2.I. Весовые функции аддитивного типа. Аддитивной системой весовых функций на матрице данных в частотном виде $Z=(z_{ij})$, $i=1,\dots,N$; $j=1,\dots,M$ назовем функции типа

$$S_i=\sum_j g(n_{ij}), \qquad P_j=\sum_i g(n_{ij}),$$

где $i=1,\dots,N$; $j=1,\dots,M$ $g(x)$ – функция частот.

Для обеспечения применимости метода выделения ядер необходимо требовать, чтобы функция $g(x)$ в области реальных частот x была монотонной при переходе от $g(x)$ к $g(x+1)$ или при переходе от $g(x)$ к $g(x-1)$.

В качестве весовых функций можно таким образом формально использовать многие известные из курса математического анализа функции. Однако требование содержательной интерпретации полученных группировок заставляют использовать весовые функции, имеющие приемлемую интерпретируемость. К таким функциям относятся различные энтропийные функции.

Вместо классической энтропийной функции $H(x)$ множества лучше использовать ее приближение

$$H_2(x)=\sum_i p(x_i)^2.$$

Определенная указанным способом энтропия $H_2(x)$ обладает всеми свойствами обычной энтропии и как показал Нильсон [4], может служить основой для создания богатой системы многомерной статистики. Для наших целей важно, что для пары векторов X и Y можно составить обычную двумерную частотную таблицу с частотами n_{ij}, $i=1,\dots,p$, $j=1,\dots,q$ и взаимная информация (см. [4]) выражается как

$$I(X,Y)=N^2\sum_i\sum_j n_{ij}^2\Big(\sum n_{i\cdot}^2\cdot\sum n_{\cdot j}^2\Big)^{-1}.$$

Другой тип аддитивных весовых функций принадлежит к классу функции влияния $g(n)$ объектов на систему (см. 2.2.I).

После выбора весовой функции и подтверждения ее монотонности путем проверки направления изменения весов элементов как результата от осуществления $\ominus$-действий на эти элементы, принимается процедура выделения ядер. При этом однако важно установить стратегию применения этой процедуры ПВЯ.

Например, в системе объект–признак возможны следующие стратегии, обеспечивающие применение ПВЯ.

I. В качестве элемента W следует выбирать объект и ПВЯ произведет исключение объектов, начиная с объекта с минимальным весом.

2. В качестве элемента W выбирается объект или признак и исключение по правилам ПВЯ организуется по минимуму влияния на систему как объектов, так и признаков.

3. В качестве элемента W выбирается элемент матрицы данных, находящийся в строке i в столбце j и по правилам ПВЯ исключение элемента начинается:

а) $\min\limits_{i,j} \pi(R_i, V_j)$,

б) $\min\limits_{i,j} (R_i + V_j)$,

в) $\min\limits_{i,j} (q_{ij} + \bar{q}_{ij})$,

где q_{ij} – функция влияния по гистограмме признака;
$\bar{q}_{ij}$ – та же функция влияния по гистограмме объекта.

2.2.2.2. Весовые функции мультипликативного типа. Наряду с аддитивными весовыми функциями можно предположить весовые функции мультипликативного типа:

а) вес элемента матрицы данных $X = (x_{ij})$ определяется как число

$$W_{ij} = q(x_{ij}) \cdot \bar{q}(x_{ij}),$$

где $q(x_{ij})$ – значение функции влияния по частоте n_{ij} значения элемента матрицы данных x_{ij} по признаку j;

$\bar{q}(x_{ij})$ – такое же значение по гистограмме значений объекта i.

В качестве функции $q(x)$ можно, например, предположить функцию $q(n) = 2n^2 - 5n + 2$ или функцию $q(n) = n^{\beta} - 1 \; (\beta \geq 1)$;

б) вес элемента матрицы данных x_{ij} определяется как число

$$W_{ij} = \sum_i q(x_{ij}) \cdot \sum_j q(x_{ij}).$$

Стратегию выбора элемента монотонной системы можно опять варьировать:

а) в качестве элемента W выбирается объект и полностью исключается минимальный по влиянию объект;

б) в качестве элемента W выбирается объект или признак;

в) в качестве элемента W выбирается элемент матрицы данных, определенный индексами i и j.

3. Алгоритм выделения ядра монотонной системы для матрицы данных

3. Алгоритм приводится в форме шагов. Рассматривается двумерный N×M -массив X , элементами которого служат натуральные числа в диапазоне от 0 до 255.

A0. По матрице данных X вычисляется таблица частотных полос W для признаков. Инициируются массивы S и V: $S(I) = 0$ для $I=1,\ldots,N$ и $V(J)=0$ для $J=1,\ldots,M$.

A1. (Вычисление влияний). Для $I = 1(1)N$, $J=1(1)M$ ($q_{ij} = 5z_{ij} - 2z_{ij}^2 - 2$; $S(I) = S(I) + q_{ij}$; $V(J) = V(J) + q_{ij}$).

A2. (исключить или объект или признак?). Найти $R = \max_i S(I)$, $g = \max_j V(J)$ и $\max(h,g)$. Идти соответственно к A3 или A4.

A3. (Исключается признак). Для всех пока неисключенных $S(I)$: $S(I) = S(I) - q_{ij}$. Идти к A5.

A4. (Исключается объект). Для всех пока неисключенных $S(I)$ $S(I) = S(I) + P_i$, где $P_i = 4T - 7L$ (T - сумма частот совпадающих признаков объектов I) и H (H соответствует $\max S(I)$, L - количество совпадений); одновременно с вычислением P_i исправляется и $V(J) = V(J) + (6z_i - 20)z_i + 16$.

Во вспомогательной таблице W для всех исключенных значений объекта H вычесть единицу. Приравнять $S(H) = -\infty$ в целях указания на исключенность объекта H.

A5. Если еще имеются неисключенные объекты или признаки, то идти к A2, иначе конец.

После применения алгоритма A матрица данных X выводится на печать с переставленными строками и столбцами в порядке исключения объектов и признаков. Ядро помещается в нижний правый угол. Затем по образованной последовательности элементов монотонной системы следует двигаться в обратном направлении до первого локального максимума среди

весов элементов в момент их исключения. Все элементы последовательности с конца до этого максимума включительно принадлежат ядру.

Литература

I. Муллат И.Э. Экстремальные подсистемы монотонных систем.I. - Автоматика и телемеханика, № 5, 1976, с. 130-139.

2. Муллат И.Э. Экстремальные подсистемы монотонных систем. II. - Автоматика и телемеханика, № 8, 1976, с. 169-178.

3. Муллат И.Э. Экстремальные подсистемы монотонных систем. III. - Автоматика и телемеханика, № 1, 1977, с. 109-119.

4. Нильсон А. Свойства сумм квадратов. Изв. АН ЭССР. Сер. физ.-мат.наук. 1965.

L. Vyhandu

Some Methods to Order Objects and Variables in Data Systems

Summary

In this paper a theory of monotonic systems for data tables is presented. As a result a new family of data ordering methods is developed.

№ 482

TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED
ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 681.3.06

Л.К. Выханду, Т.Ф. Лучковский,
Я.Я. Тепанди

СИСТЕМА ХРАНЕНИЯ И ОБРАБОТКИ ДИСКРЕТНОЙ ИНФОРМАЦИИ (СХОДИ)

I. Введение. В настоящей статье обобщается опыт построения пакетов прикладных программ (ППП), способных настраиваться на разные задачи из специализированной области. Рассматриваемый ППП предназначен для ввода, контроля и обновления данных определенного вида, поиска и выдачи нужной информации и составления отчетов на основе хранимой информации.

Пакет построен как надстройка над системой управления базой данных (СУБД) кодасиловского типа [4] для ОС ЕС. Такое решение позволило эффективно организовать структуризацию данных, обеспечить ускорение и упрощение их обработки. Основное внимание уделено настраиваемости системы на различные приложения.

Система СХОДИ с идейной стороны обобщает опыт эксплуатации системы СОДИ [1,2,3].

2. Данные и требования пользователя. Данными пользователя служат документы. Обычно документы, хранимые в СХОДИ, имеют дополнительную естественную структуру, например, данные поступают из разных районов, в различные промежутки времени, по различным каналам и т.д. Учитывается, что на практике требуется сохранение всех старых версий документов и возможность работы со старыми версиями.

Выходные данные выбираются по структуре или по различным логическим условиям из хранимых данных. Во время составления отчетов с данными осуществимы различные преобразования и вычисления.

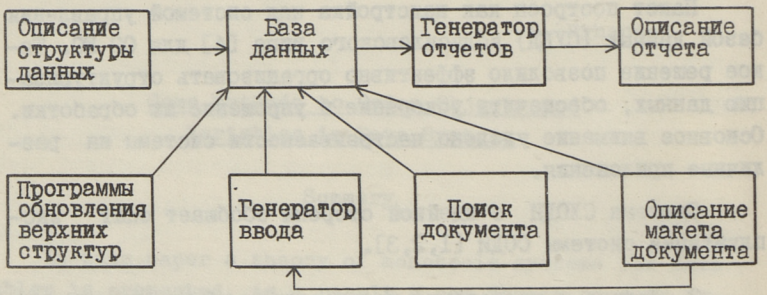
При создании системы учитывались также следующие требования пользователя:

- настраиваемость на различные типы данных как при вводе, так и при выводе;
- настраиваемость на различные конфигурации ЭВМ;
- простота внедрения и эксплуатации.

3. Основные концепции СХОДИ. На уровне вводных данных пользователя единицей ввода, хранения и выдачи информации служит документ, который хранится в виде упакованной записи в базе данных. Данные в базе, на основе внутренней структуры вводимой информации, объединяются в различные подмножества.

Выводные данные могут представлять собой распечатки некоторых документов или их частей, а также таблицы (матричные отчеты) различной формы.

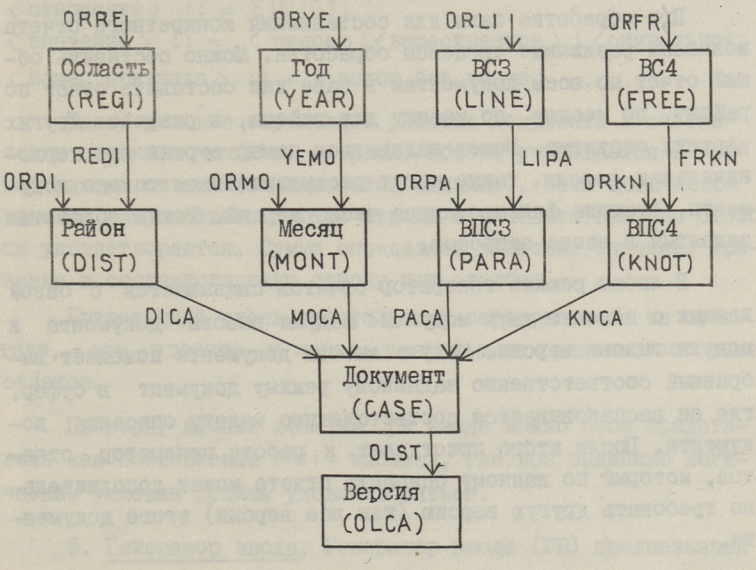
Структурная схема ШП СХОДИ следующая



Запись в базе данных представляет собой совокупность признаков документа в целочисленном, реальном или алфавитно-цифровом представлении. Признаки упакованы соответственно заданному пользователем макету документа. Генератор ввода, используя макет документа и описание базы, вводит данные в базу и присоединяет их к определенным пользователем структурам базы.

При составлении отчета по определенным в описании отчета значениям структур базы программы генератора отчетов вызывают из базы документы и составляют таблицу. Ниже приводится детальное описание структуры базы и принципов реализации генератора ввода и генератора отчетов.

Структура базы следующая:



(ВС – верхняя структура, ВПС – верхняя подструктура).

Прежде всего подчеркиваем, что все имена в схеме и их смысловое содержание – условные. Они служат двум целям: для облегчения описания данной схемы и как иллюстрированный пример.

Все области и годы упорядочены соответственно отношениями ORRE и ORYE. (В дальнейшем предлог OR показывает всегда упорядоченность отношения). Каждой области посредством отношения REDI подчинены все районы этой области, которые в свою очередь упорядочены отношением ORDI. Наконец через отношение DICA районы связаны с относящимися к ним документами. Аналогичная связь имеется для годов, месяцев и документов. В базе для потребителя дополнительно предвидена возможность определения еще двух структур такого же вида (верхние и подструктуры 3 и 4).

Еще раз подчеркиваем, что конкретная семантика всех четырех структур полностью определяется пользователем.

Практика показала, что часто оказывается полезным сохранить в базе или все или часть старых версий документов.

Поэтому каждому документу через отношение OLST поставлены в соответствие все его старые версии.

При обработке базы для составления конкретного отчета возможны различные процессы обработки. Можно составить общий отчет по всем документам в базе или составить отчет по району, по месяцу, по месяцу для района, в разрезе других верхних структур. Может задаваться номер версии или первоначальная версия, также могут рассматриваться только документы, имеющие фиксированное число версий. Режим обработки задается в языке запросов.

В любом режиме генератор отчетов связывается с базой данных с помощью двух модулей: модуля вызова документа и модуля вызова версии. Модуль вызова документа помещает выбранный соответственно заданному режиму документ в буфер, где он распаковывается соответственно макету описания документа. После этого приступает к работе генератор отчетов, который по данному описанию отчета может дополнительно требовать другую версию (или все версии) этого документа.

4. Язык логических условий. Главным оружием при проверке данных, выборе объектов для отчетов и составления самого отчета служат логические условия.

В роли логического условия допускается любая формула исчисления высказываний в дизъюнктивной или конъюнктивной нормальной форме. Элементарные высказывания должны иметь вид aRb, где a - номер признака, R - отношение сравнения ($=, \neq, <, >$), b - константа.

Скобки в логических условиях прямо не используются. Вместо этого имеется возможность создать новые признаки. Аппаратура новых признаков на практике намного мощнее аппараторы скобок в логических условиях.

Синтаксис логического условия в нотации Бэкуса следующий:

⟨составное условие⟩ :: = ⟨сумма⟩ ⟨условие⟩ | ⟨условие⟩
⟨сумма⟩ :: = S ⟨номер признака⟩
⟨условие⟩ :: = K ⟨выражение⟩ | D ⟨выражение⟩
⟨выражение⟩ :: = ⟨терм⟩ | ⟨терм⟩ & ⟨выражение⟩ | ⟨терм⟩ V ⟨выражение⟩

< терм > :: = < номер признака > < отношение > < значение >
< отношение > :: = E|N|G|L
< значение > :: = < целое > | < вещественное > | < символьное >
< номер признака > :: = < целое без знака >

Такое условие определяет участие документа в составлении строки и столбца матрицы. Объект учитывается в данной клетке отчета (складывается единица), если логическое условие для соответствующей строки и соответствующего столбца удовлетворяются. Сумма определяет сложение значения признака в соответствующую строку или столбец.

Приведенный здесь простой язык оправдал себя на практике, как удобное и легко усваиваемое средство описания отчетов.

Проверку данных объекта при вводе можно себе представить как составление N×1 – таблицы, где все заданные логические условия должны удовлетворяться.

5. Генератор ввода. Генератор ввода (ГВ) предназначен для введения изменений в базу данных. К таким изменениям относятся:

1) добавление записей в базу данных,
2) стирание записей из базы данных,
3) введение изменений в существующие записи с одновременным сохранением старых версий записи.

Данные могут быть оформлены различными способами. находиться на перфокартах или на магнитной ленте. ГВ преобразует данные в единый формат. Во время этого этапа производится первый контроль данных – соответствие данных с макетом. Проверяется, правильно ли оформлены данные (только допустимые символы, правильные разделители); соответствует ли количество признаков значению, указанному в макете; не превышают ли значения цифровых признаков максимально допустимых значений; соответствуют ли типы вводимых признаков типам данных, указанных в макете.

После проверки вводимых данных по макету неупакованная запись находится в буфере. Здесь происходит второй этап контроля данных – проверка логических условий. Логические условия проверяются не только при добавлении запи-

сей в базу, но и при введении изменений в существующие записи. Набор логических условий для конкретного типа вводимых документов удобно оформить как член библиотечного набора данных. При работе ГВ логические условия читаются в общие области памяти.

Когда запись удовлетворяет всем логическим требованиям, производится третий этап контроля вводимых данных – контроль по структуре базы данных. В записи имеются ключевые признаки, значения которых определяют, к каким записям верхних структур данная запись присоединяется. Значения ключевых признаков должны быть допустимы, т.е. соответствующие записи верхних структур должны находиться в базе данных. Значения ключевых признаков должны однозначно идентифицировать запись в базе данных. При добавлении данных в базе не должно быть записи с теми же значениями ключевых признаков, при введении изменений и при стирании такая запись должна существовать.

ГВ выдает диагностические сообщения об ошибках, обнаруженных при проверке вводимых данных, а также статистику о базе данных (сколько записей создано, сколько стерто) после завершения работы.

6. Генератор отчетов. Генератор отчетов (ГО) использует язык запросов, состоящий из функциональных операторов, каждый из которых определяет работу некоторой подпрограммы ГО. В общем случае функциональные операторы имеют следующий стандартный формат:

<функциональный оператор> :: = <вызов программы> <операнды> @

<операнды> :: = <пусто>|<операнд>|<операнды><операнд>

<вызов программы> :: =<ключевое слово> = <имя программы>,

<операнд> :: = <ключевое слово> = <значение>,

<значение> :: = <ключевое слово>|<число>|<специальное значение>

Пример: PRO=RGE, ROW = 8, COL=10, DUM=5@
PRO=PRINT, @@.

Содержание: составить таблицу отчета (RRO=RGE), заполняя каждую клетку числом 5 (DUM = 5). Число строк – 8 (ROW = 8), число столбцов – IO (COL = IO). Таблица печа-

тается в стандартной форме (ширина клетки – 6 позиций) без заголовок (PRO=PRINT). Первый и второй символы @ служат концом функциональных операторов, третий @ заканчивает входные данные.

Список функциональных операторов следующий:

– переход на другие каналы ввода-вывода;

– образование новых признаков; составление таблиц для быстрой проверки логических условий;

– определение значений верхних структур и уровня версий;

– составление таблицы отчета;

– арифметика над таблицей;

– печать таблицы;

– каталогизация отчета.

Отметим, что в рамках этого языка запросов имеются еще четыре простых подязыка, объединенные под термином "специальное значение". Это подязык арифметики на таблицах ARIT (определены сумма, разность, отношение и прирост на строках, столбцах и клетках таблицы); подязык оформления таблицы для оператора PRINT; подязык логических условий, следующий за ключевым словом TEXT в операторе SYNT (см. 4); подязык определения новых признаков, следующий за ключевым словом VALUE в операторе SYNT.

Наиболее интересным из них и является определение новых признаков, так как здесь могут произойти вторичные обращения генератора отчетов к базе данных (первое обращение происходит на основе режима, определенного в операторе REG, который вызывает очередной документ).

Введение нового признака объекта может определить некоторое значение (арифметические действия с участием других признаков или других версий, объекта, численное значение некоторого логического условия, число версий объекта, удовлетворяющих некоторому логическому условию), которое присваивается некоторому дополнительному полю распакованного объекта. По характеристикам этого поля, данным в определении нового признака, составляется расширенный макет объекта и проверка логических условий происходит в обычном порядке.

Введение нового признака может также определить некоторое действие, связанное с изменением версии данного доку-

мента (запоминание данной версии, переход к новой версии некоторого уровня или удовлетворяющей некоторому условию, возвращение к запомненной версии).

Язык создания новых признаков может рассматриваться как простой язык манипуляции данными.

В качестве примера использования механизма новых признаков вычислим для районов P1 и P2 (признак 2 объекта)сумму значений признака 4 в тех документах, которые поступили в 1979 году и в течение 10 дней после поступления изменились более чем в два раза.

Функциональный оператор определения новых признаков и логических условий выглядит так:

```
PRO=SYN,VAL=M82P1A201*M8R3S(8-10)A202*,
TEXT=K10G7812*/K2E'P1'*K2E'P2'*/
S4K201G2*/K202L11*@.
```

Предполагается, что признак 8 хранит год и месяц изменения, а признак 9 – число (день) изменения. В признаках 10 и 11 аналогичным образом содержится дата поступления документа; число признаков в документе меньше 200.

После ключевого слова определяется новый признак с порядковым номером 201(A201), для которого M82 определяет значение "число версий (включая последнюю), которые удовлетворяют первому логическому условию (P1) после условий описания таблицы (т.е. после третьей наклонной черты)". Это условие K202L11. Символом конца определения признака служит звездочка. Признак 202 определен (M8) как значение разности дат, представленных в вышеописанном виде; номера признаков, содержащих год и месяц для обеих дат, даны после буквы S. Значение R3 определяет этот признак как вычисляемый каждый раз, когда он встречается в логических условиях.

После вызова очередного документа и проверки общего условия (год больше 1979), генератор отчетов вычисляет признак 201, сосчитав число его версий (включая его самого), удовлетворяющих условию K202L11 (разность дат изменения и поступления, вычисляемая для этой версии, меньше 11).

Затем формируется таблица отчета, в которой имеется два ряда (условия рядов оканчиваются второй наклонной чертой) и один столбец (условия столбцов оканчиваются третьей наклонной чертой). Обозначение S4 условия столбца определяет добавление значения признака 4 (тип его берется из макета) в клетку первого столбца таблицы, если для данного документа удовлетворено общее условие, условие ряда и столбца этой клетки.

7. Настраиваемость системы. Внедрение любого ШПП представляет собой настройку системы для входных и выходных данных, для заданной конфигурации ЭВМ и для режима эксплуатации. Дальше и описываются возможности настройки базы данных пользователем.

I. Пользователь может в макете объекта отказаться от некоторых верхних структур.

2. При желании можно изменить имена выбранных верхних структур в зависимости от их конкретного содержания.

3. В соответствии с выбранной структурой данных можно генерировать новое описание базы данных.

4. В соответствии с предвиденным количеством документов выбирается размер базы.

5. В соответствии с типами и максимальными значениями признаков в документе составляется и каталогизируется макет документа.

6. Задаются возможные значения верхних структур базы. Система обеспечивает контроль синтаксиса вводимых данных и создание верхних структур.

7. Составляются и каталогизируются наборы данных логических условий контроля вводимых объектов. Задача облегчается самонастройкой системы - а именно, каждый документ должен участвовать во всех определенных пользователем верхних структурах, а допустимость значений соответствующих признаков проверяется автоматически.

8. Для облегчения составления пакетов задания в СХОДИ имеются готовые процедуры на языке ЯУЗ, которые организуют вызов нужной базы данных (по имени), а также программу составления заданного отчета (по имени).

При настройке на выходные данные пользователя основной работой является составление и каталогизация запросов на отчеты. Здесь следует решить, какие документы выбираются через верхние структуры схемы базы, какие через логические условия по признакам.

Аналогично надо решить (если это требуется), как определить уровень версии документа. Логические условия обеспечивают более богатые возможности, но оператор REG модифицируется легче и выглядит естественнее.

Для настройки на имеющееся оборудование достаточно иметь средства ОС ЕС, которые обеспечивают независимость программ от конкретной конфигурации ЭВМ. Если требуется настройка на память ЭВМ, то размеры всех рабочих полей для базы данных, генератора ввода и генератора отчетов заданы в четырех подпрограммах BLOCK DATA.

8. Подготовка к эксплуатации. В конечном счете результатом настройки системы является пакет программы для эксплуатации, поэтому на всех этапах внедрения следует кроме обеспечения требуемых функций заботиться и об эффективности принятых решений в смысле скорости работы ЭВМ, удобства работы и уменьшения возможных ошибок.

Для этих целей следует по возможности пользоваться каталогизированными запросами и процедурами. Для обновления и дополнения базы данных должно быть достаточным знание идентификатора модифицируемой базы данных и правил перфорации документов, а для составления отчета - знание идентификаторов базы и данного отчета плюс умение определять значения верхних структур, по которым составляется отчет.

Наконец следует написать руководства для эксплуатации (макет перфорации документа, руководство по составлению входного пакета перфокарт для обновления базы и составления отчетов, руководство оператора), так как эти руководства зависят от конкретного применения.

9. Проблемы настройки системы на базу данных. Из литературы по базам данных известно, что выбор окончательной структуры базы довольно трудоемкий и нелегкий процесс.

Нами выбрана одна сравнительно простая структура, которую можно автоматически изменить только в сторону уменьшения сложности структуры. Основной единицей хранения в системе служит документ (с его старыми версиями). В теории баз данных указывается, что выбор документа в качестве основной единицы хранения не особенно удачное решение, так как процессы обработки не учитываются. Мы думаем все-таки, что умелое использование верхних структур, а также эффективно работающий механизм логических условий дает приемлемый компромисс между неструктурированными данными и базой данных, предвиденной для конкретных процессов. Наш опыт показывает, что СХОДИ работает сравнительно хорошо как со многими ранее не предвиденными поисковыми запросами, так и каталогизированными заданиями.

Литература

I. Выханду Л.К., Архитектура "СОДИ". - Сб. Трудов НИИ Культуры РСФСР, № 32, 1976, с. 339-367.

2. Выханду Л.К. Об интегрированных системах обработки дискретной информации. - Тр. Таллинск. политехн. ин-та, № 313, 1971, с. 3-14.

3. Выханду Л.К., Микли Т.И., Рыбовыйтра М.Г., Томбак М.О. Об одной методике создания интегрированных систем обработки данных. - Труды ВЦ ТГУ, № 30, 1974, с. 3-7.

4. Мартин Д. Организация баз данных в вычислительных системах. М., Мир, 1978.

L. Vyhandu, T. Luchkovsky, J. Tepandi

A System for Discrete Data Storage and Processing (SHODI)

Summary

In this paper an overview of a data processing system "SHODI" is presented. It is built on a Codasyl-type data base system for ES-1022 (OC 4.0) computer and is specialized for count-type reports.

№ 482

TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 658.5.011.56.001.63:681.3.016

Ю.Г. Лааст-Лаас

О ПРОЕКТИРОВАНИИ ЛОГИЧЕСКОЙ СТРУКТУРЫ БАЗ ДАННЫХ ДЛЯ ЭКОНОМИЧЕСКОГО АНАЛИЗА

Введение

Информационное обеспечение задач экономического анализа деятельности промышленного предприятия рассматривается и проектируется нами как часть информационного обеспечения управления предприятием. Комплексный подход к задачам экономического анализа [3] требует обработки информации на основе единой базы интегрированных данных (БД). В настоящей статье рассматривается логический этап проектирования такой БД.

Будем исходить из постановки задачи создания БД, которая отражала бы состояние и динамику объектов экономического анализа на определенном промышленном предприятии и соответствовала бы информационным потребностям потенциальных пользователей. Применяемый нами подход к созданию БД основывается на анализе ответов на запросы, необходимые для решения задач экономического анализа. Общие положения и преимущества подобного подхода приводятся в работе [1].

На этапе логического проектирования БД ограничивается ее информационный объем на концептуальном уровне[1] и создается описание логической структуры БД. Отметим, что рассматриваемый здесь этап логического проектирования подобно работе [8] разделяется на две стадии - инфологического и даталогического проектирования.

---

[1] Под концептуальным уровнем в настоящей статье понимается уровень обобщенного описания объектов экономического анализа, информационных совокупностей и действий, выполняемых с БД.

Средством описания логических структур и содержания БД служит схема (см., например, [6]), за основу которой в данном примере взята сетевая информационная модель, сходная с моделью данных типа КОДАСИЛ (см., например, [7]). Схема состоит из диаграммы логической структуры БД и таблиц, описывающих содержание БД.

Конструирование схемы логической структуры БД рассматривается как состоящее из двух крупных задач, соответствующих стадиям логического проектирования: I) создание инфологического варианта схемы БД, ориентированного на удовлетворение всех предварительно известных видов информационных запросов (ВИЗ); 2) модификация схемы с целью придать ей более рациональный вид с точки зрения сокращения избыточности информации (сокращения дублирующих элементов схемы) и упрощения схемы для удобства дальнейшей физической структуризации совокупностей данных, а также для настройки схемы на операции обновления БД.

Инфологическое проектирование

По предварительным результатам изучения задач комплексного экономического анализа можно выделить множество основных объектов анализа, состояние, динамика и взаимодействие которых представляет интерес для потенциальных пользователей создаваемой БД. С этими объектами связаны (прямым или косвенным образом) показатели экономического анализа. Для получения систематического представления о множестве названных объектов[I] и для краткости изложения объекты сгруппированы в следующие виды: предприятие, отделы, цехи, участки, рабочие, установки, материалы, продукция.

Понятие вида объектов соответствует так называемому обобщенному (концептуальному) объекту; с другой стороны, виды рассматриваются как группы однородных объектов, имеющих одинаковый набор показателей. Данное первоначальное определение видов не должно оказаться окончательным вари-

---

[I] Проблемы выделения и группирования объектов, а также – отображения их в информационных моделях подробнее рассматривались в статье [2].

антом. Может возникнуть необходимость рассмотрения дополнительных объектов, важность выявления которых не сумели предусмотреть при предварительном изучении задач экономического анализа, или же проведенная группировка может не соответствовать требованиям отображения состояния и деятельности в БД предприятия.

Для определения информационных потребностей, которым должна удовлетворять БД, описываются основные виды предварительно известных информационных запросов (ВИЗ). Так как на данном этапе проектирования БД безразлично, как реализуются поиск и выборка информации, как обеспечиваются связи между совокупностями информации, когда и какие вычисления выполняются и т.п., описания ВИЗ даются непроцедурным образом. Непроцедурные описания ВИЗ дополняются сведениями о приблизительной частоте запросов данного вида и требованиях на оперативность их выполнения. В данной статье пример описания ВИЗ представлен в виде описания структуры и информационного содержания ответа на запрос диаграммой Джексона [5] (см. фиг. I).

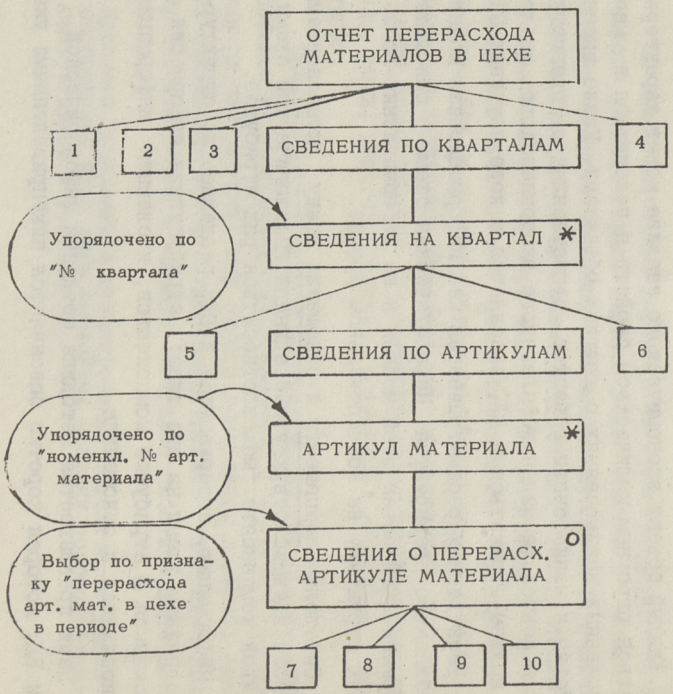
Каждый объект экономического анализа может характеризоваться огромным количеством свойств и находиться в многочисленных взаимосвязях с другими объектами. Какие именно свойства и взаимосвязи объектов, существенные в прагматическом контексте задач комплексного экономического анализа, должны отражаться в БД, выясняется в ходе изучения ВИЗ.

База данных рассматривается как информационная модель управляемого предприятия. Инфологическая схема описывает информационный объем этой модели и ее информационно-логическую структуру на концептуальном уровне.

Определим компоненты и отношения между компонентами как составляющие структурные единицы инфологической схемы. К каждой компоненте может относиться ряд атрибутов.

Информацию, содержащуюся в БД, разделяем по структурным единицам схемы на два типа: атрибутивная информация - относящаяся к атрибутам компонентов, и схемная информация - относящаяся к отношениям.

Нами предлагается методика создания инфологической схемы БД, которая коротко описывается нижеприведенными ука-



Фиг. 1. Описание ВИЗ: отчет перерасхода материалов в цехе в течение года по кварталам и артикулам материала.

Обозначения:

– составные структуры

– условия структуризации отчета и выбора информации

– элементарные структуры (элементы):

1 – название цеха,
2 – номер цеха,
3 – номер года,
4 – индекс перерасхода материалов в цехе на год,
5 – номер квартала,
6 – индекс перерасхода материалов в цехе на квартал,
7 – номенклатурный номер артикула материала,
8 – действительный расход материала в цехе в квартале,
9 – нормативный расход материала в цехе в квартале,
10 – индекс перерасхода материала в цехе в квартале.

заниями I - 6. Указания иллюстрируются примерами.

I. Каждому виду объектов экономического анализа $\mathcal{O} = \{o_i\}$ (где o_i - отдельный объект данного вида) сопоставляется компонента K инфологической схемы БД. Компонента K является информационным отражением вида $\mathcal{O}$. В информационной модели управляемого предприятия компоненте соответствует множество однородных информационных объектов (ИО)[I] $\{k_i\}$, которое далее будем называть видом информационных объектов (ВИО) (условно $K = \{k_i\}$)[2]. Каждому объекту ($o_i \in \mathcal{O}$) однозначно соответствует некоторый ИО ($k_i \in K$).

Компонентам присваиваются имена, затем дается семантическое описание соответствующего ВИО, описание количества ИО этого вида и требований к их временной стабильности[3].

Пример I. Краткое семантическое, количественное и временное описание ВИО "МАТ" приведено в следующей таблице.

Таблица I

Имя компоненты	Семантическое описание компоненты и примечания	Прибл. колич. ИО	Стабильность ИО во времени
МАТ	Компонента МАТ соответствует всем материалам и полуфабрикатам, получаемым извне данного предприятия. ИО данного вида соответствует отдельному артикулу в "Номенклатуре ...снабжения..."	около 5000	ИО соответственного вида хранятся долгосрочно. Порождение новых ИО предусмотрено в пределах 200 шт. в год, а выпадение "устаревших" ИО сериями в начале каждого года

[I] Подчеркнем, что ИО не является определенным комплектом данных, а логической совокупностью информации. Каждый ИО может найти множество различных представлений в записях данных и процедурах генерации значений атрибутов. Нахождение наиболее разумного способа представления ИО является задачей физического проектирования БД.

[2] ВИО является структурной единицей информационной модели, а компонента - структурной единицей схемы, описывающей эту модель. Каждая компонента описывает определенный ВИО. Имена компонент и ВИО условно отождествляются.

[3] Компонента логической схемы БД дается совокупностью следующих описаний: I) семантическое, количественное и временное описание ВИО (табл. I); 2) перечень и семантическое описание атрибутов компоненты (см. указание 3 и табл. 2); 3) описание места компоненты в структуре схемы (в данной статье описание дается графически на диаграммах: (фиг. 4, 5 и др.).

2. Каждому упомянутому в описаниях ВИЗ (например, фиг. I) экономическому показателю, а также каждому концептуальному количественному или качественному свойству, по которому проводится классифицирование или выборка объектов и структуризация ответа на запрос ставится в соответствие атрибут. Атрибуты именуются и описываются в виде таблицы. Каждый атрибут должен привязываться к некоторой одной компоненте (см. указание 3).

Пример 2. Атрибут будем определять по каждому элементу на диаграммах описания ВИЗ (см. фиг. I), а также по каждой величине, упомянутой в условиях структуризации ответов на запросы. Определение двух атрибутов представлено в таблице 2.

Таблица 2

Название элемента в описании ВИЗ	Имя атрибута	Описание атрибута
"Номенклатурный номер артикула материала"	MNR	MNR определяется уникальным для каждого артикула в "Номенклатуре...снабжения...". Значение MNR для каждого артикула во времени не изменяется.
"Действительный расход материала в цехе в квартал"	MKTK	MKTK является переменной величиной в течение текущего квартала, после окончания квартала значение ее останется неизменным. MKTK связан с определенным производственным цехом, определенным артикулом материала и определенным кварталом.

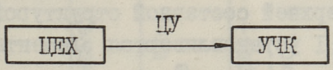
3. Компонента, к которой прикрепляется артибут, определяется по описанию атрибута (см. табл. 2). Имя атрибута вводится в перечень атрибутов определенной компоненты. Если соответствующей одной компоненты не найдется, то опре-

деляется дополнительная компонента (см. указание 5)[1].

Пример 3. Атрибут "MNR" естественно прикрепляется к исходной компоненте "МАТ", но атрибут "МКТК" логически связан одновременно с несколькими исходными компонентами, а значит, не может быть прикреплен ни к одной из них. В данном случае "МКТК" логически является атрибутом триарного отношения между ВИО "ЦЕХ", "МАТ" и "ПЕР" (см. пример 5).

4. Существенные связи прикрепленности и классификации объектов экономического анализа отображаются в информационной модели в виде отношений набора[2] между видами информационных объектов, в схеме они описываются наборами. Наборы именуются и описываются.

Пример 4. Каждый цех может включать произвольное количество производственных участков, а каждый участок обязательно принадлежит некоторому одному цеху. Определим соответственный набор между ВИО "ЦЕХ" и "УЧК", именуем набор через "ЦУ"; семантическим описанием набора "ЦУ" может служить объяснение в начале данного примера. Набор "ЦУ" представлен диаграммой (фиг. 2).



Фиг. 2.

5. Для каждого атрибута, который логически является атрибутом n-арного отношения ($n \geq 2$) между видами объектов $\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_n$ (это обстоятельство выясняется по описанию атрибута) и для которого еще не определено соответствующей компоненты, определяется дополнительная компо-

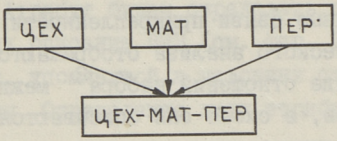
---

[1] Согласно требованию прикрепления каждого атрибута к некоторой одной компоненте, возникает потребность в новой дополнительной компоненте, соответствующей календарным периодам. Период не является объектом экономического анализа, а оказывается универсальным атрибутом всех явлений и действий, проходящих связанно с теми или иными объектами анализа. Назовем эту особую компоненту через "ПЕР" и будем считать ее условно исходной компонентой.

[2] Отношения набора (set relationships) описываются и используются, например, в работах [4,7]. Для различения понятий "set type" и "set occurrence" пользуемся терминами "набор" и "экземпляр набора" соответственно.

нента $К_m$. Она именуется[I] и описывается. Кроме того в схему вводятся n наборов, владельцами которых являются компоненты $К_1, К_2, \ldots, К_n$, соответствующие видам объектов $О_1$, $О_2, \ldots, О_n$, и общим членом которых является компонента $К_m$.

Пример 5. Для атрибута "МКТК" (см. табл. 2 и пример 3) определим компоненту, которую именуем через "ЦЕХ-МАТ-ПЕР" и введем три набора под именами "ЦПМ", "МЦП" и "ПЦМ" (см. фиг. 3).



Фиг. 3.

6. Для каждого ВИЗ определяется коренная компонента. Обычно это удается интуитивно по вербальному описанию вида запроса. Для более сложных ВИЗ коренная компонента может определяться следующими действиями (I-4).

I) На диаграмме описания ВИЗ устанавливаются элементы и величины структуризации отчета, которые непосредственно связаны с верхней составной структурой. Например, по диаграмме на фиг. I устанавливаются элементы 1,2,3,4.

2) По описаниям компонент устанавливаются компоненты, к которым прикреплены атрибуты, соответствующие элементам и величинам, установленные действием I). Например, по таблице 3 устанавливаются компоненты "ЦЕХ", "ПЕР", "ЦЕХ-ПЕР".

3) Из установленных компонент (если их будет больше одной) выбираются компоненты, соответствующие видам основных объектов экономического анализа (табл. 3). В данном случае, например, такой компонентой будет "ЦЕХ".

[I] Зачастую дополнительным компонентам можно присвоить имена по их семантическим значениям, т.е. имена, соответствующие определенным связям (действиям или ролям), соединяющим объекты экономического анализа. Но данная методика предусматривает образование (составных) имен дополнительных компонент формальным соединением простых имен соответствующих исходных компонент $К_1, К_2, \ldots, К_n$ (для наглядности простые имена соединяются через черточки).

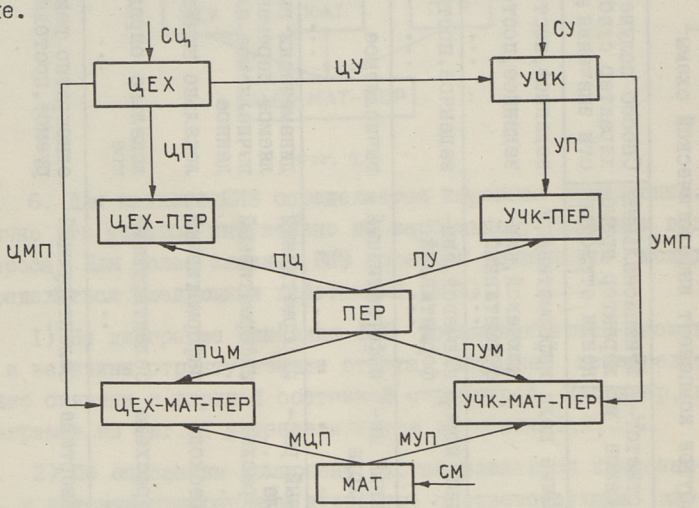
Таблица 3

Перечень и описание атрибутов компонент инфологической схемы[I]

Тип компонент	Имена компонент	Имена атрибутов	Названия соответствующих признаков и показателей	Функциональный характер применения атрибута	Способ получения и характер стабильности значения атрибута	Примечания
Исходные	ЦЕХ	TN	полное название цеха	информативный	заданное, постоянное	
		TNO	номер цеха	ключевой, информативный	заданное, постоянное	
		...	...	...	...	
	МАТ	MNR	номенклатурный номер материала	ключевой, информативный	заданное, постоянное	
		...	...	...	...	
Дополнительные	ЦЕХ-ПЕР	MKID	комбинированный индекс перерасхода материалов	информативный	вычисляемое	Вычисление должно блокироваться при несоответствии актуальности значений атрибутов MKTK и MKNK
		...	...	...	...	
	ЦЕХ-МАТ-ПЕР	MKTK	действительный расход материала ...	информативный	динамически вычисляемое переменное	
		MKNK	нормативный расход материала ...	информативный	вычисляемое или заданное	
		MKI	индекс перерасхода материала	информативный	локально вычисляемое	
		MKT	признак перерасхода материала	селектирующий	локально определяемое	
	...	...	...	...	...	
	ПЕР	PNO	номер года/квартала	селектирующий	однократно генерируемое, постоянное	

[I] Данная таблица включает лишь компоненты и атрибуты, соответствующие ВИЗ (фиг. I), также опущена графа с формулами вычисления значений атрибутов.

4) Если же выбранных компонент больше чем одна, то среди них найдется компонента с минимальным количеством соответствующих ей ИО. Для каждой новой коренной компоненты, ранее не определенной как коренная для некоторого другого ВИЗ, определяется системный набор R_S, в котором данная компонента выступает в качестве члена, а владельцем становится "система" (см. например,[7]). На диаграммах (фиг. 4,5,6 и 7) системные наборы обозначены стрелками извне.



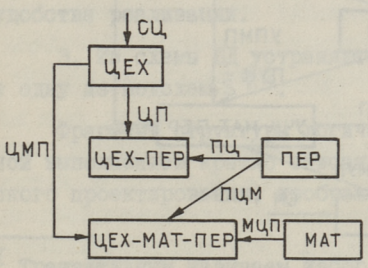
Фиг. 4.

Проанализировав все описания ВИЗ, следуя указаниям I-6, получим инфологическую схему, которая может рассматриваться описанием инфологической модели предприятия, содержащей весь объем сведений для задач экономического анализа (предполагается, что данная совокупность описаний ВИЗ описывает весь круг информационных потребностей). Фрагмент структуры инфологической схемы БД, полученный посредством анализа пяти различных ВИЗ, изображен на фиг. 4. Однако информационная модель должна быть не только адекватной информационным потребностям в содержательном смысле, но и обеспечить удовлетворение функциональных требований к БД. В качестве последних можно в первую очередь выделить требования на оперативность удовлетворения запросов, экономии

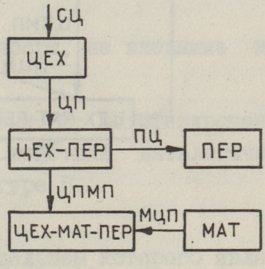
ресурсов памяти и сокращения общей нагрузки процесса ЭВМ. Окончательно решаются эти вопросы учитывая возможности конкретной СУБД и ЭВМ, но целый ряд принципиальных проблем такого рода можно решать еще и на этапе логического проектирования БД.

Даталогическое проектирование

Определим для каждого ВИЗ некоторую подсхему $S^q = (\mathcal{K}^q, R^q)$ следующим образом. Пусть множество компонент - $\mathcal{K}^q$ подсхемы S^q состоит из компонент, атрибуты которых соответствуют величинам, упомянутым (в виде элементов и величин структуризации отчета) в описании ВИЗ. И множество отношений - R^q состоит из всех наборов схемы, связывающих компоненты $\mathcal{K}_i \in \mathcal{K}^q$ между собой, и дополнительно системного набора R_s для данного ВИЗ. Структура подсхемы S^q, соответствующей ВИЗ (фиг. I), изображена на диаграмме фиг. 5.



Фиг. 5.



Фиг. 6.

Даже поверхностным изучением структуры подсхемы (фиг.5) можно установить, что логическая процедура, реализующая описанную ВИЗ на данной схеме БД, получается неэффективной. Например, удовлетворение запроса (фиг. I) требует просмотра огромного количества ИО "ЦЕХ-МАТ-ПЕР", не имеющих никакого отношения к конкретному запросу.

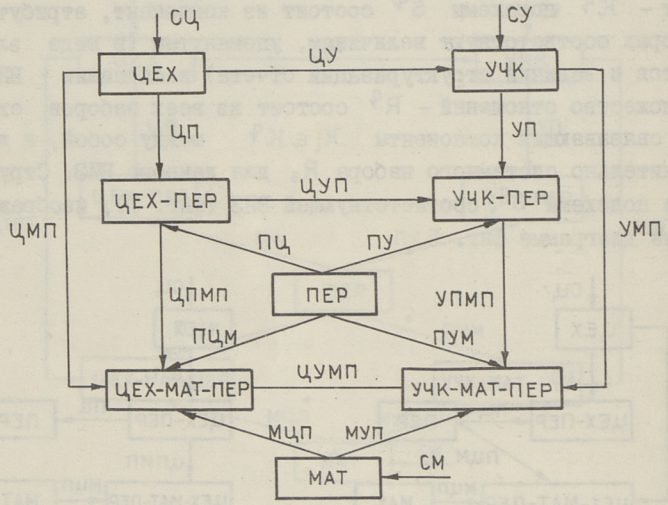
Существенная рационализация схемы достигается следующими формальными преобразованиями.

I. Пусть дано множество простых имен $N = \{n_j\}$ исходных компонент. Составное имя N_i дополнительной компоненты $\mathcal{K}_i$ состоит из множества $N_i = \{n_j \mid n_j \in N\}$. Опреде-

лим дополнительные отношения в схеме следующим образом:

I. Если $N_u \subset N_v$, то в схему вводится новое отношение набора, владельцем которого становится компонента под именем N_u (коротко "N_u"), а членом - "N_v".

Пример. Фрагмент структуры схемы (фиг. 4) дополняется наборами "ЦПМП" и "ОПМП" (см. фиг. 6), так как $\{ЦЕХ, ПЕР\} \subset \{ЦЕХ, МАТ, ПЕР\}$ и $\{УЧК, ПЕР\} \subset \{УЧК, МАТ, ПЕР\}$.



Фиг. 7.

2. Если удовлетворены следующие условия а,б,в:

а) определен набор, владельцем которого является "n_s" ($n_s \in N$), а членом - "n_t" ($n_t \in N$);

б) определены два набора, владельцами которых являются компоненты "n_s" и "n_t", а членами - "N_u" и "N_v" соответственно;

в) $N_u \setminus n_s = N_v \setminus n_t$,

то в схему вводится набор, владельцем которого становится компонента "N_u", а членом "N_v".

Пример. Фрагмент структуры схемы (фиг. 4) дополняется набором "ЦУП", так как

а1) определен набор "ЦУ",

б1) определены наборы "ЦП" и "УП",

в1) {ЦЕХ,ПЕР}\{ЦЕХ}={УЧК,ПЕР}\{УЧК},

а также набором "ЦУМП", так как

б2) определены наборы "ЦМП" и "УМП",

в2) {ЦЕХ,МАТ,ПЕР}\{ЦЕХ}={УЧК,МАТ,ПЕР}\{УЧК}.

Дополненный фрагмент схемы изображен на фиг. 7.

2. Из полученной дополнительной схемы выделим для каждого ВИЗ (аналогично S^{q}) дополненную подсхему $\overset{*}{S}^{q}$. Из подсхемы $\overset{*}{S}^{q}$ устраняются все транзитивные наборы[I]. Полученную таким образом подсхему обозначим через $\bar{S}^{q}$. Структура подсхемы $\bar{S}^{q}$ для ВИЗ (фиг. I) изображена на диаграмме фиг. 6.

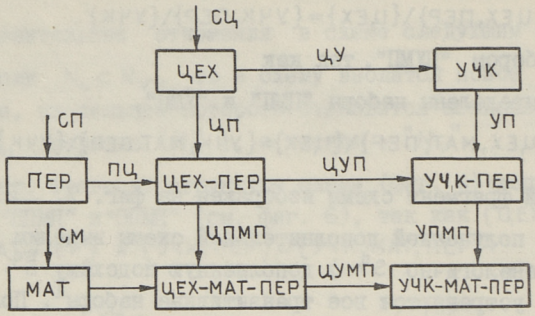
Отметим, что наличие транзитивных наборов является лишь частичным случаем избыточности схемной информации (возможны разные циклы, параллельные наборы и т.п.). В других случаях требуется решение ситуации в отношении с конкретными критериями экономии ресурсов памяти или времени или же удобства реализации.

3. Из схемы БД устраняются все наборы, не входящие ни в одну из подсхем $\bar{S}^{q}$[2].

Фрагмент структуры логической схемы БД, модифицированной выполнением кратко описанных трех действий даталогического проектирования, изображен на фигуре 8.

[I] Транзитивным называем набор R_i, владельцем которого является компонента K_1, а членом – компонента K_n, если определены наборы $R_1,\dots,R_{n-1}$, владельцами которых являются компоненты $K_1,\dots,K_{n-1}$, а членами $K_2,\dots,K_n$ соответственно.

[2] При этом, для настройки схемы на виды операций обновления БД (ВИО), учитываются и подсхемы, соответствующие всем ВИО. Рассмотрение проектирования ВИО не вмещается в рамки данной статьи, отметим, только, что оно логически проводится от инфологической схемы БД; методика описания ВИО и определение соответствующих подсхем сходна с методикой относительно ВИЗ. Как правило, дополнительных компонент и атрибутов по описаниям ВИО БД не определяется. Эти операции должны обеспечиваться процедурами предварительной обработки вводимых данных. Без специальных процедур часто невозможно обеспечить непротиворечивость вводимых данных и привести их в соответствие с атрибутами компонент. По мере необходимости обновление БД может быть предусмотрено с применением специального промежуточного файла (файлов).



Фиг. 8.

Значительное место при модификации логической схемы БД занимает организация атрибутивной информации. Суть рациональной организации заключается в том, как в каждом конкретном случае будет решаться проблема выбора между хранением значений атрибутов в явном виде (при этом увеличивается объем памяти) или же вычислением этих значений во время удовлетворения запросов (при этом теряем во времени). Решение таких вопросов сопровождается логическим проектированием процедурного обеспечения БД.

По связям между объектами экономического анализа и по функциональным связям между экономическими показателями можно формально определить правила получения значений одних показателей от значений других. Однако среди большого количества показателей, характеризующих одни и те же объекты, не всегда можно однозначно определить, какие следует считать исходными и какие вычисляемыми.

Пусть задано некоторое множество процедур генерации (вычисления) $G = \{g_i\}$ значений атрибутов, соответствующих определенным ИО. Исходя из отношений между множеством и множеством атрибутов инфологической схемы, можно атрибуты разделить на две группы: фундаментальные и производные атрибуты, (формальные определения этих понятий даются ниже).

Обозначим множество атрибутов компоненты $\mathcal{K}_i$ ($i=1,2,\ldots,r$, где r – количество компонент инфологической схемы) через $A^i = \{a_{i1}, \ldots, a_{i\delta_i}\}$ (где δ_i – количество атрибутов компоненты $\mathcal{K}_i$), а множество всех атрибутов схемы через A ($A = \bigcup_{i=1}^{r} A^i$).

Экземпляры ИО, входящие в ВИО K_i, обозначим через $k_{i_1}, k_{i_2}, \ldots, k_{iq_i}$ (где q_i – количество экземпляров ИО вида K_i), а значения атрибута a_{ij} для k_{in} – через v_{ijn} ($i=1,\ldots,r$; $j=1,\ldots,s_i$; $n=1,\ldots,q_i$).

Определим множество атрибутов A_G ($A_G \subset A$), на основе значений которых при помощи некоторой комбинации процедур $\{g_1,\ldots,g_p\} \subset G$ удается получить значения любого атрибута ($a_{ij} \in A^i$) v_{ijn} для любого ИО $k_{in} \in K_i$.

Определим множество $\bar{A}_G$ ($\bar{A}_G \subseteq A_G$), которое минимально в том смысле, что не найдется такого атрибута $a_f \in \bar{A}_G$, при исключении всех значений которого из БД, множество $\bar{A}_G$ также будет соответствовать определению множества A_G.

Атрибуты $a_f \in \bar{A}_G$ называются фундаментальными, а атрибуты $a_d \in (A \setminus \bar{A}_G)$ – производными.

Производный атрибут a_{ij} компоненты K_i, значение которого v_{ijn} для каждого k_{in} генерируемо с помощью некоторой комбинации $\{g_1,\ldots,g_p\} \subset G$ без привлечения значений других атрибутов, кроме фундаментальных атрибутов того же самого k_{in}, называется локально производным.

Хранение в БД значений производных атрибутов трактуется нами как избыточность информации, также как и дублирование схемной информации в атрибутивном виде. Избыточность может быть оправдана только необходимостью выяснения противоречивости данных и обоснованными временными суждениями относительно выполнения определенных операций с БД.

Как правило, в БД не хранятся значения локально производных атрибутов. Что касается других производных атрибутов, то этот вопрос решается отдельно для каждого случая, учитывая:

I) прогнозированное время, затрачиваемое на генерацию значения атрибута в режиме удовлетворения запроса и в режиме обновления данных, влияющих на значение данного атрибута.[I]

[I] Предварительные прогнозы на время выполнения операций БД делаются опытными специалистами по проектированию и эксплуатации БД на основе структуры даталогической схемы БД, описаний логических процедур генерации значений атрибутов и опыта эксплуатации конкретной СУБД.

2) прогнозированную частоту запросов на значение данного атрибута[1];

3) прогнозированную частоту операций обновления, влияющих на значение данного атрибута[2];

4) требования на оперативность удовлетворения запросов, относящихся к данному атрибуту.

5) требования на логическую проверку вводимых в БД данных.

В даталогической схеме останутся только те атрибуты инфологической схемы, хранение значений которых предусматривается в явном виде. Таким образом, некоторые компоненты даталогической схемы оказываются без атрибутов. Такие компоненты часто устраняются из схемы БД. Рассмотрение условий на устранение компонент и вытекающих изменений во множестве наборов схемы и в описаниях ВИЗ и ВИО не вмещается в рамки данной статьи.

Даталогический проект БД в принципе должен включать следующие описания:

I) структуры единой даталогической схемы БД (в виде диаграммы);

2) информационного содержания БД - описания компонент, атрибутов[3] и наборов;

3) процедур генерации значений производных атрибутов;

4) ВИЗ и ВИО с определением соответствующей для каждого из них подсхемы БД.

Заключение

В статье рассматривалась возможная методика логического проектирования БД для задач экономического анализа деятельности промышленного предприятия. В целях интеграции обработки данных до последующей реализации этой БД предстоит

---

[1] Прогноз получается формально по описаниям ВИЗ.

[2] Прогноз получается по описаниям ВИО и процедур генерации.

[3] В описаниях производных атрибутов кроме их семантики и "активности" указываются процедуры генерации их значений.

еще "переплетение" полученной схемы с существующими (или одновременно проектируемыми) схемами других функциональных частей системы обработки данных (СОД) предприятия. Отметим, что присоединение функциональных подсистем к реализованной СОД можно проводить различными способами. Если СОД основана на интегрированной БД, система управления которой дает возможность частичного дополнения и перестроктуризации логических структур данных, а технические условия позволяют увеличение объема БД, то полученный вариант схемы можно рассматривать основой проекта изменений БД. Если же существующая СОД не отвечает названным общим требованиям, то предлагается создать отдельную БД для задач экономического анализа, где обновление данных должно предусматриваться непосредственно из существующей БД(из системы файлов) СОД - без дублирования ввода данных.

Полученный вариант схемы можно рассматривать как результат логического проектирования БД. Далее надо довести до конца проектирование структур хранения данных, в том числе решить следующие принципиальные вопросы:

I) выбор средств реализации БД - СУБД;

2) принципиальное совмещение структуры логической схемы БД с возможностями конкретной СУБД;

3) окончательная организация атрибутивной информации с одновременным проектированием процедур генерации значений производных атрибутов.

Описанная в данной статье методика построения и модификации инфологической схемы БД, принципиально пригодна для логического проектирования различных баз данных.

Литература

I. Жеребин В.М., Мальцев В.Н., Совалов М.С. Экономические информационные системы: Рационализация проектирования. М., Наука, 1978.

2. Лаast - Лаас Ю.Г. Об инфологическом подходе к базам данных. - Тр. Таллинск. политехн. ин-та, № 464, 1979, с. 39-57.

3. Шеремет А.Д. Комплексный экономический анализ деятельности предприятия (вопросы методологии). М., Экономика, 1974.

4. Bubenko, J.A., Berild, S., Lindencrona-Ohlin, E., Nachmens, S. Information analysis and design of data base schemata. - CADIS Report, TRITA-BADB-3091, 1975.

5. Jackson, M.A. Principles of Program Design. London, Acad. Press, 1975.

6. Nijssen, G.M. A gross architecture for the next generation database management systems. - Proc. of the IFIP Working Conference on Modelling in Data Base Management Systems. North-Holland Publ. Co., 1978.

7. Olle T.W. The CODASYL Approach to Data Base Management. New York, Wiley, 1978.

8. Sundgren B. An Infological Approach to Data Bases (URVAL NO.7) National Central Bureau of Statistics, Stockholm., 1973.

J. Laast-Laas

On Schema-Design for an Economical Analysis Application Data Base

Summary

An infological schema-design process for an economical analysis application data base is described. In the first part the process of user's information requirements definition, the information structure determination and a query-oriented schema alternative creation is treated. In the second part some suggestions for improving the schema to get a feasible data structure for a later implementation are made.

№ 482
TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 519.21

Т.В. Арак

ОБ АППРОКСИМАЦИИ n-КРАТНЫХ СВЕРТОК РАСПРЕДЕЛЕНИЙ, ИМЕЮЩИХ НЕОТРИЦАТЕЛЬНУЮ ХАРАКТЕРИСТИЧЕСКУЮ ФУНКЦИЮ, БЕЗГРАНИЧНО ДЕЛИМЫМИ РАСПРЕДЕЛЕНИЯМИ

В данной статье будут рассмотрены вероятностные распределения, заданные на борелевских подмножествах вещественной прямой. Произведения и степени мер будем понимать в смысле свертки. Пусть $\tilde{f}_+$ – совокупность всех таких распределений, характеристическая функция которых неотрицательна на всей оси, и пусть $\mathfrak{O}$ – множество всех безгранично делимых распределений. Обозначим через ρ равномерное расстояние в пространстве распределений

$$\rho(F,G)=\sup_x|F((-\infty,x))-G((-\infty,x))|.$$

Рассмотрим величины

$$\psi_n=\sup_{F\in\tilde{f}_+}\rho(F^n,\mathfrak{O}) \qquad n=1,2,\dots$$

В работе [1] была доказана оценка

$$\psi_n\le\frac{C_1}{n}. \tag{1}$$

Здесь и в дальнейшем буква C с индексом служит для обозначения абсолютных постоянных.

В настоящей заметке мы докажем, что

$$\psi_n\ge\frac{C_2}{n}, \tag{2}$$

где $C_2>0$.

Тем самым будет установлено, что (1) является наилучшей по порядку оценкой величины ψ_n. (К моменту написания работы [1] наилучшей известной оценкой снизу для ψ_n была $C_3/(n\ln^2 n)$. Эта оценка следует из одного результата Ю.В. Прохорова [2] и получается рассмотрением свертки симметричного двухточечного распределения с самим собой).

При построении оценки снизу величины ψ_n будем пользоваться распределениями F_α $(0<\alpha\leq\frac{1}{4})$, определяемыми следующим образом: распределение F_α сосредоточено в точках $-1, 0, 1$ и имеет в этих точках массы α, $1-2\alpha$, α соответственно. Отсюда следует, что $F_\alpha \in \tilde{\mathcal{F}}_+$ при $0<\alpha\leq\frac{1}{4}$. Требуемая оценка вытекает из нижеследующих двух лемм, в которых $\mathfrak{A}$ обозначает совокупность вырожденных распределений, E – распределение с единичной массой в начале координат, $\Gamma(G,0) = \max\limits_{-\infty<x<\infty} G(\{x\})$.

<u>Лемма I</u>. Существуют такие положительные постоянные C_4 и C_5, что при любом натуральном n из предположения

$$\rho(F,\mathfrak{A}) \leq \frac{C_4}{n} \tag{3}$$

следует выполнение неравенства

$$\rho(F^n, G^n) \geq C_5 \min\{1, n\rho(F,G)\}, \tag{4}$$

каково бы ни было распределение G.

Д о к а з а т е л ь с т в о. Подходящим сдвигом начала отсчета можем условие (3) свести к следующему:

$$\rho(F,E) \leq \frac{C_4}{n}. \tag{3'}$$

Будем считать, что $C_4 \leq \frac{1}{8}$. Тогда из $(3')$ следует,

$$1-F(\{0\}) \leq \frac{1}{4n}.$$

Рассмотрим сперва случай, когда $1-G(\{0\}) \leq 1/(3n)$. В этом случае, очевидно, существуют такие распределения F' и G', что $F = qE+pF'$ и $G = qE+pG'$, где $p = 1/(3n)$ и $q = 1-p$. Из равенств

$$F^n = \sum_{k=0}^{n} \binom{n}{k} q^{n-k} p^k (F')^k, \quad G^n = \sum_{k=0}^{n} \binom{n}{k} q^{n-k} p^k (G')^k$$

следует, что

$$\rho(F^n, G^n) \geq \binom{n}{1} q^{n-1} p\rho(F',G') - \sum_{k=2}^{n} \binom{n}{k} q^{n-k} p^k \rho((F')^k, (G')^k).$$

Применяя неравенство $\rho((F')^k,(G')^k) \leq k\rho(F',G')$ (см. например, [4]), получаем

$$\rho(F^n,G^n) \geq \binom{n}{1} q^{n-1} p\rho(F',G') - \sum_{k=2}^{n} q^{n-k} p^k k\rho(F',G') =$$

$$= np(2q^{n-1}-1)\rho(F',G') \geq \frac{1}{3} np\rho(F',G') = \frac{1}{3} n\rho(F,G).$$

Рассмотрим случай

$$1-G(\{0\}) \geqslant \frac{1}{3n}. \tag{5}$$

В этом случае достаточно доказать, что

$$1-G^n(\{0\}) \geqslant C_6 > 0. \tag{6}$$

Действительно, из (3') следует, что

$$F^n(\{0\}) \geqslant (1-\frac{C_4}{n})^n \geqslant 1-C_4.$$

Отсюда и из (6) в свою очередь находим, что

$$\rho(F^n, G^n) > C_6 - C_4 > 0, \tag{7}$$

если C_4 выбрано достаточно малым. Далее, из (5) заключаем, что

$$\rho(F,G) \geqslant \rho(G,E) - \rho(F,E) \geqslant \frac{1}{6n} - \frac{1}{8n} = \frac{1}{24n}.$$

Из этого неравенства и (7) вытекает (4).

Итак, будем доказывать, что из (5) следует (6). Начнем с применения неравенства

$$G^n(\{0\}) \leqslant \max\left\{C_7/\sqrt{n(1-\Gamma(G,0))},\ \Gamma(G,0)\right\}.$$

(См. теорему 4 из [3], гл. III). Отсюда вытекает (6), если, например, $1-\Gamma(G,0) \geqslant \min\{\frac{1}{2},\ 4C_7^2/n\} = \delta_n$. Пусть теперь $1-\Gamma(G,0) \leqslant \delta_n$ и $1-G(\{0\}) > \delta_n$. Тогда, очевидно, найдется такое число $a \neq 0$, что $G(\{a\}) = 1-\Gamma(G,0)$ и поэтому

$$1-G^n(\{0\}) \geqslant G^n(\{na\}) \geqslant (G(\{a\}))^n \geqslant (1-\delta_n)^n \geqslant C_8 > 0.$$

Пусть, наконец, $1/(3n) \leqslant 1-G(\{0\}) \leqslant \delta_n$. Тогда мы имеем

$$1-G^n(\{0\}) \geqslant n\,[1-G(\{0\})]\cdot[G(\{0\})]^{n-1} \geqslant$$

$$\geqslant \inf_{\frac{1}{3} \leqslant x \leqslant n\delta_n} x(1-\frac{x}{n})^{n-1} \geqslant C_9 > 0.$$

Этим лемма доказана.

Лемма 2. Существует такая положительная постоянная C_{10}, что

$$\rho(F_\alpha, O_\delta) \geqslant C_{10}\alpha^2 \tag{8}$$

при любом α из интервала $0 < \alpha \leqslant \frac{1}{4}$.

Доказательство. Так как при любом $\varepsilon > 0$ множества O_δ и $\{F_\alpha: \varepsilon \leqslant \alpha \leqslant \frac{1}{4}\}$ компактны в метрике ρ и не пересекаются друг с другом, то расстояние между ними положительно. Поэтому требуется лишь доказать, что неравенство (8) имеет место при всех достаточно малых α. Более того, будет достаточно, если мы убедимся в справедливости

неравенства $\rho(F_\alpha, G) \geq C_{10}\alpha^2$

для любого распределения G, имеющего следующий вид

$$G = \exp\{\lambda(L-E)\}E_a = \sum_{k=0}^{\infty} e^{-\lambda}\frac{\lambda^k L^k}{k!}E_a, \qquad (9)$$

где L – произвольное распределение, не имеющее атома в точке 0;

E_a – единичная масса в точке a $(-\infty < a < \infty)$ и $\lambda > 0$. (Действительно, распределения вышеописанного типа образуют всюду плотное в $\mathcal{O}_f$ множество).

Предположим, что для некоторого распределения G вида (9)

$$\rho(F_\alpha, G) \leq \alpha^2. \qquad (10)$$

Очевидно, $\Gamma(G,0) \geq \Gamma(F_\alpha, 0) - 2\rho(F_\alpha, G) \geq 1 - 2\alpha - 2\alpha^2 \geq \frac{1}{4}$.

С другой стороны, $\Gamma(G,0) \leq C_{11}\lambda^{-\frac{1}{2}}$.

(См. теорему I из [3], гл. III). Таким образом,

$$\lambda \leq C_{11}^2\Gamma^{-2}(G,0) \leq 16C_{11}^2 = C_{12}. \qquad (11)$$

Применяя неравенство треугольника, находим, что

$$\rho(G, \mathcal{O}_f) \leq \rho(G, F_\alpha) + \rho(F_\alpha, \mathcal{O}_f) \leq \alpha^2 + \alpha \leq \frac{5}{4}\alpha. \qquad (12)$$

Выясняется также, что

$$\rho(G, \mathcal{O}_f) \geq \min\{\rho(G, E_a),\ \rho(G, \mathcal{O}_f \setminus \{E_a\})\} \geq$$
$$\geq \min\{e^{-\lambda},\ \lambda e^{-\lambda}\}. \qquad (13)$$

Из (11)–(13) получаем:

$$\lambda \leq C_{13}\alpha. \qquad (14)$$

В силу (10) и (14)

$$\rho(F_\alpha, e^{-\lambda}(E+\lambda L)E_a) \leq \rho(F_\alpha, G) + \sum_{k=2}^{\infty} e^{-\lambda}\frac{\lambda^k}{k!} \leq \alpha^2 + \lambda^2 \leq C_{14}\alpha^2.$$

Отсюда следует, что мера $e^{-\lambda}\lambda L$ имеет атомы в точках $1-a$ и $-1-a$ и массы этих атомов больше или равны $\alpha - 2C_{14}\alpha^2$. Предположим теперь еще, что $\alpha \leq 1/(4C_{14})$. Тогда $\alpha - 2C_{14}\alpha^2 \geq \alpha/2$ и мера $e^{-\lambda}\lambda^2 L^2$ имеет атомы в точках $2-2a$ и $-2-2a$, массы которых больше или равны $\alpha^2/4$. Из этого факта заключаем, что

$$\rho(F_\alpha, G) \geq \alpha^2/8. \qquad (15)$$

Таким образом, нами показано, что в предположении (10) при достаточно малом α имеет место (15). Согласно вышесказанному, этим лемма доказана.

Чтобы доказать неравенство (2), рассмотрим распределения F_α, полагая $\alpha = C_4/n$. В силу лемм I и 2 имеем

$$\psi_n \geq \inf_{G \in \mathcal{G}} \rho(F_p^n, G^n) \geq C_5 \min\{1, nC_{10}\alpha^2\} \geq C_{15}/n.$$

Этим неравенство (2) доказано.

Литература

I. Арак Т.В. О сближении n-кратных сверток распределений, имеющих неотрицательную характеристическую функцию, с сопровождающими законами. – Теория вероят. и ее примен., XXУ, 2 (I980), 225-246.

2. Прохоров Ю.В. О равномерной предельной теореме А.Н. Колмогорова. – Теория вероят. и ее примен., У, I (I960), I03-II3.

3. Петров В.В. Суммы независимых случайных величин. М., Наука, I972. 4I4 с.

4. Золотарев В.М. Метрические расстояния в пространствах случайных величин и их распределений. -Матем. сб., I0I (I43), 3 (I976), I04-I4I.

T. Arak

On the Approximation of n-fold Convolutions of Distributions Having Nonnegative Characteristic Functions by Infinitely Divisible Distributions

Summary

It is shown that an estimate obtained in [1] cannot be improved.

№ 482

TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 658.5.012.122.001.57.681

Х.Ф. Симанова

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ ПОТОЧНОГО ПРОИЗВОДСТВА НА БАНКЕ ДАННЫХ

1. Введение

Одной из задач управления производственной системой является составление прогнозов эффективности материальных затрат. Получение подобных прогнозов позволяет управленческому персоналу проводить анализ деятельности своей производственной системы. В результате анализа могут быть выявлены узкие места, действия которых можно ослабить либо ликвидировать вообще своевременными организационными мероприятиями.

В статье предпринята попытка разработки на ЭВМ некоторых средств для получения прогнозов функционирования анализируемой экономической системы. Такой системой может быть одно предприятие, совокупность предприятий министерства и т.п. В иллюстративном примере, приведенном в разделе 2, за экономическую систему принимается одно предприятие.

В теоретическом аспекте предлагаемые средства получения прогнозов опираются на представление сетевой структуры экономической системы взвешенным графом, который, в свою очередь, служит основой создания базы данных для расчета показателей деятельности экономической системы.

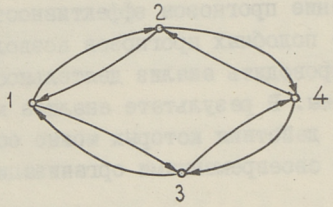
2. Расчетная модель для частного случая поточного производства

В данном разделе разрабатывается модель экономической системы на конкретном примере.

Рассмотрим экономическую систему, потребляющую материальные ресурсы и выпускающую готовую продукцию. Разобьем

такую систему на три класса ячеек, которые условно назовем: I) ресурсными, 2) производственными и 3) выпускающими. Эти ячейки связаны друг с другом двумя типами отношений: отношением спроса и отношением предложения.

С иллюстративной целью рассмотрим конкретную систему, состоящую из одной ресурсной (I), двух производственных (2, и 3) и одной выпускающей (4) ячеек. В результате разбиения образуется модель (см. фиг. I), в которой ресурсная ячейка с производственными и производственные с выпускающей связаны отношением предложения; в обратном направлении, выпускающая ячейка с производственными и производственные с ресурсной - отношением спроса.



Фиг. 1.

Выделив пару ячеек, связанных отношением спроса и предложения, ту, от которой исходит отношение спроса, будем называть заказчиком, а ячейку, от которой исходит отношение предложения - поставщиком. Таким образом, ресурсная ячейка является только поставщиком, выпускающая - только заказчиком, а производственные являются как поставщиками, так и заказчиками одновременно.

Обычно в принятой номенклатуре показателей, отражающих деятельность экономической системы, производится их подразделение. Так, основным показателем является объем реализации товарной продукции,[I] который имеет три разновидности:

[I] Объем реализации продукции определяется как стоимость готовых изделий и полуфабрикатов, предназначенных к поставке потребителям и подлежащих оплате последними, исчисляется в действующих оптовых ценах предприятия. Фактический объем - это уже оплаченные поставки продукции, [4].

I) план по объему реализации товарной продукции (d, η);

2) объем реализации товарной продукции (δ, ξ);

3) фактический объем реализованной товарной продукции (γ).

Следует заметить, что в то время, как план реализации представляет собой производственное задание, а объем реализации – имеющуюся в наличии и подлежащую реализации товарную продукцию, фактический объем реализации выступает измерителем товарной продукции в сделке между поставщиком и заказчиком. Следовательно, фактический объем реализации (γ) характеризует отношения спроса и предложения между поставщиком и заказчиком, а показатели плана (d) и объема (δ) товарной продукции характеризуют или поставщика или заказчика. План по объему реализации товарной продукции выпускающей ячейки зададим в размере d рублей, а объем реализации товарной продукции ресурсной ячейки – в размере δ рублей.

Предположим теперь, что процессы, описываемые моделью, являются линейными, т.е. с увеличением потребления в той же пропорции увеличивается выпуск. При этом производственный процесс будем считать поточным и некомплектным, так что момент выпуска готовой продукции производственной ячейки на уровне модели можно сдвинуть непосредственно в отношение между поставщиком и заказчиком, в результате чего отношения спроса и предложения для конкретной пары ячеек характеризуются показателем материальных затрат на рубль товарной продукции. Кроме того, предположим, что имеются данные о пропускной способности, определяемой как максимум производимой производственной ячейкой товарной продукции в денежном выражении.

Каждая ячейка экономической системы и каждое отношение между ячейками характеризуется показателями. Все показатели будем исчислять в денежном выражении. Показатели, характеризующие ячейки модели, будем помечать индексом: соответственно для ресурсной – I, для производственной – 2 или 3, для выпускающей – 4. Показатели, характеризующие отношения, будем помечать двумя индексами, причем первый индекс будет указывать на поставщика, а второй – на заказчика.

Рассмотрим процесс расчетов основных показателей на модели (фиг. I). Для того, чтобы выполнить свой план, выпускающая ячейка должна представить материальные заявки[I], т.е. задать план производственным ячейкам. Выпускающая ячейка определяет величину заявки для каждой производственной по своему усмотрению. Следовательно, отношение спроса можно охарактеризовать показателем относительной доли заявок (β). Из вышесказанного вытекает, что показатель β надо пометить двумя индексами. Например, β_{24} – характеризует отношение спроса между производственной ячейкой с индексом 2 и выпускающей ячейкой с индексом 4. Аналогично, отношение предложения характеризуется показателем относительной доли предложения (α), который также будем помечать двумя индексами.

Пусть M_2 и M_3 – пропускные способности производственных ячеек, ν_{wj} – материальные затраты на рубль товарной продукции каждой заявки, где индекс w – поставщик, а j – заказчик.

Итак, выпускающая ячейка представит заявки производственным ячейкам 2 и 3 на сумму $\eta_{24} = d \cdot \beta_{24}/\nu_{24}$ и $\eta_{34} = d \cdot \beta_{34}/\nu_{34}$ соответственно. Каждая производственная ячейка представит в свою очередь ресурсной ячейке заявку на сумму $\eta_{12} = \eta_{24} \cdot \beta_{12}/\nu_{12}$ и $\eta_{13} = \eta_{24} \cdot \beta_{13}/\nu_{13}$. Таким образом, ресурсная ячейка получит заявки на сумму $d_1 = \eta_{12} + \eta_{13}$.

Предположим, что все заявки выполняются по возможности соразмерно величине имеющейся товарной продукции, а излишки (если они образуются) реализуются вне системы. Следовательно, ресурсная ячейка выполнит заявки производственных ячеек в размере $\gamma_{12} = \min(\vartheta \cdot \alpha_{12}, \eta_{12})$ и $\gamma_{13} = \min(\vartheta \cdot \alpha_{13}, \eta_{13})$. Теперь можно вычислить возможный объем реализации товарной продукции ξ с учетом материальных затрат по каждой заявке, т.е. $\xi_{12} = \gamma_{12}/\nu_{12}$ и $\xi_{13} = \gamma_{13}/\nu_{13}$. Далее, величина объема реализации товарной продукции производственных ячеек ставится в зависимость от пропускной способности:

[I] Заявка материальная – документ на потребные предприятию в предстоящем плановом периоде материалы, [4].

$$\xi_2 = \min\{M_2, S_2\},\ \xi_3 = \min\{M_3, S_3\}.$$

Для характеристики излишков, образующихся вследствие недостаточной пропускной способности, вводится показатель перегрузки (θ) производственной ячейки. Показателем θ измеряется стоимость недовыпущенной товарной продукции $\theta_2 = \xi_2 - \sigma_2$ и $\theta_3 = \xi_3 - \sigma_3$.

Далее, заявки выпускающей ячейки будут выполнены производственными ячейками в размере $\gamma_{24} = \min(\sigma_2 \cdot \alpha_{24}, \eta_{24})$ и $\gamma_{34} = \min(\sigma_3 \cdot \alpha_{24}, \eta_{34})$ соответственно. Таким образом, объем реализации товарной продукции выпускающей ячейки будет равен сумме $\sigma_4 = \gamma_{24} / \nu_{24} + \gamma_{34} / \nu_{34}$.

Тем самым закончены вычисления главных показателей модели - плана и объема реализации товарной продукции.

Кроме того, имеется возможность вычислить дополнительные показатели, довольно хорошо характеризующие деятельность предприятия.

Перечислим эти показатели и приведем формулы расчета.

Поставки вне системы поставщиков:

$$\xi_{10} = \sigma \cdot (1 - \sum_{j \in \{2,3\}} \alpha_{1j});\quad \xi_{20} = \sigma_2 (1 - \sum_{j \in \{4\}} \alpha_{2j});$$

$$\xi_{30} = \sigma_3 \cdot (1 - \sum_{j \in \{4\}} \alpha_{3j}).$$

Заявки вне системы заказчиков:

$$\eta_{02} = d_2 \cdot (1 - \sum_{w \in \{1\}} \beta_{w2}) / \nu_{02};\ \eta_{03} = d_3 \cdot (1 - \sum_{w \in \{1\}} \beta_{w3}) / \nu_{03};$$

$$\eta_{04} = d \cdot (1 - \sum_{w \in \{2,3\}} \beta_{w4}) / \nu_{04},$$

где

$\nu_{02}, \nu_{03}, \nu_{04}$ - материальные затраты на рубль товарной продукции по заявкам вне системы.

Нереализованная товарная продукция поставщика:

$$\tau_1 = \sum_{j \in \{2,3\}} (\sigma \cdot \alpha_{1j} - \gamma_{1j});\quad \tau_2 = \sum_{j \in \{4\}} (\sigma \cdot \alpha_{2j} - \gamma_{2j});$$

$$\tau_3 = \sum_{j\in\{4\}} (\delta\cdot\alpha_{3j} - \gamma_{3j})\cdot$$

Неудовлетворенный спрос заказчика:

$$\rho_2 = \sum_{w\in\{1\}} (\eta_{w2} - \gamma_{w2}); \quad \rho_3 = \sum_{w\in\{1\}} (\eta_{w3} - \gamma_{w3});$$

$$\rho_4 = \sum_{w\in\{2,3\}} (\eta_{w4} - \gamma_{w4})\cdot$$

В данном разделе, при расчете главных показателей модели, предполагается, что все заявки на материалы и поставки товарной продукции были сделаны внутри системы. Кроме того, не принята во внимание возможность неоплаченных поставок внутри самой модели, это следовало бы учесть введением показателей дебета и кредита. Эти недостающие показатели вводятся в следующем разделе.

3. Общая модель расчетов показателей

Рассматривается множество V, элементами которого являются все ячейки модели, т.е. множество V содержит три типа элементов: ресурсные, производственные, выпускающие (см. разд. 2).

Пусть имеется N элементов множества V, из них l элементов соответствуют ресурсным ячейкам модели, m – выпускающим ячейкам и n – производственным ячейкам. На множестве V определены отношения спроса DEMAND и предложения OFFER. Будем помечать индексом j все элементы множества V, находящиеся в отношении спроса (заказчики), и индексом w элементы, находящиеся в отношении предложения (поставщики). Поскольку рассматривается поточное производство (см. разд. 2), то для каждой пары элементов, связанных отношениями спроса и предложения, можно задать значения индексов w и j так, что $w < j$, т.е.

$$w = 1,2,\ldots,l+n; \quad j = l+1,\ l+2,\ldots,N.$$

Соответственно разделу 2, элементы множества V характеризуются следующими показателями:

1) план по объему реализации товарной продукции (d_j);

2) объем реализации товарной продукции (δ_w);

3) показатель перегрузки производственных ячеек (θ_j);

4) поставки вне системы (ξ_{w0});

5) заявки вне системы (η_{0j});

6) нереализованная товарная продукция поставщика (τ_w);

7) неудовлетворенный спрос заказчика (ρ_j);

8) дебет заказчика (δ_j);

9) кредит поставщика (K_w).

Предположим, что заявки на материалы вне системы обязательно выполняются и поставки товарной продукции вне системы обязательно реализуются. Предположим также, что имеются данные для вычисления величины дебета заказчика. Для этого вводится показатель относительной доли (h_{wj}) предложения товарной продукции, оставшейся нереализованной по данной сделке, которую заказчик согласен принять.

Каждое отношение между элементами множества V характеризуется показателями:

I) размер заявки на товарную продукцию (η_{wj});

2) объем реализации продукции по данной заявке (ξ_{wj});

3) фактический объем реализованной продукции (оплаченная поставка) (γ_{wj}).

Согласно разделу 2, для расчета этих показателей необходимо задать начальные значения исходным данным, которыми являются:

I) планы по объему реализации товарной продукции выпускающих ячеек d_j, где $j = l+n+1, l+n+2, \ldots, N$;

2) объем реализации товарной продукции ресурсных ячеек δ_w, где $w = 1, 2, \ldots, l$;

3) пропускная способность производственных ячеек M_j, где $j = l+1, l+2, \ldots, l+n$;

4) материальные затраты на рубль товарной продукции ν_{wj}, где $w = 1, 2, \ldots, l+n$; $j = l+1, l+2, \ldots, N$;

5) материальные затраты на рубль товарной продукции по заявкам вне системы ν_{0j}, где $j = l+1, l+2, \ldots, N$;

6) относительная доля заявок заказчика j, приходящихся на поставщика w (β_{wj});

7) относительная доля поставок поставщика w, приходящихся на заказчика j (α_{wj});

8) относительная доля предлагаемой в кредит продукции, которую заказчик j в сделке с поставщиком w согласен принять без оплаты (h_{wj}).

Ниже приводятся общие формулы расчетов показателей.

Размер заявки на товарную продукцию определяется как

$$\eta_{wj} = d_j \cdot \beta_{wj} / \nu_{wj};$$

фактический объем реализованной продукции –

$$\gamma_{wj} = \min\{\delta_w \cdot \alpha_{wj},\ \eta_{wj}\};$$

объем реализации товарной продукции по данной заявке –

$$\xi_{wj} = (\gamma_{wj} + (\delta_w \cdot \alpha_{wj} - \gamma_{wj}) \cdot h_{wj})\,\nu_{wj}.$$

Для расчета остальных показателей требуются дополнительные обозначения.

Пусть T_w – множество заказчиков, связанных с данным поставщиком отношением спроса; R_j – множество поставщиков, связанных с данным заказчиком отношением предложения.[I] Тогда план по объему реализации товарной продукции определяется как $d_w \sum_{j \in T_w} d_j \cdot \beta_{wj} / \nu_{wj}$,

объем реализации товарной продукции – $\delta_j = \min\{\sum_{w \in R_j} \xi_{wj} + d_j \cdot (1 - \sum_{w \in R_j} \beta_{wj}), M_j\}$;

показатель перегрузки – $\theta_j = \sum_{w \in R_j} \xi_{wj} + d_j (1 - \sum_{w \in R_j} \beta_{wj}) - \delta_j$;

поставка вне системы – $\xi_{w0} = \delta_w \cdot (1 - \sum_{j \in T_w} \alpha_{wj})$;

заявка вне системы – $\eta_{0j} = d_j (1 - \sum_{w \in R_j} \beta_{wj})\,\nu_{0j}$;

дебет заказчика – $\delta_j = \sum_{w \in R_j} (\delta_w \cdot \alpha_{wj} - \gamma_{wj}) \cdot h_{wj}$;

кредит поставщика $K_w = \sum_{j \in T_w} (\delta_w \cdot \alpha_{wj} - \gamma_{wj}) \cdot h_{wj}$;

[I] Подчеркнем, что расчет по данным формулам можно проводить лишь в том случае, когда для любой пары ячеек, связанных отношениями спроса и предложения, значение индекса w меньше значения индекса j.

неудовлетворенный спрос заказчика $\rho_j = \sum_{w \in R_j} (\eta_{wj} - \gamma_{wj})$;

нереализованное предложение поставщика $\tau_w = \sum_{j \in T_w} (\delta_w \cdot \alpha_{wj} - \gamma_{wj}) \cdot (1 - h_{wj})$.

Как видно, по этим формулам несложно (несмотря на их громоздкость) проводить расчеты всех описанных выше показателей. Однако наша задача состоит не просто в расчете показателей, а в просчете альтернативных вариантов с целью выяснения эффективности материальных затрат на производимую в сети продукцию. Решение этой задачи уже неосуществимо даже простой алгоритмизацией расчетных формул и требует применения специализированной системы.

В следующем разделе рассматривается возможность реализации расчетов на ЭВМ при помощи системы управления данными "Банк ОС".

4. Реализация общей модели расчетов на основе банка данных

В данном разделе рассматривается возможность реализации необходимых расчетов на ЭВМ при помощи СУБД "БАНК-ОС". Преимущество этой системы в том, что она позволяет пользователю создавать сетевую базу данных [1], и приведенный в приложении алгоритм, предназначенный для расчета основных показателей легко реализуется при помощи ее. Основным типом структуры данных [2] в СУБД "БАНК-ОС" является запись. Основным инструментом структурирования данных является связывание записей в цепь. ЦЕПЬ состоит из одной "главной" записи и одной (нескольких) "детальной" записи. Связывание записей в цепь отражает реально существующие отношения между показателями, находящимися в главной и детальной записях.

В целях выявления структуры, необходимой для организации данных в памяти ЭВМ, модель экономической системы рассматривается как взвешенный граф G=(NODE, ARC), где элементы множества V представляются вершинами графа, отношения между элементами - дугами, а показатели - атрибутами вершин и весами дуг графа. Рассматриваемый граф представляет собой сеть.

Вершины и дуги графа выделим как объектные системы NODE и ARC. Вершины графа характеризуются показателями, которые относятся к элементам множества V. Все остальные показатели (см. разд. 3) характеризуют дуги графа.

Следовательно, при построении структуры базы данных можно выделить два типа записей (NODE и ARC). Множество записей под названием NODE будет соответствовать множеству вершин графа, а множество записей под названием ARC - множеству дуг графа. Показатели, характеризующие вершины графа, будут элементами данных записи NODE, а показатели, характеризующие дуги графа - элементами записи ARC.

В приложении I приводится алгоритм прохождения графа для расчета показателей, где используются обозначения, приведенные в разделе 3.

Из приведенных в алгоритме (см. прил. I) процессов прохождения графа, записанных операторами DO $w \in R_j$ и DO $j \in T_w$, следует, что объектные системы NODE и ARC связаны между собой двумя типами отношений. Каждую запись NODE, соответствующую некоторой ячейке модели, которая является заказчиком, необходимо связать с определяющими ее заявки записями ARC. Точно также каждую запись NODE, соответствующую ячейке модели, которая является поставщиком, необходимо связать с определяющими ее поставки записями ARC. Для этой цели определим цепи DEMAND и OFFER, где запись NODE является главной и запись ARC детальной. Здесь конкретная цепь DEMAND определяет заявки, а цепь OFFER — поставки, относящиеся к конкретной записи NODE.

В базе данных необходимо выделить три подмножества записей NODE:

I) записи, у которых цепь DEMAND пуста;

2) записи, у которых цепь OFFER пуста;

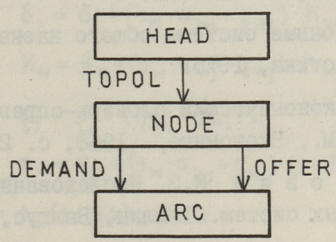
3) записи, у которых цепи DEMAND и OFFER не пусты.

Пустой называем цепь, состоящую только из главной записи. Для упрощения считается, что база данных не содержит

записей, у которых пусты как цепь DEMAND, так и цепь OFFER.

Записи NODE, у которых цепь DEMAND пуста, можно сопоставить с ресурсными ячейками модели, записи NODE, у которых пуста цепь OFFER – с выпускающими ячейками модели и остальные – с производственными ячейками. Кроме того, запись NODE, с непустой цепью DEMAND можно назвать заказчиком, а запись NODE с непустой цепью OFFER – поставщиком.

Далее, в приведенном алгоритме операторы DO j=N TO l+1 BY-1 и DO w=1 TO l+n требуют, чтобы каждая вершина графа имела индекс, а также чтобы они были организованы в линейную последовательность в порядке возрастания (убывания) значения этого индекса. Кроме того, необходимо, чтобы значение индекса поставщика было всегда меньше значения индекса заказчика. Следовательно, необходимо добавить в запись NODE элемент данных INDEX и организовать цепь (TOPOL) с ключом сортировки [I] INDEX, которая объединит все записи NODE в нужном порядке. Это требует ввода в структуру базы данных еще одного типа записи, чтобы иметь "главную" запись в цепи TOPOL. Этот тип записи несет вспомогательную функцию, и база данных будет содержать всего один экземпляр записи HEAD. Полученная структура базы данных графически представлена на фигуре 2. Здесь каждая запись изображается прямоугольником, а каждая цепь – стрелкой, направленной от главной записи к детальной. При этом нужно учитывать, что один прямоугольник отображает все множество однотипных записей, а одна стрелка множество цепей одного имени.



Фиг. 2.

В заключение несколько слов о сфере применения описанной модели. К настоящему моменту осознан факт, что решение задач оптимизации выступает в качестве средства проведения имитационных экспериментов по моделированию деятельности экономических систем. Во-первых, это вызвано невозможностью проведения экспериментов на практике, во-вторых, это один из путей повышения квалификации управленческого персонала.

Рассмотренная выше модель производственной сети выполняет именно эти функции. Так, непосредственно на банке данных осуществимо изменение структуры связей сети, спроса и предложения, мощности предприятия и т.д. Анализ различных вариантов решений углубляет понимание проблемы и повышает способность человека принимать решения в условиях неопределенности [5].

Возможности предложенной модели этим не ограничиваются, так как в ней использованы номенклатурные показатели. Поэтому в реально действующих производственных сетях всегда имеется необходимая информация для создания модели на банке данных. Это обстоятельство позволяет проводить сравнительный анализ расчетных и фактических показателей деятельности производственной сети.

Литература

I. СМО "БАНК ОС". Руководство программиста. Части I, 2, 3. Калинин, 1977.

2. Кнут Д. Искусство программирования для ЭВМ. Т. I. М., Мир, 1976, с. 324-332.

3. Информационные системы общего назначения. Ред. Е.Л. Ющенко. М., Статистика, 1975.

4. Краткий экономический словарь-справочник мастера и начальника цеха. М., Экономика, 1968, с. 280.

5. Каганович И.З. Исследование макроэкономических и отраслевых систем. Таллин, Валгус, 1979, с. 185.

Алгоритм расчета показателей

Вводятся исходные данные:

DO $j = N$ TO $l+1$ BY -1

BEGIN $SB = 0$;

DO $w \in R_j$

BEGIN $\eta_{wj} = d_j \cdot \beta_{wj} \cdot \nu_{wj}$;

$d_w = d_w + \eta_{wj}$;

$SB = SB + \beta_{wj}$; END;

$\eta_{0j} = d_j \cdot (1 - SB) \cdot \nu_{0j}$;

$\delta_j = d_j \cdot (1 - SB)$; END;

DO $w = 1$ TO $l+n$

BEGIN $SA = 0$;

IF $w > l$ THEN $\delta_w = \min(\delta_w, M_w)$;

DO $j \in T_w$

BEGIN $\gamma_{wj} = \min(\delta_w \cdot \alpha_{wj}, \eta_{wj})$;

$\xi_{wj} = (\gamma_{wj} + (\delta_w \cdot \alpha_{wj} - \gamma_{wj}) \cdot h_{wj}) / \nu_{wj}$;

$\delta_j = \delta_j + \xi_{wj}$;

$SA = SA + \alpha_{wj}$;

$\delta_j = \delta_j + (\delta_w \cdot \alpha_{wj} - \gamma_{wj}) \cdot h_{wj}$;

$K_w = K_w + (\delta_w \cdot \alpha_{wj} - \gamma_{wj}) \cdot h_{wj}$;

$\rho_j = \rho_j + (\eta_{wj} - \gamma_{wj})$;

$\tau_w = \tau_w + (\delta_w \cdot \alpha_{wj} - \gamma_{wj}) \cdot (1 - h_{wj})$; END;

$\xi_{w0} = \delta_w (1 - SA)$; END

H. Simanova

The Simulation of a Production Flow Network by Data Base

Summary

A dynamic data base model to simulate production flow network is discussed. Using some behaviour hypothesis of the participants of production process the procedures for computing the indicators of economic activities are described. A process of data base implementation is presented.

№ 482

TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 681.3.06

В.А. Йокк

ОБЗОР ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

I. Введение

В последние годы обращают много внимания на вопросы технологии программирования и проектирования программ. В связи с этим программирование следует рассматривать с двух точек зрения: обучение программированию и создание сложных программных систем. Вопросам создания сложных систем была посвящена I Всесоюзная конференция "Технология программирования", проводившаяся в июне 1979 г. в Киеве. На конференции были обсуждены вопросы проектирования программных систем, методы и средства написания программ, документирование и сопровождение программных систем и их приложения, методы повышения качества программных систем.

Очевидно, не все методы, которые применяются для сложных задач, подходят для простых задач и для обучения программированию. Но надо отметить, что подход к решению задач должен быть в обоих случаях одинаковым, чтобы изучить современные концепции программирования.

В литературе предлагается целый ряд технологий программирования для проектирования программ. Различные технологии используют различные средства для описания и документирования проектирования. Эти технологии будут рассмотрены и проанализированы ниже в различной степени подробности в зависимости от их возможностей практического использования для обучения программированию. Тем самым будет сделана попытка выделить подходящие для обучения технологии.

2. Нисходящее проектирование программ

Теоретически наиболее обоснованным способом проектирования является проектирование сверху вниз или нисходя-

щее проектирование [I, 2]. Сущность этого метода заключается в последовательной декомпозиции функций программ до тех пор, пока программа не будет определена как композиция операторов входного языка ЭВМ. Вышесказанное представляет собой самое общее описание идеи нисходящего проектирования. Более подробно этот метод рассматривается в книге Йодана [3]. Он пишет, что ключом к успешному нисходящему проектированию может служить формализованное и строгое описание входов, функций и выходов всех модулей программы; проектированию структур данных следует уделять не меньше внимания, чем проектированию процессов и алгоритмов.

3. HIPO-технология [4]

Одним из широко распространенных приемов формализованного подхода к нисходящему проектированию является созданная в фирме IBM HIPO-технология (Hierarchy plus Input-Process-Output), первоначально предназначенная для документирования программ. Согласно технологии HIPO полная структура системы должна быть представлена общей диаграммой в виде дерева структур, которое показывает, как каждая функция разделена на подфункции. Каждая отдельная функция в этой иерархии в свою очередь представляется посредством подробной диаграммы входа-обработки-выхода в терминах их входов и выходов. Фазы обработки и их входы-выходы соединяются стрелками. HIPO-диаграмма в основном показывает действия и поток данных.

Так как при использовании этого метода документация проектирования программы очень объемна, то он не представляет с аспекта обучения большого интереса. Технология HIPO может быть полезна для системных аналитиков.

4. Структурное программирование

Сравнительно новым подходом к проектированию и написанию программ является структурное программирование, предложенное Дейкстра и другими. В 1965 г. на конгрессе IFIP Дейкстра высказал предположения, что GOTO оператор мог бы быть исключен из языков программирования. Он повторно изложил свои идеи структурного программирования в 1968-69 гг. [5]. В то же время определенный интерес к этой проблеме проявляли и другие исследователи. Структурное или GOTO-

свободное программирование назвали "революцией в программировании" [6], ему посвящено много работ. Доналдсон пишет, что структурное программирование представляет собой некоторые принципы написания программ, которые делают программы хорошо читаемыми и ясными [7]. Несмотря на довольно широкое освещение этого вопроса, не существует общепринятого определения структурного программирования [8]. Большинство программистов представляет себе структурное программирование как отказ от оператора GOTO и разбиение программы на модули, которые, в свою очередь, могут быть разбиты на еще меньшие модули и т.д. [9]. С точки зрения структурного программирования программа должна проектироваться как линейная последовательность модулей. Каждый модуль имеет один вход и один выход, причем выход предыдущего модуля служит входом следующего модуля, если он существует. Структура каждого модуля может быть при этом довольно сложной. В статье [10] Бёма и Джакопини было показано, что такая структура может быть реализована в языке, включающем только две основные управляющие конструкции: итерацию и выбор. Реализация этих конструкций может быть в разных языках программирования различной. Некоторые теоретические и практические аспекты структурного программирования рассматриваются в книге [3].

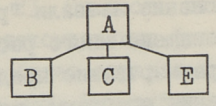
5. Метод Джексона

В 1971 году Джексон предложил методику структурного программирования, которая учит, как программировать сверху-вниз. Технология проектирования программ Джексона (Jackson's method [11]) завоевала широкое признание, особенно в Европе. Он пишет, что проектирование в первую очередь имеет дело со структурой.

Программирование Джексона - это преобразование входных структур данных в выходные структуры данных. Структура программы должна отражать структуру проблемы. Для представления структур Джексон использует так называемые диаграммы структур, компонентами которых являются:

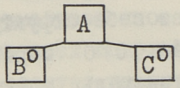
1) элементарная компонента, с именем A

2) последовательность



A – это последовательность, которая состоит из частей B, C, E в данном порядке,

3) выбор



A – это выбор. При этом выбирается либо B, либо C

4) итерация

A – это итерация, в которой повторно выполняется B.

На одном уровне дерева диаграммы структур все элементы одинаковы (либо последовательность, либо выбор, либо итерация). Используя для проектирования программ методику Джексона, следует:

I) определить структуру входных и выходных данных,

2) построить структуру программы, которая определяется структурами данных, т.е. структура программы должна содержать компоненты, связанные с компонентами данных (должны быть представлены то же число раз, в том же порядке и при тех же условиях),

3) записать операции и связать их с соответствующими компонентами структуры программы.

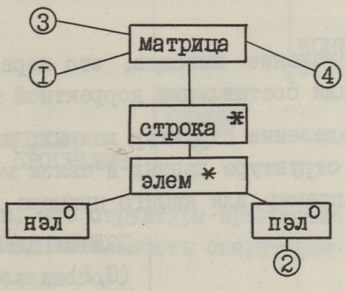
Вышеизложенный метод легко применяется в случае т.н. простых программ ([8], с. 246). Но, к сожалению далеко не все программы простые. В случае, если программа непосредственно не определяется при помощи входных и выходных структур данных из-за их несовместимости, возникает т.н. структурный конфликт. Стандартным решением такого конфликта является попытка разбиения программы на ряд подпрограмм, представляемых при помощи вспомогательного файла как отдельные фазы обработки. Записи вспомогательного файла выбираются так, чтобы они не вступали в конфликт с входными и выходными структурами данных. Структурные конфликты анализируются

в статье [I2]. Там же показывается, что дерево Джексона дает законченное представление языка и представляет только регулярные языки.

Метод Джексона является систематическим методом для решения широкого класса задач обработки данных. Примеры таких задач приведены в работах Джексона.

Так как метод Джексона хорошо обозрим и легко применим для простых задач, то его можно рекомендовать для обучения программированию.

В качестве примера возьмем простую задачу, которая содержит основные конструкции диаграммы структур. Например, нахождение суммы положительных элементов квадратной матрицы, расположенных ниже главной диагонали.



Действия: I. S = 0
2. S = S + A(i,k)
3. Ввод N, A
4. Печать S

Прикрепим действия к элементам структуры программы и запишем программу на произвольном языке, двигаясь по структуре программы слева вниз и справа вверх:

```
DIM A(N,N)
READ A
S=0
FOR I=2 TO N
FOR K=1 TO I-1
IF A(I,K)>0 THEN
S=S+A(I,K)
NEXT K
NEXT I
PRINT S.
```

6. Технология Варнье-Орра

Варнье и Орр развили технологию программирования (Warnier-Orr diagrams) [I3], которая основывается на теории множеств и булевой алгебре и содержит целый ряд возможностей для проектирования программ. Технология представляется при помощи диаграмм Варнье-Орра, которые для своего изображения используют фигурные скобки. Эти диаграммы дают возможность легко проектировать, кодировать и составлять программы, обычно правильно работающие с первой попытки. Такой подход может быть применен к маленьким или к большим программам.

Эта технология представляет собой сжатый, пошаговый метод, который дает хорошие результаты. Шаги метода следующие.

Первый шаг: определение выходов, это первое, основное и важнейшее правило для составления корректной программы.

Второй шаг: определение структур данных; имеется в виду только логическая структура данных и связи между компонентами структуры, например, для нашего примера

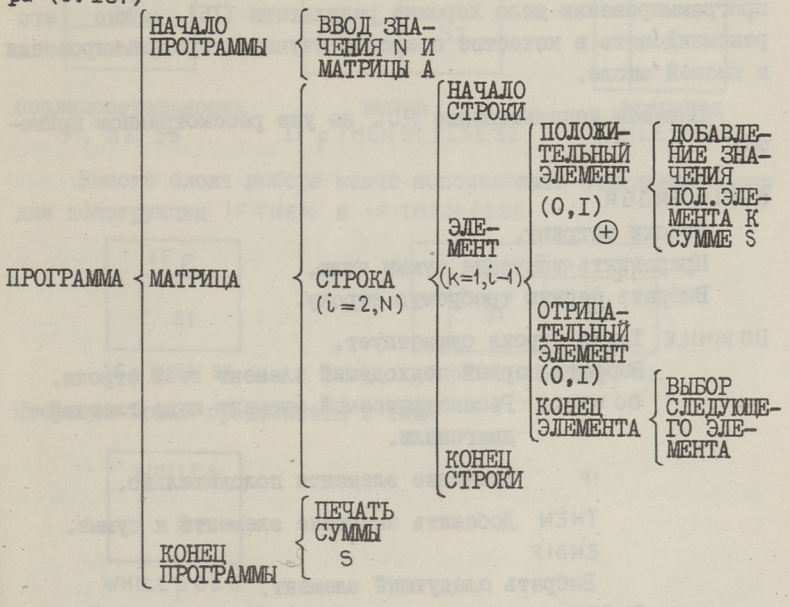
МАТРИЦА { СТРОКА (i = 2, N) { ЭЛЕМЕНТ (k=1, i-1) { ПОЛОЖИТЕЛЬНЫЙ ЭЛЕМЕНТ (0,I) ⊕ ОТРИЦАТЕЛЬНЫЙ ЭЛЕМЕНТ (0,I)

Третий шаг: определение физических структур данных; один уровень структуры соответствует одной фигурной скобке, которые перечисляются слева направо; для каждого уровня структуры задаются требуемые уточнения.

Четвертый шаг: проектирование структуры процесса обработки.

В случае работы с одной программой структура процесса представляет собой структуру этой программы. В случае работы нескольких программ структура процесса представляет эти программы и управление их работой происходит при помощи объединенной системы процедур. Структура процесса зависит от логических структур данных.

Задаем полную диаграмму Варнье-Орро для нашего примера (с. 107)



Отметим, что структура программы записывается слева направо и последовательность операторов – сверху (с первого) вниз (до последнего).

7. Язык проектирования программ

Язык проектирования программ (Program Design Language-PDL) – это структурный язык псевдо-кода, который используется в структурном программировании сверху вниз. Первоначально PDL был использован для обучения структурному программированию. Вначале PDL как средство обучения помог программистам перейти к мышлению в терминах структурного программирования, в дальнейшем его начали использовать для проектирования программ. PDL обеспечивает выполнение элементов структурного программирования и вызов других модулей. Действия (функции) могут быть описаны либо на обычном языке, либо на более ограниченном языке программирования. Язык PDL может быть легко преобразован в нужный язык программирования, так как его синтаксис сходен с синтаксисом обычных языков программирования, таких как PL/1 и COBOL. Подробно синтаксис PDL описывается в статье [15].

Поскольку использование PDL для обучения структурному программированию дало хорошие результаты [I5], можно его рекомендовать в качестве средства изучения программирования в высшей школе.

Покажем использование PDL на уже рассмотренном примере.

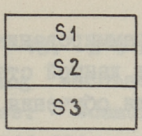
```
BEGIN PROGR
    Ввести матрицу.
    Приравнять значение суммы нулю.
    Выбрать первую требуемую строку.
DO WHILE  Такая строка существует.
          Выбрать первый подходящий элемент этой строки.
          DO WHILE  Рассматриваемый элемент ниже главной
                    диагонали.
              IF    Значение элемента положительно.
              THEN  Добавить значение элемента к сумме.
              ENDIF
              Выбрать следующий элемент.
          ENDDO
          Выбрать следующую строку.
ENDDO
          Напечатать значение суммы.
END PROGR
```

Заметим, что благодаря ступенчатой записи, легко прослеживается структура алгоритма в программе. В частичности, инструкции, принадлежащие одной и той же структурной единице, должны напечататься на одинаковом расстоянии от левого края.

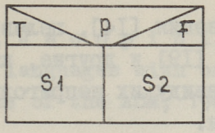
8. NSCC-схема

Развитие структурного программирования привело к замене обычных блок-схем альтернативными схемами.

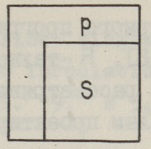
Одна из таких схем NSCC-схема (Nassi/Schneiderman, Chapin charts) [I6], которая описывает три основные операции структурного программирования - последовательность, выбор и итерацию при помощи прямоугольников.



последовательность
S1, S2, S3

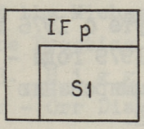


выбор
IF p THEN S1 ELSE S2

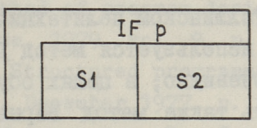


итерация
WHILE p DO S

Вместо блока выбора можно использовать отдельные блоки для конструкции IF THEN и IF THEN ELSE

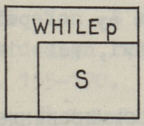


IF p THEN S1



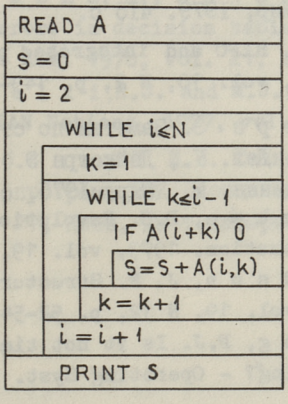
IF p THEN S1 ELSE S2

Итерацию можно представить в виде



WHILE p DO S

Представим наш пример в терминах NSCC-схемы:



Этот метод успешно применялся для написания программы на ФОРТРАНЕ, КОБОЛЕ, БЕЙСИКЕ и ПАСКАЛЕ.

9. Заключение

Существует еще целый ряд интересных технологий программирования, таких как таблицы решений [I7], метод струк-

турного программирования [I4], целевое программирование [I8], R-технология [I9] и другие, которые в данной статье не рассматриваются ввиду их непригодности для обучения началам проектирования.

Анализ различных вариантов технологий программирования и опыт внедрения некоторых из них позволяют сделать вывод, что для обучения началам программирования лучше всего подходят графические методы.

В Таллинском политехническом институте с I976 года с успехом используется метод Джексона и с I979 года - NSCC-схемы. Очевидно, в целях обучения программированию можно применять также метод Варнье-Орра и PDL.

Литература

I. W i r t h, N. Program development by stepwise refinement. - Communications of the ACM, 1971, vol. 14, N 4, p. 221-227.

2. В и р т Н. Систематическое программирование. М., Мир, I977. I88 с.

3. Йодан Э. Структурное проектирование и конструирование программ. М., Мир, I979. 4I5 с.

4. S t a y, J. HIPO and integrated program design. - IBM Syst. J., 1976, vol. 10, N 2, p. 143-154.

5. Д е й к с т р а Э. Заметки по структурному проектированию. - В кн.: Дал. У., Дейкстра Э., Хоор К. Структурное программирование. М., Мир, I975, с. 7-97.

6. M c C r a c k e n, D.D. Revolution in programming: an overview. - Datamation, 1973, vol. 19, N 12, p. 50-52.

7. D o n a l d s o n, J. R. Structured programming. - Datamation, 1973, vol. 19, N 12, p. 52-54.

8. D e n n i n g, P.J. Is it not time to define "Structure programming"? - Operating Syst. Rev., Jan. 1974, 8, p. 6-7.

9. T a n e n b a u m, A.S. Implications of structured programming for machine architecture. - Communications of the ACM, 1978, vol. 21, N 3, p. 237-246.

10. B ö h m, C., J a c o p i n i, G. Flow diagrams, Turing Machines, and languages with only two formulation rules. - Communications of the ACM, 1966, vol. 9, N 5, p. 366-371.

11. J a c k s o n, M.A. Constructive methods of program design. - Lecture Notes in Computer Science. ECI Conference 1976, p. 236-262.

12. H u g h e s, J.W. A formalization and explication of the Michael Jackson Method of program design. - Software Practice and Experience, 1979, vol. 9, p. 191-202.

13. H i g g i n s, D.A. Structured programming with Warnier - Orr Diagrams. - BYTE, December 1977, p. 104-110.

14. S t e v e n s, W., M e y e r s, G., C o n s t a n t i n e, L. Structured design. - IBM Syst. J., 1974, vol. 13, N 1, p. 114-139.

15. Van L e e r, P. Top-down development using a program design language. - IBM Syst. J., 1976., vol. 15, N 2, p. 155-170.

16. N a s s i, I., S h n e i d e r m a n, B. Flowchart techniques for structured programming. - SIGPLAN Notices, 1973, vol. 8, N 8, p. 12-26.

17. I b r a m s h a, M., R a j a r a m a n, V. Detection of logical errors in decision table programs. - Communications of the ACM. 1978, vol. 21, N 12, p. 1016-1025.

18. C i c h e l l i, R.J. and M.J., Coal directed programming.- SIGPLAN Notices, 1977, vol. 6, N 6, p. 51-59.

19. В е л ь б и ц к и й И.В. Технологический комплекс производства программ РТК (концепции и возможности). - препринт - 79 - 27, Киев, Ин-т кибернетики, 1979. 52 с.

V. Jokk

A Review of Programming Methodology

Summary

The past few years have seen a massive upsurge in interest in programming methodology. The interest in programming methodology has led to the development of rigid rules for program design. The primary benefit of program design is a program which is written with fewer errors, easier to understand, and easier to test and debug.

This paper surveys several important design methodologies and examines these approaches for application in higher educational institutions.

№ 482

TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 518.5:519.2

Л.К. Выханду, Э.Х.-Т. Нунацуу

ОБРАБОТКА СОЦИАЛЬНО-ЭКОНОМИЧЕСКИХ ПОКАЗАТЕЛЕЙ В ДИАЛОГОВОМ РЕЖИМЕ

I. Введение

Анализ данных в любой области социально-экономической жизни должен представить средства и методы для оценивания конкретных явлений и извлечения информации, нужной для управления рассмотренным явлением. Типичным для такого подхода является нахождение т.н. производственных функций [6]. Эти функции должны ответить на вопрос, как именно связано изменение какой-то экономической величины с изменением влияющих на нее существенных факторов. В теории и практике статистики известны хорошие пособия, основательно описывающие методику получения таких функций. Для проведения нужных вычислений имеются прикладные пакеты статистических программ, работающие на ЭВМ в пакетном режиме [4].

Наряду с классической (утверждающей) статистикой в последние годы на научную арену вышла новая ветвь статистики, т.н. исследовательский анализ данных. (Exploratory Data Analysis). Главной целью этой школы является выдвижение целого ряда методов т.н. обозревания данных. Основной принцип подхода можно описать в двух предложениях [5]:

Все, что дает более простое описание какого-то явления, делает это описание легко используемым.

Все, что затрагивает более глубокие по сравнению с ранее описанными уровни анализируемого явления, делает описание эффективнее.

Если при изучении данного явления на более глубоком уровне ничего не обнаружится, то и это определенный шаг вперед, хотя и менее полезный, чем в первом случае.

Для изучения явления по слоям особенно важно иметь возможность его графического изображения, но как подчеркивает Тьюки, ценность такого изображения возрастет максимально, если она заставляет нас заметить то, чего мы никогда не ожидали увидеть [5].

Конечно, без выдвижения и подтверждения статистических гипотез, результаты, предлагаемые на основании графоаналитических методов, являются только дескриптивными, несмотря на то, насколько поучительные были эти результаты.

К счастью, в последние годы развита эффективная методика апостериорного подтверждения статистических гипотез, т.н. метод складного ножа [2].

Так как исследовательский анализ данных тесно связан с графическими методами, специально с изучением остатков от использованных моделей, то естественным является использование интерактивных методов в диалоговом режиме. В нашем распоряжении имеется удобный для этих целей миникомпьютер ВАНГ-2200, для которого и была создана система анализа структуры систем данных и пошагового анализа подбора моделей. Описание всех возможностей системы требует отдельной статьи, мы приводим только результаты использования системы для описания рождаемости в ЭССР.

2. Графики связей. Выпрямление их

Основная методика графоаналитического подхода состоит в последовательном разложении данных на основании равенства:

заданные данные = оценки + остатки

Оценки при помощи разных моделей дают текущее описание данных и являются всегда неполной приближенной картиной общего поведения данных. Остатки являются основным средством более глубокого анализа данных. Оценки значений данных дают обычно описание общих тенденций, которые нам зачастую "и так известны". Остатки как увеличительное стекло дают более точную информацию о совместимости данных и модели. Они же служат и новыми данными для дальнейшего анализа.

Основным приемом на каждом шаге последовательного изучения данных является преобразование переменных так, что-

бы функциональная связь между ними приобрела прямолинейный вид. Проводя прямую линию через совокупность точек (x, y), мы получаем выражение вида $\bar{y} = A + Bx$. График остатков $(\bar{y}, y - \bar{y})$ и дает нам правило останова. Если в графике остатков еще имеется явная тенденция связи, то требуется дальнейшее выпрямление.

Для выпрямления используется шкала степеней

$t^p (p = n, n-1, \ldots, 1;\ p = 1,\ 1/1+\varepsilon, \ldots, 1/2, \ldots, 1/n)$,

$\ln t$,

$t^p (p = -1/n, \ldots, -1/2, \ldots, -1/1+\varepsilon;\ p = -1, -2, \ldots, -n)$.

Подбор правильного преобразования управляется простым мнемоническим правилом. Если график остатков имеет вогнутость, то при выборе преобразования надо двигаться по шкале направо, в противном случае – налево.

Конкретное значение степени близкое к оптимальному достигается, если в остатках уже нет никакого порядка.

Пример. Пусть задана таолица значений функции

x	I	2	3	4	5	6
y	3	3,78	4,33	4,76	5,I3	5,45

В качестве первого приближения сглаживаем данные при помощи прямой $y = A + Bx$. Для сглаживания можно выбирать или обычный метод наименьших квадратов (ОМНК), или какой-то вариант т.н. взвешенного робастного приближения [7]. Если воспользоваться последним вариантом, то уравнение прямой запишется как $\bar{y} = 2.793 + 0{,}475\ x$. График остатков $(\bar{y},\ y - \bar{y})$ имеет значения

$\bar{y}$	3,27	3,74	4,22	4,69	5,I7	5,64
$y-\bar{y}$	-0,27	0,04	0,II	0,07	-0,04	-0,I9

и выпуклость вверх. Значит при выборе преобразования надо двигаться по шкале от непреобразованного значения y налево. Выбираем y^2. Преобразуем данные и сглаживаем при помощи прямой. График остатков имеет значения

$\bar{y}^2$	I0,07	I4,I7	I8,26	22,35	26,44	30,53
$y^2-\bar{y}^2$	-I,07	0,I2	0,49	0,3I	-0,I2	-0,83

Выпуклость направлена снова вверх. При преобразовании y^3 распределение остатков уже носит случайный характер:

y^3	27,00	53,00	81,00	107,97	134,95	161,93
$y^3-\bar{y}^3$	-0,00	0,01	0,20	0,12	0,06	-0,06

следовательно, выпрямленная функциональная связь имеет вид

$$y = \sqrt{A + Bx}, \qquad (1)$$

где робастная оценка параметров

$$A = 0{,}02866 \text{ и } B = 26{,}98427$$

Вычисленные по формуле значения (I) совпадают с заданными в пределах точности.

3. Подбор функции рождаемости в ЭССР

В моделях демографических прогнозов, базирующихся на методе передвижки возрастов [I], в качестве экзогенных переменных используются матрицы коэффициентов смертности, рождаемости и миграции. Эти матрицы довольно больших размеров, и изолированное долгосрочное прогнозирование их элементов на основе формальных методов затруднительно. Поэтому среди экзогенных переменных необходимо найти такие, которые с достаточной для практики точностью определяют функцию полного распределения возрастных групп.

Для коэффициентов рождаемости таким элементом является рождаемость r_a для женщин в возрасте a, при котором рождаемость максимальна. До сих пор самым признанным аналитическим выражением связи между фертильностью и возрастом (x) женщины является следующее, предложенное Ястремским [3]

$$r_x = r_a\left(1+\frac{x-a}{a-c}\right)\left(1-\frac{x-a}{b-a}\right)^{(b-a)(a-c)}, \quad x \neq c,\ x \neq b,$$

где r_x – коэффициент рождаемости для женщин в возрасте x;
a – возраст с максимальной фертильностью;
c – начальный возраст периода фертильности женщин;
b – конечный возраст периода фертильности женщин.

Использование методики Ястремского дало на конкретных данных (за 1970 и 1973 годы) Эстонской ССР следующую картину (табл. I). Совпадение прогноза с реальными данными хорошее только в возрастных группах с 21 по 32 год, в остальных группах имеются заметные отклонения. Поэтому нами была поставлена задача нахождения лучшего аналитическо-

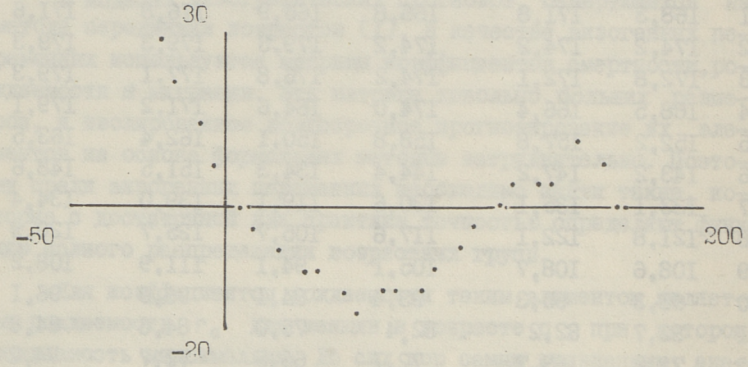
Таблица I

Таблица повозрастных коэффициентов рождаемости в ЭССР и их приближения

Возраст	Действительные значения	По Ястремскому (2)	По формулам (3),(4)	Действительные значения	По Ястремскому (2)	По формулам (3),(4)
I5	0,9	0	I,0	0,9	0	I,0
I6	4,2	53,8	3,8	3,8	55,4	4,0
I7	I5,0	95,7	I4,4	I6,I	98,5	I4,8
I8	44,7	I27,0	45,8	43,9	I30,7	47,2
I9	94,3	I49,3	I02,2	9I,6	I53,7	I05,2
20	I37,7	I63,9	I48,0	I39,I	I68,7	I52,3
2I	I68,3	I7I,8	I66,8	I66,9	I76,8	I7I,6
22	I74,2	I74,2	I74,2	I79,3	I79,3	I79,3
23	I72,8	I72,I	I74,2	I76,8	I77,I	I79,3
24	I68,5	I66,4	I74,0	I64,6	I7I,2	I79,I
25	I52,2	I57,8	I58,8	I50,I	I62,4	I63,5
26	I43,2	I47,2	I44,4	I34,3	I5I,5	I48,6
27	I33,I	I35,I	I30,6	II9,I	I39,0	I34,4
28	I2I,8	I22,I	II7,6	I05,7	I25,7	I20,9
29	I08,6	I08,7	I05,I	94,I	III,9	I08,2
30	96,3	95,3	93,4	84,I	98,0	96,I
3I	82,7	82,2	82,4	75,0	84,6	84,8
32	72,2	69,7	72,I	66,5	7I,7	74,2
33	62,3	58,I	62,5	58,I	59,8	64,3
34	53,0	47,4	53,6	50,0	48,8	55,I
35	43,4	37,9	45,3	42,0	39,0	46,6
36	37,8	29,6	37,7	34,6	30,4	38,8
37	30,7	22,4	30,8	27,9	23,I	3I,7
38	23,I	I6,5	24,7	2I,9	I6,9	25,4
39	I8,9	II,6	I9,2	I6,8	I2,0	I9,7
40	I3,2	7,8	I4,4	I2,5	8,0	I4,8
4I	9,6	4,9	I0,3	9,0	5,I	I0,6
42	6,9	2,9	6,8	6,2	3,0	7,0
43	3,2	I,5	4,I	4,0	I,6	4,2
44	2,4	0,7	2,I	2,4	0,7	2,I
45	I,3	0,2	0,7	I,3	0,3	0,7
46	0,7	0,I	0,I	0,5	0,I	0,I

го выражения для оценки фертильности женщин в ЭССР. Одновременно кроме качества приближения соблюдалась и возможная простота нужных вычислений.

После нескольких неудачных попыток найти единую простую функцию для всех возрастных групп, было принято решение разделить изучаемые данные в точке максимальной фертильности на две части и найти приближения для каждой в отдельности. По методике предыдущего раздела была проведена прямая линия через точки правее значения x = 22. Вогнутость остатков (фиг. I) указывает на то, что по шкале преобразования надо двигаться направо. Преобразование $\sqrt{y}$ дает уже почти случайный характер распределения остатков (фиг. 2).



Фиг. 1.

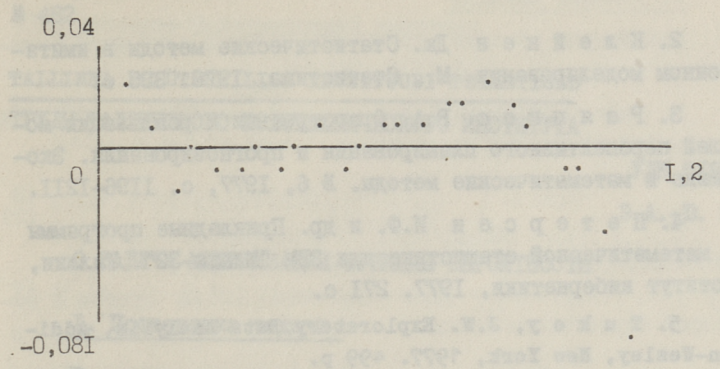
Чтобы убедиться в добротности предложенного преобразования, проводим робастным методом прямую через преобразованные точки $(x, \sqrt{r_x/r_a})$ и получим в оригинальных переменных уравнение

$$r_x = r_a\,(2{,}0676 - 0{,}0445\,x)^2, \quad (r_x > r_a). \qquad (3)$$

Для первой части кривой $(r_x < r_a)$ ввиду ее S-образности используем логит-преобразование $\bar{y} = \ln(y/(1-y))$, где $y = r_x/r_a$. Проводя робастным методом прямую через точки $(x, \ln(r_x/(r_a - r_x)))$ после обратного преобразования

$$r_x = r_a/(1 + e^{25{,}8728 - 1{,}3802x}), \quad (r_x < r_a) \qquad (4)$$

с достаточно случайным расположением остатков. Здесь можно отметить, что пробит-преобразование дало бы еще лучшее при-



Фиг. 2.

ближение, но сложность вычислений при использовании результатов (требуются специальные таблицы) заставляет нас остановиться на логит-преобразовании.

Если дополнительно взять $r_a = r_{max}$, то полученные уравнения (3) и (4) дают довольно хорошее представление о фертильности в Эстонской ССР. Конкретные результаты приведены в табл. I. Как следует из таблицы, полученное описание намного лучше приближения Ястремского.

Для сравнения используем те же уравнения для I973 года. И здесь приведенное нами приближение лучше, чем предсказания по Ястремскому (табл. I).

Замечание. Полученное выпрямление связи имеет принципиальное значение для дальнейшего уточнения параметров кривых связи. Приведенный метод просто указывает, какое преобразование лучше всего подходит для описания взаимосвязи между переменными x и y. Дальнейшее уточнение коэффициентов происходит уже при помощи какого-то варианта метода наименьших квадратов (обычный МНК, нелинейный МНК, робастный МНК). В нашем случае такое уточнение и было проведено, но результаты практически не отличались от здесь приведенных.

Литература

I. Басалаева Н.А. Моделирование демографических процессов и трудовых ресурсов. М., Наука, I973. 87 с.

2. К л е й н е н Дж. Статистические методы в имитационном моделировании. М., Статистика, 1978. 335 с.

3. Р а я ц к а с Р.А. О построении и реализации моделей перспективного планирования и прогнозирования. Экономика и математические методы. № 6, 1977, с. 1196-1211.

4. П е т е р с е н И.Ф. и др. Прикладные программы по математической статистике для ЭВМ "Минск-32". Таллин, Институт кибернетики, 1977. 271 с.

5. T u k e y, J.W. Exploratory Data Analysis. Addison-Wesley, New York, 1977. 499 p.

6. V e n s e l, V. Tootmisfunktsioonid. Tallinn, TPI, 1975. 59 lk.

7. M c N e i l, D.R. Interactive Data Analysis. New York, Wiley, 1977. 185 p.

L. Vyhandu, E. Õunapuu

Interactive Processing of Socio-economic Data

Summary

In this paper a robust method to choose a data transformation for linear model fitting is presented. As a practical example the fertility function of Estonian women is analysed.

№ 482

TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 519.95

Э.А.-Ю. Юби

О ЗАДАЧЕ МИНИМИЗАЦИИ ФУНКЦИИ ВЕРОЯТНОСТИ

I. Постановка задачи

В статье уточняются условия сходимости предложенного автором в работе [I] метода безусловной минимизации функции вероятности.

Минимизировать

$$v(x) = P[f(x,y) \geq 0], \qquad (I)$$

где $f(x,y): R^n \times R^m \to R^1$, $x = (x_1,\dots,x_n) \in R^n$, $y = (y_1,\dots,y_m) \in Y \subset R^m$,

y – случайный вектор, обладающий плотностью $p(y)$.

В работе [I] предлагается для решения задачи (I) прямой итеративный метод, аналогичный методу стохастической аппроксимации Кифера-Вольфовица для минимизации функции регрессии. В нем используется недифференцируемая оценка вероятности $P[f(x,y) \geq 0]$ – эмпирическая частота $\eta[f(x,y) \geq 0;\ N_1]/N_1$, где $\eta[f(x,y) \geq 0;\ N_1]$ обозначает число реализаций y, удовлетворяющих условию $f(x,y) \geq 0$ при общем числе независимых реализаций y, равном N_1.

В работе [I] выведены только для сепарабельной функции $f(x,y) = f_1(x) + f_2(y)$ условия того, когда градиент $v'(x)$ удовлетворяет некоторому условию Липшица. В данной работе рассматривается несепарабельный случай. Получены также более общие условия псевдоградиентности рассматриваемого алгоритма.

В работе [2] предлагается для решения задачи (I) заменить функцию $v(x)$ ее оценкой Парзена – дифференцируемой по x оценкой, построенной по реализациям y.

2. Условие Липшица

Рассмотрим сначала случай многомерной случайной величины y, $y=(y_1, y_2, \dots, y_m)$, $m>1$.

Э. Райк [3] доказал, что при выполнении условий I – 4 нижеследующей теоремы I градиент $v'(x)$ имеет вид

$$v'(x)=\int_{S(x,0)} \frac{f'_x(x,y)}{\|f'_y(x,y)\|}\, p(y)\, d\, S_x(y), \qquad (2)$$

где интегрируется по $m-1$-мерной поверхности $S(x,0)=\{y: y\in Y, f(x,y)=0\}$. Если множество $S(x,0)$ пустое в некоторой окрестности точки x, то $v'(x)=0$.

Если $v(x)$ меняется только в ограниченной области пространстве R^n, то на границе этой области градиент $v'(x)$ может и не существовать. Достаточное условие, гарантирующее существование $v'(x)$ в таком случае при выполнении условий теоремы I, например, такое:

если в любой окрестности точки x существуют как точки $x_1\in R^n$, такие, что $S(x_1,0)=\phi$, так и точки $x_2\in R^n$, такие, что $S(x_2,0)\neq\phi$, то $s(x_k)=\int_{S(x_k,0)} dS_{x_k}(y)\to 0$, $x_k\to x$, где $\{x_k\}$ любая последовательность точек, таких, что $S(x_k,0)\neq\phi$, $x_k\in R^n$.

При выполнении этого условия и по условиям 3 и 6 теоремы I

$$\|v'(x_k)\|\le \frac{c_1 M_p}{c_0}\, s(x_k)\to 0, \quad x_k\to x,$$

где $\{x_k\}$ любая последовательность, такая что $S(x_k,0)\neq\phi, x_k\in R^n$. Следовательно, $v'(x)$ существует и $v'(x)=0$. В остальных точках x производная $v'(x)$ либо равняется нулю, либо вычисляется по формуле Райка (2).

Теорема I. Пусть для любых x, x_1 и $x_2\in R^n$ выполнены следующие условия

1) $f'_x(x,y)$, $f'_y(x,y)$, $p(y)$ существуют и непрерывны, $y\in Y$,
2) множество $S(x,0)=\{y: y\in Y, f(x,y)=0\}$ непустое и равномерно ограничено в некоторой окрестности точки x,
3) $\|f'_y(x,y_1)-f'_y(x,y_2)\|\le L_1\|y_1-y_2\|$, $y_1, y_2\in Y$ и $0<c_0\le\|f'_y(x,y)\|$, $y\in S(x,0)$,

4) существует поверхностный интеграл

$$\int\limits_{S(x,0)} dS_x(y) = s(x) \leq M_S,$$

5) $\max\limits_{y_1 \in S(x_1,0)} \min\limits_{y_2 \in S(x_2,0)} \|y_1 - y_2\| \leq c_3 \|x_1 - x_2\|$.

6) $\|f'_y(x,y)\| \leq c_1$, $y \in S(x,0)$ и

$\|f'_x(x_1,y_1) - f'_x(x_2,y_2)\| \leq L_0[\|x_1 - x_2\| + \|y_1 - y_2\|]$, $y_1, y_2 \in Y$,

7) $\|f'_y(x_1,y) - f'_y(x_2,y)\| \leq L_2 \|x_1 - x_2\|$, $y \in Y$,

а $p(y) \leq M_p$, $y \in Y$,

$\|f'_y(x,y)\| \leq M_0$, $y \in S(x,0)$,

$|y_i| \leq R$, $i = 1,\ldots,m$, $y = (y_1,\ldots,y_m) \in S(x,0)$,

тогда

$$\|v'(x_1) - v'(x_2)\| \leq L \|x_1 - x_2\|,$$

где

$$L = M_S M_p \left[\frac{L_0}{c_0} + \frac{2(c_1+1)L_2}{c_0^2}\right] + m M_p \left\{\frac{(2R)^{m-2} c_3}{m} \cdot \left[\frac{4m(c_1+1)M_0}{c_0^2} + \frac{mc_1}{c_0}\right] + \frac{2M_S}{c_0}\left[\frac{2m^2 c_1 M_0 L_0}{c_0^2} + \frac{2mc_1 L_2}{c_0} + \frac{4m^2 c_1 (M_0+1) L_2}{c_0^2} + \frac{2m^3 c_1^2 (M_0+1) L_1}{c_0^3}\right]\right\}.$$

Доказательство.

$$\left\| \int\limits_{S(x_1,0)} \frac{f'_x(x_1,y_1)}{\|f'_y(x_1,y_1)\|} p(y_1) dS_{x_1}(y_1) - \int\limits_{S(x_2,0)} \frac{f'_x(x_2,y_2)}{\|f'_y(x_2,y_2)\|} p(y_2) dS_{x_2}(y_2) \right\| \leq$$

$$\leq \left\| \int\limits_{S(x_1,0)} \frac{f'_x(x_1,y_1)}{\|f'_y(x_1,y_1)\|} p(y_1) dS_{x_1}(y_1) - \int\limits_{S(x_1,0)} \frac{f'_x(x_2,y_1)}{\|f'_y(x_2,y_1)\|} p(y_1) dS_{x_1}(y_1) \right\| +$$

$$+ \left\| \int\limits_{S(x_1,0)} \frac{f'_x(x_2,y_1)}{\|f'_y(x_2,y_1)\|} p(y_1) dS_{x_1}(y_1) - \int\limits_{S(x_2,0)} \frac{f'_x(x_2,y_2)}{\|f'_y(x_2,y_2)\|} p(y_2) dS_{x_2}(y_2) \right\| =$$

$$= I_1 + I_2, \quad y_1 \in S(x_1,0), \quad y_2 \in S(x_2,0).$$

Подинтегральное выражение в I_1 оценивается следующим образом.

$$\left\| \frac{f'_x(x_1,y_1)}{\|f'_y(x_1,y_1)\|} - \frac{f'_x(x_2,y_1)}{\|f'_y(x_2,y_1)\|} \right\| \le \left\| \frac{f'_x(x_1,y_1)}{\|f'_y(x_1,y_1)\|} - \frac{f'_x(x_2,y_1)}{\|f'_y(x_1,y_1)\|} \right\| +$$

$$+ \left\| \frac{f'_x(x_2,y_1)}{\|f'_y(x_1,y_1)\|} - \frac{f'_x(x_2,y_1)}{\|f'_y(x_2,y_1)\|} \right\| \le \frac{L_0\|x_1-x_2\|}{c_0} + \frac{c_1+L_0\|x_1-x_2\|}{c_0(c_0-L_2\|x_1-x_2\|)} \cdot$$

$$\cdot \left| \|f'_y(x_2,y_1)\| - \|f'_y(x_1,y_1)\| \right| \le \frac{L_0}{c_0}\|x_1-x_2\| + \frac{c_1+L_0\|x_1-x_2\|}{c_0(c_0-L_2\|x_1-x_2\|)} \cdot$$

$$\cdot L_2\|x_1-x_2\| \le \left[\frac{L_0}{c_0} + \frac{2(c_1+1)L_2}{c_0^2}\right]\|x_1-x_2\| \text{ при достаточно малой } \|x_1-x_2\|.$$

Поэтому $I_1 \le M_S M_P \left[\frac{L_0}{c_0} + \frac{2(c_1+1)L_2}{c_0^2}\right]\|x_1-x_2\|$.

Для оценки I_2 разделим поверхность $S(x_2,0)$ на m непересекающихся частей $S^i(x_2,0)$, где

$$S^i(x_2,0) = \left\{ y_2 : y_2 \in S(x_2,0) \setminus \bigcup_{j=1}^{i-1} S^j(x_2,0),\ |f'_{y_i}(x_2,y_2)| \ge \frac{c_0}{m} \right\},$$

$1 \le i \le m$. По условию $\|f'_y(x_2,y_2)\| \ge c_0$, $y_2 \in S(x_2,0)$. Возможно, что для некоторых i $S^i(x_2,0) = \phi$, $1 \le i \le m$. Разделим также поверхность $S(x_1,0)$ на m непересекающихся частей

$$S^i(x_1,0) = \left\{ y_1 : y_1 \in S(x_1,0) \setminus \bigcup_{j=1}^{i-1} S^j(x_1,0),\right.$$

$$\left. \exists\, y_2 \in S^i(x_2,0),\ \|y_1-y_2\| \le c_3\|x_1-x_2\| \right\},\quad 1 \le i \le m.$$

$\bigcup_{j=1}^{m} S^j(x_1,0) = S(x_1,0)$, так как для любой точки $y_1^0 \in S(x_1,0)$ существует точка $y_2^0 \in S(x_2,0)$, такая, что $\|y_1^0 - y_2^0\| \le c_3\|x_1-x_2\|$. Если $S^j(x_2,0) \ne \phi$, то при достаточно малой $\|x_1-x_2\|$ и $S^j(x_1,0) \ne \phi$.

Представим оба интеграла в I_2 в виде суммы m интегралов по $S^i(x_k,0)$, $k=1,2$ и пусть, например, $S^m(x_2,0) \ne \phi$. Оценим норму разности соответствующих слагаемых, обозначая через $D^m(x_k)$ проекцию $S^m(x_k,0)$ на координатную плоскость $y_1,\dots,y_{m-1}$, $k=1,2$.

$$A_m \equiv \Big\| \int\limits_{S^m(x_1,0)} \frac{f'_x(x_2,y_1)}{\|f'_y(x_2,y_1)\|}\, p(y_1)\, dS_{x_1}(y_1) - \int\limits_{S^m(x_2,0)} \frac{f'_x(x_2,y_2)}{\|f'_y(x_2,y_2)\|}\, p(y_2)\, dS_{x_2}(y_2) \Big\| =$$

$$= \Big\| \int\limits_{D^m(x_1)} \frac{f'_x(x_2,\tilde{y}_1)}{\|f'_y(x_2,\tilde{y}_1)\|}\, p(\tilde{y}_1) \det \Big|\frac{\partial S}{\partial \tilde{y}_1}\Big|\, dy_1 \dots dy_{m-1} -$$

$$- \int\limits_{D^m(x_2)} \frac{f'_x(x_2,\tilde{y}_2)}{\|f'_y(x_2,\tilde{y}_2)\|}\, p(\tilde{y}_2) \det \Big|\frac{\partial S}{\partial \tilde{y}_2}\Big|\, dy_1 \dots dy_{m-1} \Big\|,$$

где $\tilde{y}_k = (\tilde{y}_{k1}, \dots, \tilde{y}_{km})$ обозначает точку, первые $m-1$ координат которой меняются произвольно на координатной плоскости $y_1 \dots y_{m-1}$ во множестве $D^m(x_k)$, а m-ая координата определяется из условия $f(x_k, \tilde{y}_k) = 0$, $k = 1, 2$. Определитель преобразования $\det \Big|\frac{\partial S}{\partial \tilde{y}_k}\Big| = \|f'_y(x_k, y_k)\| \, |f'_{y_m}(x_k, y_k)|^{-1}$. В дальнейшем для простоты записи будем вместо $\tilde{y}_k$ писать y_k.

$$A_m = \Big\| \int\limits_{D^m(x_1) \setminus D^m(x_2)} \frac{f'_x(x_2,y_1)}{\|f'_y(x_2,y_1)\|}\, p(y_1)\, \frac{\|f'_y(x_1,y_1)\|}{|f'_{y_m}(x_1,y_1)|}\, dy_1 \dots dy_{m-1} +$$

$$+ \int\limits_{D^m(x_1) \cap D^m(x_2)} \left[\frac{f'_x(x_2,y_1)}{\|f'_y(x_2,y_1)\|}\, p(y_1)\, \frac{\|f'_y(x_1,y_1)\|}{|f'_{y_m}(x_1,y_1)|} - \frac{f'_x(x_2,y_2)\, p(y_2)\, \|f'_y(x_2,y_2)\|}{\|f'_y(x_2,y_2)\|\, |f'_{y_m}(x_2,y_2)|} \right] dy_1 \dots dy_{m-1} -$$

$$- \int\limits_{D^m(x_2) \setminus D^m(x_1)} \frac{f'_x(x_2,y_2)\, p(y_2)\, \|f'_y(x_2,y_2)\|}{\|f'_y(x_2,y_2)\|\, |f'_{y_m}(x_2,y_2)|}\, dy_1 \dots dy_{m-1} \Big\|.$$

Во втором интеграле переменные интегрирования $y_1, \dots, y_{m-1}$ меняются произвольно в $D^m(x_1) \cap D^m(x_2)$, причем у точек y_1 и y_2 отличаются только m-ые координаты, которые определяются из условий $f(x_1, y_1) = 0$ и $f(x_2, y_2) = 0$. Так как

$$f(x_2,y_2) = f(x_1,y_1) + (f'_x(\tilde{x}(\Theta), \tilde{y}(\Theta)), x_2 - x_1) + (f'_y(\tilde{x}(\Theta), \tilde{y}(\Theta)), y_2 - y_1),$$

где $\tilde{x}(\Theta) = x_1 + \Theta(y)(x_2 - x_1)$, $\tilde{y}(\Theta) = y_1 + \Theta(y)(y_2 - y_1)$, $0 \le \Theta(y) \le 1$, откуда $|y_{1m} - y_{2m}| = \|y_1 - y_2\| \le \frac{m c_1}{c_0} \|x_1 - x_2\|$. Этим неравенством мы воспользуемся для оценки малости подинтегральной функции во втором интеграле.

Нетрудно убедиться, что норма подинтегральной функции в первом интеграле удовлетворяет неравенству

$$\frac{\|f'_x(x_2,y_1)\|\,\|f'_y(x_1,y_1)\|}{\|f'_y(x_2,y_1)\|\,|f'_{y_m}(x_1,y_1)|} \leqslant \frac{(c_1+1)M_0}{\frac{c_0}{2}\cdot\frac{c_0}{2m}} = \frac{4mM_0(c_1+1)}{c_0^2}$$

при достаточно малой $\|x_1-x_2\|$.

Норма подинтегральной функции в третьем интеграле

$$\left\|\frac{f'_x(x_2,y_2)\|f'_y(x_2,y_2)\|}{\|f'_y(x_2,y_2)\|\,|f'_{y_m}(x_2,y_2)|}\right\| = \left\|\frac{f'_x(x_2,y_2)}{|f'_{y_m}(x_2,y_2)|}\right\| \leqslant \frac{mc_1}{c_0}.$$

Оценим

$$\int_{D^m(x_1)\setminus D^m(x_2)} dy_1\ldots dy_{m-1}.$$

В любой точке модуль разности пределов интегрирования хотя бы по одной переменной не превосходит $\frac{c_0}{m}\|x_1-x_2\|$. В противном случае существовала бы точка $y_1\in D^m(x_1)$ такая, что

$$\min_{y_2\in D^m(x_2)}\|y_1-y_2\| > c_3\|x_1-x_2\|.$$

Поэтому

$$\int_{D^m(x_1)\setminus D^m(x_2)} dy_1\ldots dy_{m-1} \leqslant (2R)^{m-2}\frac{c_3}{m}\|x_1-x_2\|,$$

где для всех $x\in R^n$ выполняется $S(x,0)\subset\prod_{i=1}^{m}[-R,R]\subset R^m$.

Точно также

$$\int_{D^m(x_2)\setminus D^m(x_1)} dy_1\ldots dy_{m-1} \leqslant (2R)^{m-2}\frac{c_3}{m}\|x_1-x_2\|.$$

Так как

$$s(x) = \int_{S(x,0)} dS_x(y) \leqslant M_S,\quad x\in R^n,$$

то

$$\int_{D^m(x_1)\cap D^m(x_2)} dy_1\ldots dy_{m-1} \leqslant M_S.$$

После громоздких преобразований можно убедиться, что при достаточно малой $\|x_1-x_2\|$ норма подинтегральной функции во втором интеграле оценивается сверху величиной

$$\frac{2}{c_0}\left[\frac{2m^2c_1M_0L_0}{c_0^2}+\frac{2mc_1L_2}{c_0}+\frac{4m^2c_1(M_0+1)L_2}{c_0^2}+\frac{2m^3c_1^2(M_0+1)L_1}{c_0^3}\right]\|x_1-x_2\|.$$

При выводе неравенства следует воспользоваться тем, что при достаточно малой $\|x_1-x_2\|$ выполняется $\|y_1-y_2\|\leqslant\frac{mc_1}{c_0}\|x_1-x_2\|$, $y_1\in S(x_1,0)$, $y_2\in S(x_2,0)$. Следовательно,

$$I_2 \leqslant m M_p \Big\{ (2R)^{m-2} \frac{c_3}{m} \Big[\frac{4m(c_1+1)M_0}{c_0^2} + \frac{mc_1}{c_0} \Big] + \frac{2M_s}{c_0} \Big[\frac{2m^2 c_1 M_0 L_0}{c_0^2} +$$

$$+ \frac{2mc_1 L_2}{c_0} + \frac{4m^2 c_1 (M_0+1) L_2}{c_0^2} + \frac{2m^3 c_1^2 (M_0+1) L_1}{c_0^3} \Big] \Big\} \| x_1 - x_2 \|.$$

Окончательно получим

$$\| v'(x_1) - v'(x_2) \| \leqslant L \| x_1 - x_2 \|,$$

где

$$L = 1_s M_p \Big[\frac{L_0}{c_0} + \frac{2(c_1+1)L_2}{c_0^2} \Big] + m M_p \Big\{ (2R)^{m-2} \frac{c_3}{m} \cdot \Big[\frac{4m(c_1+1)M_0}{c_0^2} +$$

$$+ \frac{mc_1}{c_0} \Big] + \frac{2M_s}{c_0} \Big[\frac{2m^2 c_1 M_0 L_0}{c_0^2} + \frac{2mc_1 L_2}{c_0} + \frac{4m^2 c_1 (M_0+1) L_2}{c_0^2} +$$

$$+ \frac{2m^3 c_1^2 (M_0+1) L_1}{c_0^3} \Big] \Big\}.$$

Замечание. Для $m=1$ формула Райка имеет вид

$$v'(x) = \sum_{k=1}^{N} \frac{f'_x(x, y^k)}{|f'_y(x, y^k)|} p(y^k),$$

где суммирование распространяется на все $y^k \in Y$ и удовлетворяющие условию $f(x, y^k) = 0$. Покажем, что при выполнении условий теоремы I оценку постоянной Липшица можно провести непосредственно. По условию (5) теоремы для любого $y_1^k \in S(x_1, 0)$ выполняется

$$\min_{y_2 \in S(x_2, 0)} |y_1^k - y_2| \leqslant c_3 \| x_1 - x_2 \|,$$

т.е. существует точка $y_2^k \in S(x_2, 0)$, удовлетворяющая условию $|y_1^k - y_2^k| \leqslant c_3 \| x_1 - x_2 \|$.

$$\left\| \frac{f'_x(x_1, y_1)}{|f'_y(x_1, y_1)|} - \frac{f'_x(x_2, y_2)}{|f'_y(x_2, y_2)|} \right\| \leqslant \left\| \frac{f'_x(x_1, y_1)}{|f'_y(x_1, y_1)|} - \frac{f'_x(x_1, y_1)}{|f'_y(x_1, y_2)|} \right\| +$$

$$+ \left\| \frac{f'_x(x_1, y_1)}{|f'_y(x_1, y_2)|} - \frac{f'_x(x_1, y_1)}{|f'_y(x_2, y_2)|} \right\| + \left\| \frac{f'_x(x_1, y_1)}{|f'_y(x_2, y_2)|} - \frac{f'_x(x_2, y_2)}{|f'_y(x_2, y_2)|} \right\| \leqslant$$

$$\leqslant \frac{c_1}{c_0 (c_0 - L_1 |y_1 - y_2|)} \Big| |f'_y(x_1, y_1)| - |f'_y(x_1, y_2)| \Big| + \frac{c_1}{c_0 (c_0 - L_2 \| x_1 - x_2 \|)}.$$

$$\left|\,|f'_y(x_1,y_2)|-|f'_y(x_2,y_2)|\,\right|+\frac{L_0}{c_0}\left[\|y_1-y_2\|+\|x_1-x_2\|\right]\leqslant$$

$$\leqslant\left[\frac{2c_1c_3L_1}{c_0^2}+\frac{2c_1L_2}{c_0^2}+\frac{L_0c_3}{c_0}+\frac{L_0}{c_0}\right]\|x_1-x_2\|$$

при достаточно малой $\|x_1-x_2\|$, где $|y_1-y_2|\leqslant c_3\|x_1-x_2\|$, $y_i\in S(x_i,0)$.

Окончательно получим

$$\|v'(x_1)-v'(x_2)\|\leqslant L\|x_1-x_2\|,$$

где

$$L=NM_p\left[\frac{2c_1c_3L_1}{c_0^2}+\frac{2c_1L_2}{c_0^2}+\frac{c_3L_0}{c_0}+\frac{L_0}{c_0}\right].$$

3. Псевдоградиентность алгоритма

Предложенный в работе [I] алгоритм имеет вид

$$x_k=x_{k-1}-\gamma_k S_k, \qquad (3)$$

где $x_k\in R^n$, x_0 – произвольная точка,
$0\leqslant\gamma_k$ – скалярный детерминированный множитель,
$S_k=(S_{1k},\ldots,S_{nk})$,

$$S_{ik}=\frac{\eta\left[f(x_{k-1}+\Delta_{ik}e_i,y)\geqslant 0;N_1\right]-\eta\left[f(x_{k-1}-\Delta_{ik}e_i,y)\geqslant 0;N_1\right]}{2\Delta_{ik}N_1} \qquad (4)$$

$i=1,\ldots,n$, e_i – единичные орты, $k=1,2\ldots$

Пусть $v(x)$ имеет единственную точку минимума x^*. Для расширения класса функций, удовлетворяющих следствиям теоремы Поляка–Цыпкина рассмотрим вопрос псевдоградиентности алгоритма (3)–(4) по отношению к вспомогательной функции $u(x)=v(x)+(x-x^*)^2$. Очевидно, что функция $u(x)$ имеет также единственную точку минимума x^*. Если $v(x)$ удовлетворяет предположениям теоремы Поляка–Цыпкина, то и $u(x)$ удовлетворяет этим предположениям. Однако для $u(x)$ могут быть выполнены некоторые предположения следствий теоремы, которым $v(x)$ не удовлетворяет. Например, множества $\{x: u(x)\leqslant \leqslant \mathrm{const}\}$ ограничены для $u(x)$, но аналогичные множества для $v(x)$ не ограничены.

Лемма I работы [I] утверждает, что условия теоремы Поляка–Цыпкина выполняются для некоторой подпоследовательности (в случае поискового алгоритма условие псевдоградиент-

ности может нарушаться на некоторых шагах). Пример такой подпоследовательности, члены которой удовлетворяют условию

$$(v'(x_{k_r-1}), s_{k_r}) \geq 0, \qquad (5)$$

для всех k_r.

Наибольший интерес представляет случай псевдовыпуклой функции $v(x)$. Для такой функции, если минимум $v(x)$ достигается при любом i $(i=1,\ldots,n)$ на одном из концов отрезка $[x_{k_j-1}-\Delta_{ik_j}e_i, x_{k_j-1}+\Delta_{ik_j}e_i]$ для подпоследовательности $\{x_{k_j}\}$, то знаки всех составляющих градиента $v'(x_{k_j})$ и ее оценки s_{k_j} совпадают и подпоследовательность $\{x_{k_j}\}$ удовлетворяет условию псевдовыпуклости. Действительно,

$$(u'(x_{k-1}), M_k s_k) = \sum_{i=1}^{n} [v'_{x_i}(x_{k-1}) + 2(x_{k-1i}-x_i^*)][M_k\eta[f(x_{k-1}+\Delta_{ik}e_i, y) \geq 0;$$

$$N_1] - M_k\eta[f(x_{k-1}-\Delta_{ik}e_i, y) \geq 0; N_1](2\Delta_{ik}N_1)^{-1} = \sum_{i=1}^{n} v'_{x_i}(x_{k-1})[v(x_{k-1}+$$

$$+\Delta_{ik}e_i) - v(x_{k-1}-\Delta_{ik}e_i)](2\Delta_{ik})^{-1} + \sum_{i=1}^{n} 2(x_{k-1i}-x_i^*)[v(x_{k-1}+\Delta_{ik}e_i) -$$

$$-v(x_{k-1}-\Delta_{ik}e_i)](2\Delta_{ik})^{-1} = S_1 + S_2.$$

Все произведения в первой сумме S_1 не отрицательны по условию (5). Покажем, что вторая сумма S_2 не отрицательна. Допустим противное, что $S_2 < 0$. По непрерывности $v(x)$ вторая сумма непрерывно зависит от n аргументов $\Delta_{i1}, \ldots, \Delta_{in}$ и так как

$$\lim_{\max|\Delta_{ij}| \to 0} S_2 \to 2(x_{k-1}-x^*, v'(x_{k-1})) > 0$$

по псевдовыпуклости $v(x)$ при $x_{k-1} \neq x^*$, то мы получим противоречие.

Литература

I. Юби Э.А.-Ю. Минимизация функции вероятности методом статистических испытаний. – Тр. Таллинск. политехн. ин-та, 1976, № 411, с. 57-76.

2. Тамм Э.Ф. О минимизации функции вероятности. Известия АН ЭССР. Физика. Математика, 1979, т. 28. № 1, с. 17-24.

3. Райк Э.В. Дифференцируемость по параметру функции вероятности и стохастический псевдоградиентный метод для ее оптимизации. Известия АН ЭССР. Физика. Математика, 1975, т. 24, № 1, с. 329.

E. Ubi

About the Minimization of the Probability Function

Summary

In this paper the author's earlier results on the minimization of the probability function are generalized.

№ 482

TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 519.4

П. Пуусемп

ОБ ОПРЕДЕЛЯЕМОСТИ ПОЛУПРЯМОГО ПРОИЗВЕДЕНИЯ ЦИКЛИЧЕСКИХ ГРУПП СВОЕЙ ПОЛУГРУППОЙ ЭНДОМОРФИЗМОВ

Введение

В работе [2] доказано, что каждая p-группа, являющаяся полупрямым произведением двух своих циклических подгрупп, определяется своей полугруппой эндоморфизмов. В настоящей работе обобщаем этот результат. Мы докажем, что каждая конечная группа G, являющаяся полупрямым произведением $G=\langle a\rangle \lambda \langle b\rangle$ двух своих циклических подгрупп $\langle a\rangle$ и $\langle b\rangle$, $\langle a\rangle \simeq C_{p^\alpha}$, p – произвольное простое число, определяется своей полугруппой эндоморфизмов (теорема 27). Еще покажем, что существует группа $G=\langle a\rangle \lambda \langle b\rangle$, $\langle a\rangle \not\simeq C_{p^\alpha}$, которая не определяется своей полугруппой эндоморфизмов (предложения 1 и 2).

В работе многократно ссылаемся на работы [1] и [2], знакомство с которыми было бы желательным. Из этих же работ взяты используемые обозначения. Кроме них будем пользоваться следующими обозначениями:

$\mathfrak{Z}(\mathrm{End}G)$ – центр полугруппы $\mathrm{End}G$;

$\hat{G}=\{\hat{g} \mid g\in G\}$ – группа внутренних автоморфизмов группы G;

$G_{\mathrm{End}G}(x)=\{y\in \mathrm{End}G \mid yx=xy\}$;

σ – эквивалентность на множестве $J(G)$ всех идемпотентов полугруппы $\mathrm{End}G$, при котором $x\sigma y$ тогда и только тогда, когда $xy=y$ и $yx=x$;

$[x]_\sigma$ – класс эквивалентности по σ, $x\in J(G)$.

1. Пример группы $G=\langle a\rangle \lambda \langle b\rangle$, которая не определяется своей полугруппой эндоморфизмов.

В данном пункте приводим пример, показывающий, что существуют конечные группы, которые являются полупрямыми

произведениями двух своих циклических подгрупп, но не определяются своими полугруппами эндоморфизмов.

Рассмотрим группы G и H, задаваемые соответственно определяющими соотношениями

$$b^3 = a_1^7 = a_2^{13} = 1, \quad a_1 a_2 = a_2 a_1, \quad b^{-1} a_1 b = a_1^2, \quad b^{-1} a_2 b = a_2^3 \tag{I}$$

и

$$d^3 = c_1^7 = c_2^{13} = 1, \quad c_1 c_2 = c_2 c_1, \quad d^{-1} c_1 d = c_1^2, \quad d^{-1} c_2 d = c_2^9. \tag{2}$$

В силу (I) и (2) ясно, что имеют место равенства

$$G = (\langle a_1 \rangle \times \langle a_2 \rangle) \lambda \langle b \rangle = \langle a_1 a_2 \rangle \lambda \langle b \rangle, \tag{3}$$

$$H = (\langle c_1 \rangle \times \langle c_2 \rangle) \lambda \langle d \rangle = \langle c_1 c_2 \rangle \lambda \langle d \rangle, \tag{4}$$

т.е. группы G и H являются полупрямыми произведениями двух своих циклических подгрупп. Порядки групп G и H равны: $|G| = |H| = 3 \cdot 7 \cdot 13 = 273$. Покажем, что группы G и H не изоморфны, а их полугруппы всех эндоморфизмов изоморфны.

Предложение I. Группы G и H не изоморфны.

Доказательство. Предположим, что группы G и H изоморфны. Тогда в группе H существуют элементы g, h_1 и h_2, удовлетворяющие условиям, аналогичным условиям (I). В частности, $g^{-1} h_1 g = h_1^2$ и $g^{-1} h_2 g = h_2^3$. В силу равенства (4) ясно, что единственными силовскими 7- и I3-подгруппами группы H являются соответственно $\langle c_1 \rangle$ и $\langle c_2 \rangle$. Поэтому $h_1 = c_1^i, h_2 = c_2^j$ для некоторых $i \in Z_7$, $j \in Z_{13}$; $i, j \neq 0$. Подгруппа $\langle d \rangle \simeq C_3$ является силовской 3-подгруппой группы H. В силу разложения (4) и сопряженности силовских подгрупп имеем $g = (c_1^k c_2^l)^{-1} \cdot d^m (c_1^k c_2^l)$ при некоторых $k \in Z_7$, $l \in Z_{13}$, $m \in Z_3$, $m \neq 0$ (т.е. $m = 1$ или $m = 2$).

Если $m = 1$, то не имеет место равенство $g^{-1} h_2 g = h_2^3$. Действительно,

$$\begin{aligned} g^{-1} h_2 g = g^{-1} c_2^j g = (g^{-1} c_2 g)^j = [(c_1^k c_2^l)^{-1} \cdot d^{-1} \cdot \\ \cdot (c_1^k c_2^l) \cdot c_2 \cdot (c_1^k c_2^l)^{-1} d (c_1^k c_2^l)]^j = \\ = [(c_1^k c_2^l)^{-1} d^{-1} c_2 d (c_1^k c_2^l)]^j = [(c_1^k c_2^l)^{-1} c_2^9 \cdot \\ \cdot (c_1^k c_2^l)]^j = c_2^{9j}, \quad h_2^3 = c_2^{3j}, \end{aligned}$$

но в Z_{13} в силу $j \neq 0$, $9j \neq 3j$.

Если $m = 2$, то аналогично можно убедиться, что не выполняется равенство $g^{-1}h_1 g = h_1^2$. Следовательно, группы G и H не изоморфны. Предложение доказано.

Предложение 2. Полугруппы $EndG$ и $EndH$ изоморфны.

Доказательство. Определим полугруппу $EndH$. Предположим, что $\tau \in EndH$. Как отмечено, группа $\langle d \rangle$ является силовской 3-подгруппой группы H. Поэтому 3-элементами группы H являются элементы вида $g^{-1}d^{\kappa}g = (g^{-1}dg)^{\kappa}$, где $g \in \langle c_1 \rangle \times \langle c_2 \rangle$ и $\kappa \in Z_3$ (см. также равенство (4)), т.е. $d\tau = g^{-1}d^{\kappa}g$ при некоторых $g \in \langle c_1 \rangle \times \langle c_2 \rangle$ и $\kappa \in Z_3$. Так как $\langle c_1 \rangle$ и $\langle c_2 \rangle$ являются соответственно единственными силовскими 7- и 13-подгруппами группы H, то $c_1\tau = c_1^i$ и $c_2\tau = c_2^j$ для некоторых $i \in Z_7$, $j \in Z_{13}$. Для определения элементов полугруппы $EndH$ надо определить теперь, при каких значениях i, j, κ и $g \in \langle c_1 \rangle \times \langle c_2 \rangle$ отображение τ сохраняет определяющие соотношения (2) группы H.

Ясно, что τ сохраняет четыре первых соотношения из (2) для каждых i, j, κ и g. Определим, когда оно сохраняет последние два соотношения из (2).

В силу $g \in \langle c_1 \rangle \times \langle c_2 \rangle \triangleright H$ и соотношений (2) имеем

$$(d\tau)^{-1}(c_1\tau)(d\tau) = (g^{-1}d^{\kappa}g)^{-1}c_1^i(g^{-1}d^{\kappa}g) =$$
$$= g^{-1}d^{-\kappa}gc_1^i g^{-1}d^{\kappa}g = g^{-1}d^{-\kappa}c_1^i d^{\kappa}g = d^{-\kappa}c_1^i d^{\kappa} =$$
$$= (d^{-\kappa}c_1 d^{\kappa})^i = c_1^{2^{\kappa}i}, \quad (c_1\tau)^2 = c_1^{2i},$$

т.е. равенство $(d\tau)^{-1}(c_1\tau)(d\tau) = (c_1\tau)^2$ имеет место тогда и только тогда, когда

$$2i \equiv 2^{\kappa}i \pmod 7. \qquad (5)$$

Аналогично равенство $(d\tau)^{-1}(c_2\tau)(d\tau) = (c_2\tau)^9$ имеет место тогда и только тогда, когда

$$9j \equiv 9^{\kappa}j \pmod{13}. \qquad (6)$$

Следовательно, τ сохраняет определяющие соотношения (2) группы H и является эндоморфизмом группы H тогда и только тогда, когда выполнены условия (5), (6).

Если $\kappa = 0$, то из (5) и (6) следует, что $i = 0$ и $j = 0$, т.е. $d\tau = c_1\tau = c_2\tau = 1$ и τ – нулевой эндоморфизм. Если $\kappa = 1$, то сравнения (5) и (6) имеют место для каждых i и j, т.е.

τ является эндоморфизмом группы H для каждых i, j и g. Если $\kappa = 2$, то из (5) и (6) вытекает, что $i = 0$, $j = 0$, т.е. τ является эндоморфизмом при $i = 0$, $j = 0$ и при любом $g \in \langle c_1 \rangle \times \langle c_2 \rangle$. Следовательно, эндоморфизмами группы H являются следующие отображения:

1) $d \mapsto 1$, $c_1 \mapsto 1$, $c_2 \mapsto 1$ (нулевой эндоморфизм);

2) $d \mapsto g^{-1}dg$, $c_1 \mapsto c_1^i$, $c_2 \mapsto c_2^j$, $i \in Z_7$, $j \in Z_{13}$, $g \in \langle c_1 c_2 \rangle$;

3) $d \mapsto g^{-1}d^2g = g^{-1}d^{-1}g$, $c_1 \mapsto 1$, $c_2 \mapsto 1$, $g \in \langle c_1 c_2 \rangle$.

Так как $c_1, c_2 \notin C_H(d)$ и порядки элементов c_1 и c_2 являются простыми числами, то $(\langle c_1 \rangle \times \langle c_2 \rangle) \cap C_H(d) = \langle 1 \rangle$ и различным элементам g группы $\langle c_1 \rangle \times \langle c_2 \rangle$ соответствуют различные эндоморфизмы вида 3). Обозначим эндоморфизм вида 3) через $\bar{g}$. Эндоморфизм вида 2) однозначно определяется парой (α, g), где $\alpha \in \mathrm{End}(\langle c_1 \rangle \times \langle c_2 \rangle)$ и $c_1\alpha = c_1^i$, $c_2\alpha = c_2^j$. Следовательно,

$$\mathrm{End}H = \{0\} \cup \{\bar{g} \mid g \in \langle c_1 c_2 \rangle\} \cup \{(\alpha, g) \mid \alpha \in \mathrm{End}\langle c_1 c_2 \rangle, g \in \langle c_1, c_2 \rangle\}.$$

Определим правило умножения в полугруппе $\mathrm{End}H$. Пусть $(\alpha, g), (\beta, h) \in \mathrm{End}H$. Тогда

$$d((\alpha, g)(\beta, h)) = (g^{-1}dg)(\beta, h) = (g\beta)^{-1} \cdot h^{-1}dh \cdot (g\beta),$$

$$c_1((\alpha, g)(\beta, h)) = c_1(\alpha\beta), \quad c_2((\alpha, g)(\beta, h)) = c_2(\alpha\beta),$$

т.е.

$$(\alpha, g) \cdot (\beta, h) = (\alpha\beta, h \cdot (g\beta)). \tag{7}$$

Аналогично можно убедиться в справедливости следующих равенств:

$$(\alpha, g) \cdot \bar{h} = \bar{h}, \quad \bar{h} \cdot (\alpha, g) = \overline{g \cdot (h\alpha)}, \quad \bar{g} \cdot \bar{h} = (0, h). \tag{8}$$

Поэтому $\mathrm{End}H$ является полугруппой с внешне присоединенным нулем и правило умножения в $\mathrm{End}H \setminus \{0\}$ задается равенствами (7) и (8). Легко вывести, что $|\mathrm{End}H| = 1 + 7 \cdot 13 + (7 \cdot 13)^2 = 8373$.

Совершенно аналогично доказывается, что

$$\mathrm{End}G = \{0\} \cup \{\bar{g} \mid g \in \langle a_1 a_2 \rangle\} \cup \{(\alpha, g) \mid \alpha \in \mathrm{End}\langle a_1 a_2 \rangle, g \in \langle a_1 a_2 \rangle\}$$

и умножение в $\mathrm{End}G \setminus \{0\}$ задается правилами (7) и (8). Следовательно, полугруппы $\mathrm{End}G$ и $\mathrm{End}H$ изоморфны. Предложение доказано.

2. Некоторые свойства полугруппы $\mathrm{End}(\langle a\rangle \lambda \langle b\rangle)$

Пусть в пределах данного пункта $G=\langle a\rangle\lambda\langle b\rangle$ - группа, заданная следующими определяющими соотношениями:

$$b^m = a^{p^\alpha} = 1,\ b^{-1}ab = a^r, \tag{9}$$

где p - простое число, $\alpha \geqslant 1$, $m > 1$. Предположим, что

$$m = q_1^{\beta_1} \dots q_n^{\beta_n}, \tag{10}$$

где $q_1,\dots,q_n$ - различные простые числа и $\beta_i \geqslant 1$. В данном пункте докажем некоторые свойства полугруппы $\mathrm{End}\,G$. Позднее убедимся, что эти свойства полностью определяют саму группу G.

Обозначим через x идемпотент, соответствующий полупрямому разложению $G=\langle a\rangle\lambda\langle b\rangle$. Тогда $\mathrm{Ker}x=\langle a\rangle\simeq C_{p^\alpha}$, $\mathrm{Im}x=\langle b\rangle\simeq C_m$.

Лемма 3. $K(x)\simeq \mathrm{End}\,C_m$.

Утверждение леммы сразу следует из изоморфизма $K(x)\simeq \mathrm{End}(\mathrm{Im}x)$ (см. [1], лемма I.6).

Лемма 4. $P(x)\simeq \mathrm{End}\,C_{p^\alpha}$.

Доказательство. Как известно (см. [2], лемма I), полугруппа $P(x)$ состоит из таких $\gamma\in\mathrm{End}\,G$, для которых $\gamma|_{\mathrm{Im}x}=1_{\mathrm{Im}x}$ и $\gamma|_{\mathrm{Ker}x}\in\mathrm{End}(\mathrm{Ker}x)$, т.е.

$$b\gamma = b,\ a\gamma = a^i,\ i\in Z_{p^\alpha}. \tag{11}$$

Отображение γ, заданное равенствами (II), сохраняет при каждом $i\in Z_{p^\alpha}$ определяющие соотношения (9) группы G. Поэтому ясно, что $P(x)\simeq \mathrm{End}\,C_{p^\alpha}$. Лемма доказана.

Лемма 5. Группа $V(x)$ разлагается в полупрямое произведение $V(x)=A\lambda\mathfrak{D}(x)$, где A - некоторая циклическая p-подгруппа группы $V(x)$.

Доказательство леммы 5 аналогично доказательству леммы 9 работы [2]. Оттуда следует, что $V(x)=A\lambda\mathfrak{D}(x)$, где A изоморфна некоторой подгруппе аддитивной группы кольца Z_{p^α}, т.е. A - циклическая p-группа.

Лемма 6. Пусть M - конечная разрешимая группа, $z\in J(M)$ и $\mathrm{Im}z=\langle f\rangle$. Тогда $H(z)=\{0\}$ тогда и только тогда, когда $(|\mathrm{Ker}z|,|\mathrm{Im}z|)=1$. Если $(|\mathrm{Ker}z|,|\mathrm{Im}z|)=1$, то $|[z]_\sigma|=[\mathrm{Ker}z : C_{\mathrm{Ker}z}(\mathrm{Im}z)]$.

Доказательство. Пусть выполнены предположения леммы. Тогда $M = \mathrm{Ker}z \lambda \mathrm{J}mz$. Если $H(z) \neq \{0\}$, то существует $y \in H(z)$, $y \neq 0$, т.е. $\mathrm{J}my \neq \langle 1 \rangle$. Тогда в силу $\mathrm{J}my \leqslant \mathrm{Ker}z \leqslant \mathrm{Ker}y$ (см. [2], лемма 2) ясно, что $\langle 1 \rangle \neq \mathrm{J}my = (\mathrm{J}mz)y = \langle fy \rangle \leqslant \mathrm{Ker}z$, и следовательно, $(|\mathrm{Ker}z|, |\mathrm{J}mz|) \neq 1$. Если $(|\mathrm{Ker}z|, |\mathrm{J}mz|) \neq 1$, то существует простое число q, на которое делятся $|\mathrm{Ker}z|$ и $|\mathrm{J}mz|$. Теперь для каждого $h \in \mathrm{Ker}z$, $o(h) = q$, эндоморфизм τ, где $(\mathrm{Ker}z)\tau = \langle 1 \rangle$, $f\tau = h$ является ненулевым элементом множества $H(z)$. Следовательно, $H(z) = \{0\}$ тогда и только тогда, когда $(|\mathrm{Ker}z|, |\mathrm{J}mz|) = 1$.

Предположим теперь, что $(|\mathrm{Ker}z|\,|\mathrm{J}mz|) = 1$. Тогда группы $\mathrm{Ker}z$ и $\mathrm{J}mz$ являются холловскими подгруппами группы M. Так как $[z]_\sigma = \{y \in J(M) \mid \mathrm{Ker}z = \mathrm{Ker}y\}$ (см. [2], лемма 5), то для каждого $y \in [z]_\sigma$ имеем $\mathrm{J}my \simeq \mathrm{J}mz$. Ввиду разрешимости группы M имеем $\mathrm{J}my = g^{-1} \cdot \mathrm{J}mz \cdot g$ для некоторого $g \in M$. Следовательно, при $y \in [z]_\sigma$ имеем

$$\mathrm{Ker}z = \mathrm{Ker}y, \quad \mathrm{J}my = g^{-1} \cdot \mathrm{J}mz \cdot g, \tag{I2}$$

$$M = \mathrm{Ker}z \lambda (g^{-1} \cdot \mathrm{J}mz \cdot g). \tag{I3}$$

Наоборот, при каждом $g \in M$ имеет место (I3) и идемпонент y, соответствующий полупрямому разложению (I3), удовлетворяет равенствам (I2) и, значит, принадлежит $[z]_\sigma$. Теперь ясно, что $|[z]_\sigma| = [M : C_M(\mathrm{J}mz)] = [\mathrm{Ker}z : C_{\mathrm{Ker}z}(\mathrm{J}mz)]$. Лемма доказана.

Так как $\mathrm{Ker}x \simeq C_{p^\alpha}$, то непосредственным следствием леммы 6 является

Лемма 7. Если $H(x) = \{0\}$, то $|[x]_\sigma|$ является степенью числа p.

В силу $\mathrm{J}mx \simeq C_m$ и (I0) из леммы 6 следует:

Следствие 8. Если $H(x) \neq \{0\}$, то существует единственное $i_0 \in \{1, \ldots, n\}$ со свойством $p = q_{i_0}$.

Фиксируем этот i_0 для дальнейших рассуждений.

Лемма 9. Если $H(x) = \{0\}$, то $\mathfrak{z}(x) \leqslant Z(\mathrm{End}G)$.

Доказательство. Пусть $H(x) = \{0\}$. По лемме 6 тогда $(m, p) = 1$ и поэтому $\langle a \rangle$ является единственной силовской p-подгруппой группы G. Пусть $y \in \mathfrak{z}(x)$ и $z \in \mathrm{End}G$. Тогда $ay = a^i$, $az = a^j$ при некоторых $i, j \in Z_{p^\alpha}$ и $a(yz) = a(zy)$.

По определению $\jmath(x)$ имеем $yx = xy = 0$, т.е. $b(xy) = (bx)y = by = b0 = 1$. Так как $\langle b\rangle$ – холловская p'-подгруппа группы $G = \langle a\rangle \lambda \langle b\rangle$ и группа G разрешима, то $bz = (a^{-\kappa}ba^{\kappa})^{l} = a^{-\kappa}b^{l}a^{\kappa}$ для некоторых $\kappa \in Z_{p^\alpha}$ и $l \in Z_m$. Теперь имеем $b(yz) = 1z = 1$, $b(zy) = (a^{-\kappa}b^{l}a^{\kappa})y = (ay)^{-\kappa}(ay)^{\kappa} = 1$, $b(yz) = b(zy)$. Следовательно, $yz = zy$ и $\jmath(x) \leq Z(\mathrm{End}\, G)$. Лемма доказана.

В силу равенства (I0) $\mathrm{Jm}x = \langle b\rangle = \langle b_1\rangle \times \ldots \times \langle b_n\rangle$, где $b = b_1 \cdots b_n$, $\langle b_i\rangle \simeq C_{q_i^{\beta_i}}$. Получим полупрямое разложение

$$G = [\langle a\rangle \lambda (\prod_{j=1, j\neq i}^{n} \langle b_j\rangle)] \lambda \langle b_i\rangle , \qquad (I4)$$

$i = 1, \ldots, n$. Обозначим через x_i идемпонент, соответствующий полупрямому разложению (I4), т.е.

$$\mathrm{Ker}\, x_i = \langle a\rangle \lambda (\prod_{j=1, j\neq i}^{n} \langle b_j\rangle), \quad \mathrm{Jm}x_i = \langle b_i\rangle .$$

Тогда

$$x = x_1 \times \ldots \times x_n \qquad (I5)$$

(см. [I], следствие I.II) и

$$K(x_i) \simeq \mathrm{End}\, C_{q_i^{\beta_i}}, \qquad (I6)$$

ибо $K(x_i) \simeq \mathrm{End}(\mathrm{Jm}x_i)$ (см. [I], лемма I.6). Разложение (I5) определено условием (I6) однозначно. Для каждого $i \in \{1, \ldots, n\}$ обозначим $x_i^* = x_1 \times \ldots \times x_{i-1} \times x_{i+1} \times \ldots \times x_n$, т.е.

$$\mathrm{Ker}\, x_i^* = \mathrm{Ker}\, x \lambda \mathrm{Jm}x_i, \quad \mathrm{Jm}x_i^* = \prod_{j=1, j\neq i}^{n} \mathrm{Jm}x_j .$$

Следствие I0. Если $H(x) \neq \{0\}$, то $H(x_i) = \{0\}$ для каждого $i \in \{1, \ldots, n\} \setminus \{i_0\}$.

Доказательство. Пусть $H(x) \neq \{0\}$. Тогда $p = q_{i_0}$ и поэтому $(|\mathrm{Ker}\, x_i|, |\mathrm{Jm}x_i|) = 1$ при каждом $i \in \{1, \ldots, n\} \setminus \{i_0\}$. По лемме 6 $H(x_i) = \{0\}$. Следствие доказано.

Лемма II. Если $H(x) \neq \{0\}$, то $H(x) = H(x_{i_0})$ и $|[x_{i_0}^*]_\sigma|$ является степенью числа p.

Доказательство. Известно, что для каждого $z \in J(G)$ имеет место $H(z) = \{y \in \mathrm{End}\, G \mid \mathrm{Jm}y \leq \mathrm{Ker}\, z \leq \mathrm{Ker}\, y\}$ (см. [2], лемма 2). Поэтому $H(x)$ состоит из таких $y \in \mathrm{End}\, G$, для которых

$$ay = 1, \quad by = a^{i}, \quad i \in Z_{p^\alpha}; \qquad (I7)$$

Поскольку $\langle b\rangle = \prod_{j=1}^{n} \langle b_j\rangle$ и $o(b_j) = q_j^{\beta_j}$, то при $y \in H(x)$ из (I7) вытекает, что $b_j y = 1$, $j \neq i_0$, $by = b_{i_0} y = a^{i}$, $\mathrm{Ker}\, x_{i_0} \leq \mathrm{Ker}\, y$,

$Jmy = \langle a^{i} \rangle \leqslant Ker x_{i_0}$, т.е. $y \in H(x_{i_0})$ и $H(x) \leqslant H(x_{i_0})$. Наоборот, если $y \in H(x_{i_0})$, то $Ker x \leqslant Ker x_{i_0} \leqslant Ker y$ и $Jmy = \langle b_{i_0} y \rangle \leqslant Ker x_{i_0} = \langle a \rangle \lambda (\prod_{j=1, j \neq i_0}^{n} \langle b_j \rangle)$. Поскольку $b_i y$ – p-элемент и все p-элементы группы $Ker x_{i_0}$ принадлежат $\langle a \rangle = Ker x$, то $Jmy \leqslant Ker x$. Следовательно, $Jmy \leqslant Ker x \leqslant Ker y$, $y \in H(x)$ и $H(x_{i_0}) \leqslant H(x)$. Этим доказано равенство $H(x) = H(x_{i_0})$.

Покажем, что $|[x^*_{i_0}]_\sigma|$ – степень числа p. Отметим, что $Ker x^*_{i_0} = \langle a \rangle \lambda Jmx_{i_0} = \langle a \rangle \lambda \langle b_{i_0} \rangle$ и $Jmx^*_{i_0} = \prod_{j=1, j \neq i_0}^{n} \langle b_j \rangle$ являются соответственно силовской p-подгруппой и холловской p′-подгруппой группы G. Поэтому по лемме 6 $|[x^*_{i_0}]_\sigma| = [Ker x^*_{i_0} : C_{Ker x^*_{i_0}}(Jmx^*_{i_0})]$. Так как $Jmx_{i_0} \leqslant C_{Ker x^*_{i_0}}(Jmx^*_{i_0})$, то $|[x^*_{i_0}]_\sigma| = [\langle a \rangle : C_{\langle a \rangle}(Jmx^*_{i_0})]$ и в силу $o(a) = p^\alpha$ число $|[x^*_{i_0}]_\sigma|$ является степенью числа p. Лемма доказана.

Лемма I2. Если $H(x) \neq \{0\}$, то существует $y \in K(x)$, для которого $|H(x) \cap \mathfrak{J}(y)| = p$.

Доказательство леммы I2 аналогично доказательству леммы 6 работы [2], ибо $p = q_{i_0} | m$.

Лемма I3. Если $ab \neq ba$, то $\mathfrak{J}(x) \cap J(G) = \{0\}$.

Доказательство. Пусть $z \in \mathfrak{J}(x) \cap J(G)$. Тогда $zx = xz = 0$ и $Jmz \leqslant Ker x$, $Jmx \leqslant Ker z$. Поэтому в силу $z \in J(G)$ имеем $Ker x = (Ker x \cap Ker z) \lambda Jmz$. Группа $Ker \infty = \langle a \rangle \simeq C_{p^\alpha}$ имеет только тривиальное разложение в полупрямое произведение. Следовательно, $Ker x \cap Ker z = \langle 1 \rangle$ или $Jmz = \langle 1 \rangle$. Если $Ker x \cap Ker z = \langle 1 \rangle$, то $Ker x = Jmz$, откуда ввиду $Jmx \leqslant Ker z$ и $G = Ker x \lambda Jmx = Ker z \lambda Jmz$ получим $Jmx = Ker z \lhd G$. Отсюда следует, что $G = Ker x \times Jmx = \langle a \rangle \times \langle b \rangle$, что противоречит предположению $ab \neq ba$. Поэтому $Jmz = \langle 1 \rangle$, $z = 0$ и $\mathfrak{J}(x) \cap J(G) = \{0\}$. Лемма доказана.

Лемма I4. Если $p = 2$, то группа G определяется своей полугруппой эндоморфизмов.

Доказательство. Если $p = 2$, то $G = \langle a \rangle \lambda \langle b \rangle$, $\langle a \rangle \simeq C_{2^\alpha}$ и $|Aut \langle a \rangle| = 2^{\alpha - 1}$. Поэтому $\hat{b}$ является 2-элементом группы $Aut G$ и все 2′-элементы группы G содержатся в центре $Z(G)$ группы G. Следовательно, силовская 2-подгруппа груп-

пы G имеет вид $\langle a\rangle \lambda \langle b^{i}\rangle$ для некоторого i и $G = (\langle a\rangle \lambda \langle b^{i}\rangle) \times \langle b^{j}\rangle$ для некоторого j. Группы $\langle a\rangle \lambda \langle b^{i}\rangle$ и $\langle b^{j}\rangle$ определяются своими полугруппами эндоморфизмов (см. [2], следствие 25; [I], теорема 4.2). По теореме I.I3 работы [I] группа G определяется своей полугруппой эндоморфизмов. Лемма доказана.

Лемма I5. Порядок $o(r)$ элемента r мультипликативной группы кольца $Z_{p^{\alpha}}$ равен

$$o(r) = \frac{m \cdot |\mathfrak{D}(x)|}{|\{z \in C_{End G}(x) \mid \langle z\rangle \cap K(x) = \phi\}|}. \qquad (I8)$$

Доказательство. Докажем сначала, что группа G имеет $m\,|\mathfrak{D}(x)/o(r)$ эндоморфизмов вида

$$b \longmapsto b^{j+1},\ a \longmapsto a^{i},\ j \in Z_m,\ i \in Z_{p^{\alpha}},\ (i,p) = 1. \qquad (I9)$$

Нетрудно проверить, что среди отображений вида (I9) сохраняют определяющие соотношения (9) группы G лишь те, для которых $r^{j} \equiv 1 \pmod{p^{\alpha}}$. Следовательно, отображение (I9) является эндоморфизмом тогда и только тогда, когда $r^{j} \equiv 1 \pmod{p^{\alpha}}$, т.е. $j \equiv 0 \pmod{o(r)}$. Так как $r^{m} \equiv 1 \pmod{p^{\alpha}}$ (см. [5], с. I75), то для выбора j имеем $m/o(r)$ различных способов. Группа $\mathfrak{D}(x)$ состоит из обратимых элементов полугруппы $P(x)$. Поэтому в силу (II) ясно, что $\mathfrak{D}(x)$ состоит из эндоморфизмов вида (I9), где $j = 0$ и для выбора i имеем $|\mathfrak{D}(x)|$ различных способов. Отсюда следует, что группа G имеет $m\,|\mathfrak{D}(x)|/o(r)$ эндоморфизмов вида (I9).

Покажем теперь, что эндоморфизмы вида (I9) образуют множество $\mathfrak{M} = \{z \in C_{End G}(x) \mid \langle z\rangle \cap K(x) = \phi\}$. Пусть эндоморфизм z группы G имеет вид (I9), т.е. $bz = b^{j+1}$, $az = a^{i}$. Поскольку $bx = b$, $ax = 1$, то ясно, что $xz = zx$, $z \in C_{End G}(x)$. При этом $az^{l} = a^{i^{l}} \neq 1$ для каждого l, т.е. $z^{l} \notin K(x)$ (см. [I], лемма I.6) и $\langle z\rangle \cap K(x) = \phi$. Следовательно, каждый эндоморфизм вида (I9) принадлежит множеству $\mathfrak{M}$. Пусть, наоборот, $z \in \mathfrak{M}$. Тогда $zx = xz$ и в силу леммы I.5 работы [I] $bz = b^{j+1}$, $az = a^{i}$ при некоторых $j \in Z_m$, $i \in Z_{p^{\alpha}}$. Если $(i,p) \neq 1$, то $i : p$ и $az^{\alpha} = a^{i^{\alpha}} = 1$, $(\mathrm{Ker}\, x)z^{\alpha} = \langle 1\rangle$, т.е. $z^{\alpha} \in K(x)$ (см. [I], лемма I.6) и $\langle z\rangle \cap K(x) \neq \phi$, что противоречит предположению $z \in \mathfrak{M}$. Поэтому $(i,p) = 1$ и z имеет вид (I9). Этим доказано, что эндоморфизмы вида (I9) образуют

множество $\mathfrak{M}$. Теперь ясно, что $|\mathfrak{M}| = m \cdot |\mathfrak{D}(x)| / o(r)$, откуда сразу вытекает равенство (I8). Лемма доказана.

3. Определяемость группы G своей полугруппой эндоморфизмов

В данном пункте покажем, что группа $G = \langle a \rangle \lambda \langle b \rangle$, заданная определяющими соотношениями (9), определяется своей полугруппой эндоморфизмов. Согласно лемме I4 при $p = 2$ группа G определяется своей полугруппой эндоморфизмов. Поэтому предположим, что $p > 2$. Сохраняем в этом пункте обозначения предыдущего пункта.

Если $ab = ba$, то $G = \langle a \rangle \times \langle b \rangle$ и группа G определяется своей полугруппой эндоморфизмов (см. [I], теорема 4.2). Поэтому предположим еще, что $ab \neq ba$.

Пусть G_1 -группа и $\mathrm{End}G \simeq \mathrm{End}G_1$. Докажем, что $G \simeq G_1$. Через $\bar{\gamma}$ будем обозначать эндоморфизм группы G_1, соответствующий эндоморфизму $\gamma \in \mathrm{End}G$ при изоморфизме $\mathrm{End}G \simeq \mathrm{End}G_1$. Отметим, что из конечности полугруппы $\mathrm{End}G_1$ следует конечность группы G_1 (см. [4], теорема 2). Доказательство изоморфизма $G \simeq G_1$ проведем в нескольких леммах.

В силу лемм 3-5,7,9,II-I3 и следствия I0 идемпонент $\bar{x} \in J(G_1)$, соответствующий при изоморфизме идемпоненту x, удовлетворяет условиям:

I) $K(\bar{x}) \simeq \mathrm{End}\, C_m$;

2) $P(\bar{x}) \simeq \mathrm{End}\, C_{p^\alpha}$;

3) $V(\bar{x}) \simeq B \lambda \mathfrak{D}(\bar{x})$, B - циклическая p-группа;

4) если $H(\bar{x}) = \{0\}$, то $|[\bar{x}]_\sigma|$ является степенью числа p;

5) если $H(\bar{x}) = \{0\}$, то $\rfloor(\bar{x}) \leq z(\mathrm{End}\, G_1)$;

6) $\rfloor(\bar{x}) \cap J(G_1) = \{0\}$;

7) $\bar{x} = \overline{x_1 \times \ldots \times x_n} = \bar{x}_1 \times \ldots \times \bar{x}_n$, $K(\bar{x}_i) \simeq \mathrm{End}\, C_{q_i^{\beta_i}}$, $\bar{x}^* = \bar{x}_1 \times \ldots \times \bar{x}_{i-1} \times \bar{x}_{i+1} \times \ldots \times \bar{x}_n$;

8) если $H(\bar{x}) \neq \{0\}$, то $H(\bar{x}) = H(\bar{x}_{i_0})$ и $H(\bar{x}_i) = \{0\}$, $i \neq i_0$;

9) если $H(\bar{x}) \neq \{0\}$, то $|[\bar{x}^*_{i_0}]_\sigma|$ является степенью числа p;

I0) если $H(\bar{x}) \neq \{0\}$, то существует $\bar{y} \in K(\bar{x})$, для которого $|H(\bar{x}) \cap \rfloor(\bar{y})| = p$.

Лемма I6. Имеет место

$$G_1 = \mathrm{Ker}\bar{x} \lambda \mathrm{Im}\bar{x},\ \mathrm{Im}\bar{x} = \langle d \rangle \simeq C_m,\ \mathrm{Im}\bar{x}_i = \langle d_i \rangle \simeq C_{q_i^{\beta_i}}. \quad (20)$$

Группа G_1 разрешима и коммутант G_1' группы G_1 коммутативен.

Доказательство. Разложение $G_1 = \mathrm{Ker}\bar{x} \lambda \mathrm{Im}\bar{x}$ следует из идемпотентности $\bar{x}$. В силу $K(\bar{x}) \simeq \mathrm{End}(\mathrm{Im}\bar{x})$ (см.[I] лемма I.6) и условие I) имеем $\mathrm{End}(\mathrm{Im}\bar{x}) \simeq \mathrm{End}C_m$. Отсюда следует, что $\mathrm{Im}\bar{x} \simeq C_m$, ибо каждая конечная абелева группа определяется своей полугруппой эндоморфизмов (см. [I],теорема 4.2). Поэтому $\mathrm{Im}\bar{x} = \langle d \rangle \simeq C_m$ при некотором $d \in \mathrm{Im}\bar{x}$. Аналогично в силу условия 7) $\mathrm{Im}\bar{x}_i = \langle d_i \rangle \simeq C_{q_i^{\beta_i}}$.

По лемме 4 работы [2] $\hat{G}_1 \leq V(\bar{x})$. Так как $\mathfrak{D}(\bar{x})$ является подгруппой полугруппы $P(\bar{x})$, то по условиям 2) и 3) $V(\bar{x})' \leq B$, т.е. $\hat{G}_1' = \hat{G}_1' \leq V(\bar{x})' \leq B$ и $\hat{G}_1' \simeq G_1' \cdot Z(G_1)/Z(G_1)$ -циклическая p-группа. Отсюда следует, что коммутант G_1' группы G_1 коммутативен и группа G_1 разрешима. Лемма доказана.

Лемма I7. $G_1 \neq \mathrm{Ker}\bar{x} \times \mathrm{Im}\bar{x}$.

Доказательство. Если $G_1 = \mathrm{Ker}\bar{x} \times \mathrm{Im}\bar{x}$, то из изоморфизма $\mathrm{End}G \simeq \mathrm{End}G_1$ следует, что $G = \mathrm{Ker}x \times \mathrm{Im}x = \langle a \rangle \times \langle b \rangle$ (см. [I], следствия I.8 и I.I2). Это противоречит предположению $ab \neq ba$. Лемма доказана.

В силу коммутативности группы G_1' имеет место равенство

$$G_1' = P \times Q, \quad (2I)$$

где P - p-группа и Q - p'-группа. При этом $Q \leq Z(G_1)$, ибо $G_1' \cdot Z(G_1)/Z(G_1)$ - циклическая p-группа. Пусть P^* - силовская p-подгруппа и Q^* - холловская p'-подгруппа группы G_1 (ее существование гарантирует разрешимость группы G_1). Тогда $Q \leq Q^*$, ибо $Q \leq Z(G_1)$ и холловские подгруппы в разрешимых группах сопряжены. Предположим, что $P \leq P^*$.

Лемма I8. $G_1 = P^* \lambda Q^*$.

Доказательство. Достаточно доказать, что $P^* \lhd G_1$. Пусть $u \in P^*$, $g \in G_1$. Тогда $g^{-1}ug = u[u,g]$ и $[u,g] = hc$, $h \in P \leq P^*$, $c \in Q \leq Z(G_1)$, ибо имеет место разложение (2I). Поскольку $u[u,g] = uhc$ является p-элементом и $uh \in P^*$, $c \in Z(G_1)$, то $c = 1$ и $g^{-1}ug = uh \in P^*$. Следовательно, $P^* \lhd G_1$. Лемма доказана.

В силу разложения $G_1 = P^* \lambda Q^*$ силовская p-подгруппа P^* группы G_1 определена однозначно.

Лемма 19. Если $H(\bar{x}) = \{0\}$, то $\mathrm{Ker}\bar{x}$ является p-группой.

Доказательство. Пусть $H(\bar{x}) = \{0\}$. По лемме 6 $|[\bar{x}]_\sigma| = [\mathrm{Ker}\bar{x} : C_{\mathrm{Ker}\bar{x}}(\mathrm{Im}\bar{x})]$. По условию 4)

$$[\mathrm{Ker}\bar{x} : C_{\mathrm{Ker}\bar{x}}(\mathrm{Im}\bar{x})] = p^k. \qquad (22)$$

Если $k = 0$, то $G_1 = \mathrm{Ker}\bar{x} \times \mathrm{Im}\bar{x}$, что противоречит лемме 17. В силу $k > 0$ и (22) $|\mathrm{Ker}\bar{x}| \vdots p$. По лемме 6 $p \nmid |\mathrm{Im}\bar{x}|$. Поэтому $P^* \leq \mathrm{Ker}\bar{x}$ и можно предполагать, что $\langle d \rangle = \mathrm{Im}\bar{x} \leq Q^*$. Отсюда следует разложение $Q^* = (Q^* \cap \mathrm{Ker}\bar{x}) \lambda \mathrm{Im}\bar{x}$, ибо $G_1 = \mathrm{Ker}\bar{x} \lambda \mathrm{Im}\bar{x}$. Следовательно,

$$G_1 = P^* \lambda Q^* = P^* \lambda [(Q^* \cap \mathrm{Ker}\bar{x}) \lambda \mathrm{Im}\bar{x}]. \qquad (23)$$

Обозначим через y идемпонент, соответствующий полупрямому разложению (23). Из включения $P^* \leq \mathrm{Ker}\bar{x}$ и равенства (23) легко вывести, что $y\bar{x} = \bar{x}y = \bar{x}$, т.е. $y \in P(\bar{x})$. По условию 2) $P(\bar{x})$ содержит 2 идемпонента: $1, \bar{x}$. Если $y = 1$, то $P^* = \langle 1 \rangle$, что противоречит $|\mathrm{Ker}\bar{x}| \vdots p$. Следовательно, $y = \bar{x}$, $Q^* \cap \mathrm{Ker}\bar{x} = \langle 1 \rangle$ и $P^* = \mathrm{Ker}\bar{x}$, т.е. $\mathrm{Ker}\bar{x}$ является p-группой. Лемма доказана.

Лемма 20. Если $H(\bar{x}) \neq \{0\}$, то $\mathrm{Ker}\bar{x}$ является p-группой.

Доказательство. Предположим, что $H(\bar{x}) \neq \{0\}$. Покажем, что $|\mathrm{Ker}\bar{x}| \vdots p$. Ясно, что также $H(x) \neq \{0\}$. Тогда $q_{i_0} = p$ и $\mathrm{Im}\bar{x}_{i_0} \simeq C_{p^{\beta_{i_0}}}$. По условию 7) $\mathrm{Im}\bar{x} = \prod_{i=1}^{n} \mathrm{Im}\bar{x}_i$, $\mathrm{Ker}\bar{x}_i = \mathrm{Ker}\bar{x} \lambda (\prod_{j=1, j\neq i} \mathrm{Im}\bar{x}_j)$.

Так как порядки групп $\mathrm{Im}\bar{x}_i$, $1 \leq i \leq n$, попарно взаимно просты, то

$$(|\mathrm{Im}\bar{x}_i|, |\mathrm{Ker}\bar{x}_i|) = (|\mathrm{Im}\bar{x}_i|, |\mathrm{Ker}\bar{x}|). \qquad (24)$$

По условию 8) $H(\bar{x}) = H(\bar{x}_{i_0}) \neq \{0\}$. По лемме 6 $(|\mathrm{Im}\bar{x}_{i_0}|, |\mathrm{Ker}\bar{x}_{i_0}|) \neq 1$, откуда в силу равенства (24) и $\mathrm{Im}\bar{x}_{i_0} \simeq C_{p^{\beta_{i_0}}}$ имеем $|\mathrm{Ker}\bar{x}| \vdots p$.

Аналогично из равенств $H(\bar{x}_i) = \{0\}$, $i \neq i_0$, можно при помощи леммы 6 и равенства (24) вывести, что $(|\mathrm{Ker}\bar{x}|, |\mathrm{Im}\bar{x}_i|) = 1$. Поэтому

$$(|\mathrm{Im}\bar{x}^*_{i_0}|, |\mathrm{Ker}\bar{x}^*_{i_0}|) = 1, \qquad (25)$$

ибо $\mathrm{Im}\bar{x}^*_{i_0} = \prod_{j=1, j\neq i_0}^{n} \mathrm{Im}\bar{x}_j$ и $\mathrm{Ker}\bar{x}^*_{i_0} = \mathrm{Ker}\bar{x} \lambda \mathrm{Im}\bar{x}_{i_0}$.

При этом $|\mathrm{Ker}\,\bar{x}^*_{i_0}| : p$, ибо $|\mathrm{Ker}\,\bar{x}| : p$. Отсюда следует, что силовская p-подгруппа P^* группы G_1 содержится в $\mathrm{Ker}\,\bar{x}^*_{i_0}$:

$$P^* \leq \mathrm{Ker}\,\bar{x}^*_{i_0} \tag{26}$$

и $\mathrm{Jm}\,\bar{x}^*_{i_0}$ – p'-группа.

Из равенства (25) по лемме 6 следует, что $|[\bar{x}^*_i]_\sigma| = [\mathrm{Ker}\,\bar{x}^*_{i_0} : C_{\mathrm{Ker}\,\bar{x}^*_{i_0}}(\mathrm{Jm}\,\bar{x}^*_{i_0})] = [\mathrm{Ker}\,\bar{x} : C_{\mathrm{Ker}\,\bar{x}}(\mathrm{Jm}\,\bar{x}^*_{i_0})]$. Из условия 9) и последнего равенства следует, что $\mathrm{Ker}\,\bar{x}$ имеет такую свою холловскую p'-подгруппу Q_1, для которой $Q_1 \leq C_{G_1}(\mathrm{Jm}\,\bar{x}^*_{i_0})$. Так как $\mathrm{Jm}\,\bar{x} = \mathrm{Jm}\,\bar{x}^*_{i_0} \times \mathrm{Jm}\,\bar{x}_{i_0}$, $G_1 = \mathrm{Ker}\,\bar{x} \lambda \mathrm{Jm}\,\bar{x}$, $\mathrm{Jm}\,\bar{x}_{i_0} \simeq C_{p^{\beta_{i_0}}}$ и $\mathrm{Ker}\,\bar{x} \cap \mathrm{Jm}\,\bar{x}^*_{i_0} = \langle 1 \rangle$, то ясно, что $\langle Q_1, \mathrm{Jm}\,\bar{x}^*_{i_0} \rangle = Q_1 \times \mathrm{Jm}\,\bar{x}^*_{i_0}$ является холловской p'-подгруппой группы G_1. По лемме I8 $G_1 = P^* \lambda (Q_1 \times \mathrm{Jm}\,\bar{x}^*_{i_0})$, т.е.

$$G_1 = (P^* \lambda \mathrm{Jm}\,\bar{x}^*_{i_0}) \lambda Q_1 . \tag{27}$$

Обозначим через z идемпотент, соответствующий полупрямому разложению (27). Тогда $\mathrm{Jm}\,z = Q_1 \leq \mathrm{Ker}\,\bar{x}$, $\mathrm{Jm}\,\bar{x} = \mathrm{Jm}\,\bar{x}_{i_0} \times \mathrm{Jm}\,\bar{x}^*_{i_0} \leq P^* \lambda \mathrm{Jm}\,\bar{x}^*_{i_0} = \mathrm{Ker}\,z$, ибо $\mathrm{Jm}\,\bar{x}_{i_0} \simeq C_{p^{\beta_{i_0}}}$ и P^* – единственная силовская p-подгруппа группы G_1. Следовательно, $z\bar{x} = \bar{x}z = 0$ и $z \in \mathfrak{J}(\bar{x}) \cap \mathfrak{J}(G_1)$. Согласно условию 6) $z = 0$, т.е. $Q_1 = \langle 1 \rangle$ и $\mathrm{Ker}\,\bar{x}$ является p-группой. Лемма доказана.

Из лемм I9 и 20 следует:

Следствие 2I. Группа $\mathrm{Ker}\,\bar{x}$ является p-группой.

Лемма 22. Если $\mathfrak{J}(\bar{x}) = \{0\}$, то $\mathrm{Ker}\,\bar{x}$ является циклической p-группой.

Доказательство. Пусть $\mathfrak{J}(\bar{x}) = \{0\}$. Как отмечено, G_1' – коммутативна и $G_1' \cdot Z(G_1)/Z(G_1) \simeq G_1'/(G_1' \cap Z(G_1)) = \langle g_0 (G_1' \cap Z(G_1)) \rangle$ – циклическая p-группа ($g_0 \in G_1'$). При этом $G_1' \leq \mathrm{Ker}\,\bar{x}$, ибо $\mathrm{Jm}\,\bar{x} = \langle d \rangle$ – коммутативна. При $G_1' < \mathrm{Ker}\,\bar{x}$ имеет место разложение

$$G_1/G_1' = \langle d G_1' \rangle \times \langle g_1 G_1' \rangle \times \ldots \times \langle g_s G_1' \rangle \tag{28}$$

для некоторых $g_1, \ldots, g_s \in \mathrm{Ker}\,\bar{x} \setminus G_1'$, $s \geq 1$, и $y \in \mathfrak{J}(\bar{x})$, $y \neq 0$, где $dy = g_2 y = \ldots = g_s y = 1$, $G_1' y = \langle 1 \rangle$, $g_1 y = g_1^{o(g_1)/p}$. Следовательно, $G_1' = \mathrm{Ker}\,\bar{x}$. Отсюда следует, что $\mathrm{Ker}\,\bar{x} \leq \langle g_0, Z(G_1) \rangle$ и

$$G_1 = \langle \mathrm{Jm}\,\bar{x},\ g_0, Z(G_1) \rangle = \langle d, g_0, Z(G_1) \rangle \tag{29}$$

Из равенства (29) следует, что $\langle d, g_0 \rangle \lhd G_1$ и $G_1/\langle d, g_0 \rangle$ коммутативна, т.е. $G_1' = \mathrm{Ker}\,\bar{x} \leq \langle d, g_0 \rangle$ и

$$G_1 = \langle d, g_0 \rangle . \tag{30}$$

В силу $G_1 = \mathrm{Ker}\bar{x} \lambda \mathrm{Jm}\bar{x} = G_1' \lambda \mathrm{Jm}\bar{x}$ имеем $d^{-1}g_0 d = g_0^t c$. $c \in G_1' \cap Z(G_1)$. Если $t \equiv 1 \pmod p$, то группа $G_1 / \langle G_1' \cap Z(G_1), g_0^p \rangle$ коммутативна и $G_1' \leq \langle G_1' \cap Z(G_1), g_0^p \rangle$. По выбору g_0 ясно, что $\langle G_1' \cap Z(G_1), g_0^p \rangle < G_1'$. Полученное противоречие показывает, что $t \not\equiv 1 \pmod p$ и $t - 1 \not\equiv 0 \pmod p$.

Так как $d^{-1}(g_0^{t-1}c)d = (d^{-1}g_0 d)^{t-1} c = (g_0^t c)^{t-1} c = g_0^{(t-1)t} c^t = (g_0^{t-1}c)^t$ и $\langle g_0^{t-1} c \cdot (G_1' \cap Z(G_1)) \rangle = \langle g_0^{t-1} (G_1 \cap Z(G_1)) \rangle = \langle g_0 (G_1' \cap Z(G_1)) \rangle$, то можно предполагать, что $c = 1$ и

$$d^{-1} g_0 d = g_0^t. \tag{31}$$

Теперь из равенств (30) и (31) следует $G_1 = \langle g_0 \rangle \lambda \langle d \rangle$, т.е. $\mathrm{Ker}\bar{x} = \langle g_0 \rangle$ – циклическая p-группа. Лемма доказана.

Лемма 23. Если $H(\bar{x}) = \{0\}$, то $\mathrm{Ker}\bar{x}$ является циклической p-группой.

Доказательство. Пусть $H(\bar{x}) = \{0\}$. Для доказательства леммы достаточно показать, что $\mathrm{Ker}\bar{x}$ содержит лишь одну подгруппу порядка p (см. [3], теорема I2.5.2). Можно предположить, что $\jmath(\bar{x}) \neq \{0\}$ (см. лемму 22). Покажем неравенство $G_1' < \mathrm{Ker}\bar{x}$ ($G_1' \leq \mathrm{Ker}\bar{x}$, ибо $\mathrm{Jm}\bar{x}$ коммутативна). Существует $\bar{y} \in \jmath(\bar{x})$, $\bar{y} \neq 0$, т.е. $\bar{y}\bar{x} = \bar{x}\bar{y} = 0$, $\mathrm{Jm}\bar{x} \leq \mathrm{Ker}\bar{y}$. В силу разрешимости группы G_1 существует такой $N \lhd G_1$, что $G_1 > N \geq \mathrm{Ker}\bar{y}$ и G_1/N коммутативна. Поэтому $G_1' \leq N$. Если $G_1' = \mathrm{Ker}\bar{x}$, то $\mathrm{Ker}\bar{x} \leq N$, $\mathrm{Jm}\bar{x} \leq \mathrm{Ker}\bar{y} \leq N$ и $G_1 \leq N$, $G_1 = N$, что противоречит выбору N. Следовательно, $G_1' < \mathrm{Ker}\bar{x}$ и имеет место разложение (28).

Для каждого элемента $h \in \mathrm{Ker}\bar{x}$, $o(h) = p$, построим $\varphi_h \in \jmath(\bar{x})$ следующим образом:

$$\varphi_h : G_1 \to G_1 / G_1' \to \langle g_1 G_1' \rangle \to \langle h \rangle,$$

$$G_1' \varphi_h = \langle 1 \rangle, \quad d\varphi_h = g_2 \varphi_h = \dots = g_s \varphi_h = 1, \quad g_1 \varphi_h = h.$$

Ясно, что $\mathrm{Jm}\varphi_h = \langle h \rangle \leq \mathrm{Ker}\bar{x}$, $\mathrm{Jm}\bar{x} = \langle d \rangle \leq \mathrm{Ker}\varphi_h$, т.е. $\bar{x} \cdot \varphi_h = \varphi_h \cdot \bar{x} = 0$, $\varphi_h \in \jmath(\bar{x})$. По условию 5) $\varphi_h \cdot \hat{g} = \hat{g} \cdot \varphi_h$ для каждого $g \in G_1$. По лемме I.5 работы [I] $(\mathrm{Jm}\varphi_h)\hat{g} = \langle g^{-1} h g \rangle \leq \mathrm{Jm}\varphi_h = \langle h \rangle$. Следовательно,

$$\langle h \rangle \lhd G_1 \tag{32}$$

при каждом $h \in \mathrm{Ker}\bar{x}$, $o(h) = p$.

Предположим, что $\mathrm{Ker}\bar{x}$ обладает двумя различными подгруппами $\langle h_1 \rangle$ и $\langle h_2 \rangle$ порядка p. Тогда в силу (32) h_1, $h_2 \in Z(\mathrm{Ker}\bar{x})$, ибо каждый нормальный делитель порядка p в

p-группе содержится в ее центре (см. [3], теорема 4.3.4). Согласно условию (32) $d^{-1}h_1d=h_1^{t_1}$, $d^{-1}h_2d=h_2^{t_2}$, $d^{-1}(h_1h_2)d=(h_1h_2)^{t_3}=h_1^{t_3}h_2^{t_3}$ для некоторых t_1, t_2, t_3 ($\mathrm{Im}\bar{x}=\langle d\rangle$). Тогда $d^{-1}(h_1h_2)d=d^{-1}h_1d\cdot d^{-1}h_2d=h_1^{t_1}h_2^{t_2}=h_1^{t_3}h_2^{t_3}$, откуда $t_1=t_2=t_3=t$. Отсюда следует, что $d^{-1}hd=h^t$ при каждом $h\in\mathrm{Ker}\bar{x}$, $o(h)=p$, где t не зависит от h.

Предположим, что $t\neq 1$. Тогда $\mathrm{Ker}\bar{x}\cap Z(G_1)=\langle 1\rangle$. В частности, $G_1'\cap Z(G_1)=\langle 1\rangle$. Поэтому $G_1'/G_1'\cap Z(G_1)\simeq G_1'$ и группа G' циклична: $G_1'=\langle h_0\rangle$. Если группа $\mathrm{Ker}\bar{x}$ содержит элемент h порядка p вне G_1', то $d^{-1}hd=h^t$, $d^{-1}hd=b[h,d]$ и $h^tG_1'=hG_1'$. т.е. $t=1$, что противоречит предположению $t\neq 1$. Следовательно, $t=1$ и $h_1, h_2\in Z(G_1)$ (ведь $G_1=\mathrm{Ker}\bar{x}\lambda\langle d\rangle$; $h_1, h_2\in Z(\mathrm{Ker}\bar{x})$).

В силу разложения (28) каждый элемент группы G_1 имеет вид $d^jg_1^lh$, $h\in\langle g_2,\dots,g_s,G_1'\rangle$. Этот вид единственный, если предполагать, что $0\leqslant l<o(g_1G_1')$. Если еще $d^{j'}g_1^{l'}h'\in G_1$, $l'<o(g_1G_1')$, то ввиду (28)

$$(d^jg_1^lh)(d^{j'}g_1^{l'}h')=d^{j+j'}g_1^{l+l'}h_3$$

для некоторого $h_3\in\langle g_2,\dots,g_s,G_1'\rangle$. Рассмотрим отображения τ_{h_i} ($i=1,2$):

$$(d^jg_1^lh)\tau_{h_i}=d^jg_1^l(h_ih).$$

Так как $h_i^p=1$ и $o(g_1G_1')$ – степень числа p, то ясно, что $\tau_{h_i}\in\mathrm{End}G_1$. Так как $d\tau_{h_i}=d$, $g_1\tau_{h_i}=g_1h_i$, $h\tau_{h_i}=h(h\in\langle g_2,\dots,g_s,G_1'\rangle)$, то $\tau_{h_i}\in\mathrm{Aut}G_1$. Поскольку $\tau_{h_i}|\mathrm{Im}\bar{x}=1_{\mathrm{Im}\bar{x}}$ и $(\mathrm{Ker}\bar{x})\tau_{h_i}=\mathrm{Ker}\bar{x}$, то нетрудно убедиться, что $\tau_{h_i}\in\mathfrak{D}(\bar{x})$.

По построению $o(\tau_{h_i})=p$ и $\langle\tau_{h_1}\rangle\cap\langle\tau_{h_2}\rangle=\langle 1\rangle$, т.е. $\mathfrak{D}(\bar{x})$ содержит две различные подгруппы порядка p. С другой стороны, $\mathfrak{D}(\bar{x})$ является группой обратимых элементов полугруппы $P(\bar{x})\simeq\mathrm{End}C_{p^\alpha}$ (см. условие 2)), т.е. $\mathfrak{D}(\bar{x})\simeq\mathrm{Aut}C_{p^\alpha}\simeq C_{p^{\alpha-1}(p-1)}$ (см. [6], теорема 9.I) и $\mathfrak{D}(\bar{x})$ содержит лишь одну подгруппу порядка p. Полученное противоречие показывает, что группа $\mathrm{Ker}\bar{x}$ имеет только одну подгруппу порядка p. Лемма доказана.

Лемма 24. Если $H(\bar{x})\neq\{0\}$, то $\mathrm{Ker}\bar{x}$ является циклической группой.

Доказательство. Пусть $H(\bar{x})\neq\{0\}$. По условию I0) существует $\bar{y}\in K(\bar{x})$, для которого $|H(\bar{x})\cap J(\bar{y})|=p$. Эндоморфизм $\bar{y}$ имеет вид

$$(\mathrm{Ker}\,\bar{x})\,\bar{y} = \langle 1\rangle,\ d\bar{y} = d^{i},\ i \in \mathbb{Z}_m$$

(см. [I], лемма I.6). По условию 8) $H(\bar{x}) = H(\bar{x}_{i_0})$, откуда $|H(\bar{x}_{i_0}) \cap \rfloor(\bar{y})| = p$. Ранее было отмечено, что $q_{\bar{y}i_0} = p$, $\mathrm{Im}\,\bar{x}_{i_0} = \langle d_{i_0}\rangle \simeq C_{p^{\beta i_0}}$, $\mathrm{Ker}\,\bar{x}_{i_0} = \mathrm{Ker}\,\bar{x} \lambda (\prod_{j=1, j\neq i_0} \langle d\rangle)$. Согласно лемме 2 работы [2] $H(\bar{x}) = H(\bar{x}_{i_0}) = \{\bar{z} \in \mathrm{End}\,G_1 \mid (\mathrm{Ker}\,\bar{x}_{i_0})\bar{z} = \langle 1\rangle, d_{i_0}\bar{z} \in \mathrm{Ker}\,\bar{x}\}$.

Так как при $\bar{z} \in H(\bar{x})$ имеем

$$d(\bar{y}\bar{z}) = d^{i}\bar{z} = (d\bar{z})^{i},\ d(\bar{z}\bar{y}) = 1,$$

$$(\mathrm{Ker}\,\bar{x})(\bar{y}\bar{z}) = \langle 1\rangle, (\mathrm{Ker}\,\bar{x})(\bar{z}\bar{y}) = \langle 1\rangle,$$

то $\bar{y}\bar{z} = \bar{z}\bar{y} = 0$ и $\bar{z} \in H(\bar{x}) \cap \rfloor(\bar{y})$ тогда и только тогда, когда $(d\bar{z})^{i} = 1$. Следовательно,

$$H(\bar{x}) \cap \rfloor(\bar{y}) = H(\bar{x}_{i_0}) \cap \rfloor(\bar{y}) = \{\bar{z} \in \mathrm{End}\,G_1 \mid (\mathrm{Ker}\,\bar{x}_{i_0})\bar{z} = \langle 1\rangle, \quad d\bar{z} = d_{i_0}\bar{z} \in \mathrm{Ker}\,\bar{x},\ (d_{i_0}\bar{z})^{i} = 1\}. \tag{33}$$

Поскольку множество $H(\bar{x}) \cap \rfloor(\bar{y})$ содержит ненулевой элемент $\bar{z}$ и $d_{i_0}\bar{z}$ – неединичный p-элемент, то $i : p$. Теперь ясно, что для каждого элемента $h \in \mathrm{Ker}\,\bar{x}$, $h^{p} = 1$, эндоморфизм $\bar{u}$, для которого

$$\bar{u} : G_1 = \mathrm{Ker}\,\bar{x}_{i_0} \lambda \mathrm{Im}\,\bar{x}_{i_0} \to \mathrm{Im}\,\bar{x}_{i_0} = \langle d_{i_0}\rangle \to \langle h\rangle,$$

$$(\mathrm{Ker}\,\bar{x}_{i_0})\,\bar{u} = \langle 1\rangle,\ \ d_{i_0}u = h,$$

является элементом из $H(\bar{x}) \cap \rfloor(\bar{y})$. Отсюда в силу $|H(\bar{x}) \cap \rfloor(\bar{y})| = p$ следует, что группа $\mathrm{Ker}\,\bar{x}$ содержит лишь одну подгруппу порядка p, т.е. является циклической. Лемма доказана.

Из лемм 23 и 24 следует:

Следствие 25. Группа $\mathrm{Ker}\,\bar{x}$ является циклической p-группой.

Лемма 26. Группы G и G_1 изоморфны.

Доказательство. По следствию 25 $\mathrm{Ker}\,\bar{x} = \langle c\rangle \simeq C_{p^{\gamma}}$ при некотором $\gamma \geq 1$. Аналогично лемме 4 $P(\bar{x}) \simeq \mathrm{End}\,C_{p^{\gamma}}$. По условию 2) $P(\bar{x}) \simeq \mathrm{End}\,C_{p^{\alpha}}$. Следовательно, $\alpha = \gamma$ и $\mathrm{Ker}\,\bar{x} = \langle c\rangle \simeq C_{p^{\alpha}}$. Так как $G_1 = \mathrm{Ker}\,\bar{x} \lambda \mathrm{Im}\,\bar{x}$ и $\mathrm{Im}\,\bar{x} = \langle d\rangle \simeq C_m$, то группа G_1 задается следующими определяющими соотношениями:

$$d^{m} = c^{p^{\alpha}} = 1,\ \ d^{-1}cd = c^{\nu},\ \ \nu \in \mathbb{Z}_{p^{\alpha}}.$$

Аналогично лемме I5

$$o(\nu) = \frac{m \cdot |\mathfrak{D}(\bar{x})|}{|\{\bar{z} \in C_{\mathrm{End}\,G_1}(\bar{x}) \mid \langle \bar{z}\rangle \cap K(\bar{x}) = \emptyset\}|}.$$

Поэтому в силу $\mathrm{End}G \simeq \mathrm{End}G_1$ получим $o(r)=o(v)$. Из равенства $o(r)=o(v)$ следует уже изоморфизм $G \simeq G_1$ (см.[5],лемма 7). Лемма доказана.

Сформулируем результаты пунктов 2 и 3 в виде теоремы:

Теорема 27. Каждая конечная группа G, являющаяся полупрямым произведением $G=\langle a\rangle \lambda \langle b\rangle$ двух своих циклических подгрупп $\langle a\rangle$ и $\langle b\rangle$, $\langle a\rangle \simeq C_{p^\alpha}$, p - произвольное простое число, определяется своей полугруппой эндоморфизмов.

Следствие 28. Пусть $G=\langle a\rangle \lambda \langle b\rangle$, где выполнены условие (9), и $\mathrm{Ker}x=\langle a\rangle$, $\mathrm{Im}x=\langle b\rangle$, $x\in J(G)$. Если $\mathrm{End}G \simeq \mathrm{End}G_1$, то $G_1=\mathrm{Ker}\bar{x}\lambda\mathrm{Im}\bar{x}$ и $\mathrm{Ker}\bar{x} \simeq \mathrm{Ker}x$, $\mathrm{Im}\bar{x} \simeq \mathrm{Im}x$, где $\bar{x}$ - элемент, соответствующий при изоморфизме идемпотенту x.

Если $p>2$, то при $ab=ba$ утверждение следствия 28 следует сразу, а при $ab\neq ba$ оно следует из следствия 25 и лемм 3 и 26. Если $p=2$, то следствие 28 можно вывести из леммы I4 и из теоремы I8 работы [2].

Литература

I. Пуусемп П. Идемпотенты полугрупп эндоморфизмов групп. - Уч. зап. Тартуск. гос. ун-та, I975, 366, 76–I04.

2. Пуусемп П. Полугруппа эндоморфизмов полупрямого произведения двух циклических p-групп. - Уч. зап. Тартуск. гос. ун-та, I976, 390, I04–I33.

3. Холл М. Теория групп. М. I962.

4. Alperin, J.L. Groups with finitely many automorphisms.-Pacif. J. Math., 1962, 12, N 1, 1-5.

5. Basmaji, B.G. On the isomorphisms of two metacyclic groups. - Proc. Amer. Math. Soc., 1969, 22, N 1, 175-182.

6. Passman, D. Permutation groups. New York, 1968.

P. Puusemp

Über die Bestimmung des halbdirekten Produktes der zyklischen Gruppen durch seine Endomorphismenhalbgruppe

Zusammenfassung

In vorliegenden Arbeit zeigen wir, daß die endgültige Gruppe $G = \langle a \rangle \lambda \langle b \rangle$, wo $o(a) = p^{\alpha}$, p – eine Primzahl ist, durch ihre Endomorphismenhalbgruppe eindeutig bestimmt ist. Wir zeigen auch das Vorhandensein der nichtisomorphen Gruppen $G = \langle a \rangle \lambda \langle b \rangle$ und $H = \langle c \rangle \lambda \langle d \rangle$, die den Bedingungen $o(a) = o(c) \neq p^{\alpha}$, $o(b) = o(d)$ und $EndG \simeq EndH$ genügen.

№ 482

TALLINNA POLÜTEHNILISE INSTITUUDI TOIMETISED

ТРУДЫ ТАЛЛИНСКОГО ПОЛИТЕХНИЧЕСКОГО ИНСТИТУТА

УДК 681.3I:5I9.8

А.В. Мищенко

ОБ ОДНОЙ ЗАДАЧЕ УПОРЯДОЧЕНИЯ ОБСЛУЖИВАНИЯ РАБОТ НА ОДНОЙ МАШИНЕ

I. Постановка задачи. Существует n работ, которые должны пройти обслуживание на одной машине. Одновременно на машине может обслуживаться только одна работа. Пусть $t_i (i=1,2,\ldots,n)$ - продолжительность обслуживания i-й работы. Обобщение рассматриваемой задачи по сравнению с традиционной постановкой [I] состоит в следующем. Представим множество работ машины N как $N = B_1 \cup B_2 \cup \ldots \cup B_q$. Машина состоит из q узлов, причем j-й узел предназначен для обслуживания работ из множества B_j. Переход от обслуживания работы m к обслуживанию работы l, где $(m,l) \in \beta_j$, требует перенастройки j-го узла, потребляющей $\tau^{(j)}_{m,l}$ единиц времени. Другой особенностью рассматриваемой задачи является то, что операция по перенастройке узлов происходит во время обслуживания машиной работ при помощи других узлов, что влечет за собой беспрерывное обслуживание работ машиной. Задача заключается в составлении последовательности обслуживания n работ на одной машине, при которой минимизируются суммарные затраты времени на перенастроечные операции. Заметим, что порядок обслуживания работ каждого узла зависит от порядка обслуживания работ другими узлами. Это вытекает из того, что в каждый момент времени может обслуживать работу только один из q узлов. Исходя из этого нельзя рассматриваемую задачу представить в виде q изолированных задач нахождения оптимальной последовательности обслуживания работ каждым узлом.

2. Алгоритм решения задачи. Рассмотренная задача является оптимизационной задачей и может быть описана в терминах линейного программирования. Однако размерность полу-

ченной модели линейного программирования не оставляет надежд на решение задач практического уровня сложности. В качестве альтернативного подхода решения задачи рассмотрим переборный подход. Очевидно, что всего имеется $n!$ различных последовательностей обслуживания работ. Причем не любая последовательность обслуживания работ является допустимой. Условие допустимости требует, чтобы в последовательности работ на обслуживание промежуток времени между обслуживаниями любых двух работ, использующих один и тот же узел, был бы не меньше, чем время, требуемое на соответствующую перенастройку. Так как обслуживание работ машиной происходит беспрерывно, то условие доступности для последовательности $i_1, i_2, \ldots, i_K$ можно выразить следующим соотношением

$$\tau^{(j)}_{i_r, i_s} \leq t_{i_{r+1}} + t_{i_{r+2}} + \ldots + t_{i_{s-1}},$$

для всех $(i_r, i_s) \in B_j$, где $1 \leq r < s \leq K$ и $j = 1, 2, \ldots, g$.

Теперь поставленную задачу можно представить как задачу нахождения среди всех допустимых последовательностей обслуживания n работ последовательности, требующей минимальных затрат времени на перенастроенные операции. Решение этой задачи методом полного перебора не представляется возможным, в связи со слишком большими вычислительными затратами. Для сокращения перебора в процессе поиска оптимального решения предлагается использовать метод ветвей и границ (МВГ) [2].

При использовании МВГ для решения конкретной задачи необходимо конкретизировать применительно к этой задаче правила ветвления и определения оценок нижних границ множества решений. Предполагается, что в рассматриваемой задаче исходная вершина $\mathfrak{D}$ дерева поиска оптимального решения представляет из себя множество всех последовательностей обслуживания работ, в том числе и недопустимых. Ветвление исходного множества происходит путем фиксирования в последовательности работ на обслуживания номера работы, которая будет обслуживаться первой. Таким образом, ветвление первого яруса можно представить как $\mathfrak{D} = \mathfrak{D}_1 \cup \mathfrak{D}_2 \cup \ldots \cup \mathfrak{D}_n$. Далее для выбранной вершины фиксируется вторая работа и т.д. Таким образом, в качестве некоторой вершины К-го яруса $\mathfrak{D}_{i_1, i_2, \ldots, i_K}$ будет выступать множество всех последователь-

ностей обслуживания, в которых предполагается начинать с обслуживания k работ $i_1, i_2, \ldots, i_k$. Обозначим через $Z(\mathcal{D}_{i_1, i_2, \ldots, i_k})$ оценку нижней границы суммарных затрат времени на перенастроечные операции для последовательностей обслуживания, входящих в $\mathcal{D}_{i_1, i_2, \ldots, i_k}$. Величину $Z(\mathcal{D}_{i_1, i_2, \ldots, i_k})$ будем определять по следующим правилам

$$Z(\mathcal{D}_{i_1, i_2, \ldots, i_k}) = M,$$

если $i_1, i_2, \ldots, i_k$ представляет из себя недопустимую последовательность обслуживания работ. Здесь M – любое число, большее чем значение максимального из суммарных затрат времени на переналадочные операции среди всех допустимых последовательностей обслуживания n работ,

$$Z(\mathcal{D}_{i_1, i_2, \ldots, i_k}) = C^1_{i_1, i_2, \ldots, i_k} + C^2_{i_1, i_2, \ldots, i_k}$$

– в противном случае. Здесь $C^1_{i_1, i_2, \ldots, i_k}$ – продолжительность времени для перенастройки узлов, необходимая для обслуживания последовательности работ $i_1, i_2, \ldots, i_k$; $C^2_{i_1, i_2, \ldots, i_k}$ – продолжительность времени, требуемого на перенастройки машины, необходимая для производства еще незафиксированных $n-k$ работ. Последняя величина определяется в предположении, что в каждом узле эти перенастройки будут производиться в оптимальном для этого узла порядке без учета выбранных порядков обслуживания работ другими узлами. При рассмотрении порядка обслуживания работ в каждом узле независимо от порядков обслуживания работ других узлов задачу определения оптимальной последовательности обслуживания работ в каждом узле можно представить в виде известной в комбинаторике задачи коммивояжора. В роли городов, в которых необходимо побывать коммивояжору, выступает множество работ, которые должен обслужить данный узел, в роли расстояний между городами выступает длительность соответствующих перенастроек. Рассмотренная задача неоднократно ставилась и имеются подходящие для нашего случая алгоритмы ее решения.

3. Числовой пример. Проиллюстрируем работу процедуры на простом числовом примере. Пусть требуется составить расписание обслуживания на машине восьми работ. Машина состоит из двух узлов. Первый узел обслуживает работы I,2,3,4 – второй – 5,6,7,8. Длительности перенастроек узлов и обслуживания работ заданы в таблице I.

I	I	2	3	4
I	-	5,0	5,7	3,7
2	5,8	-	5,0	4,9
3	2,5	2,4	-	2,0
4	2,I	6,4	4,4	-

Таблица перенастроек для первого узла

	5	6	7	8
5	-	I,5	3,8	3,2
6	I,I	-	I,9	2,8
7	4,2	3,5	-	2,3
8	2,8	3,3	2,4	-

Таблица перенастроек для второго узла

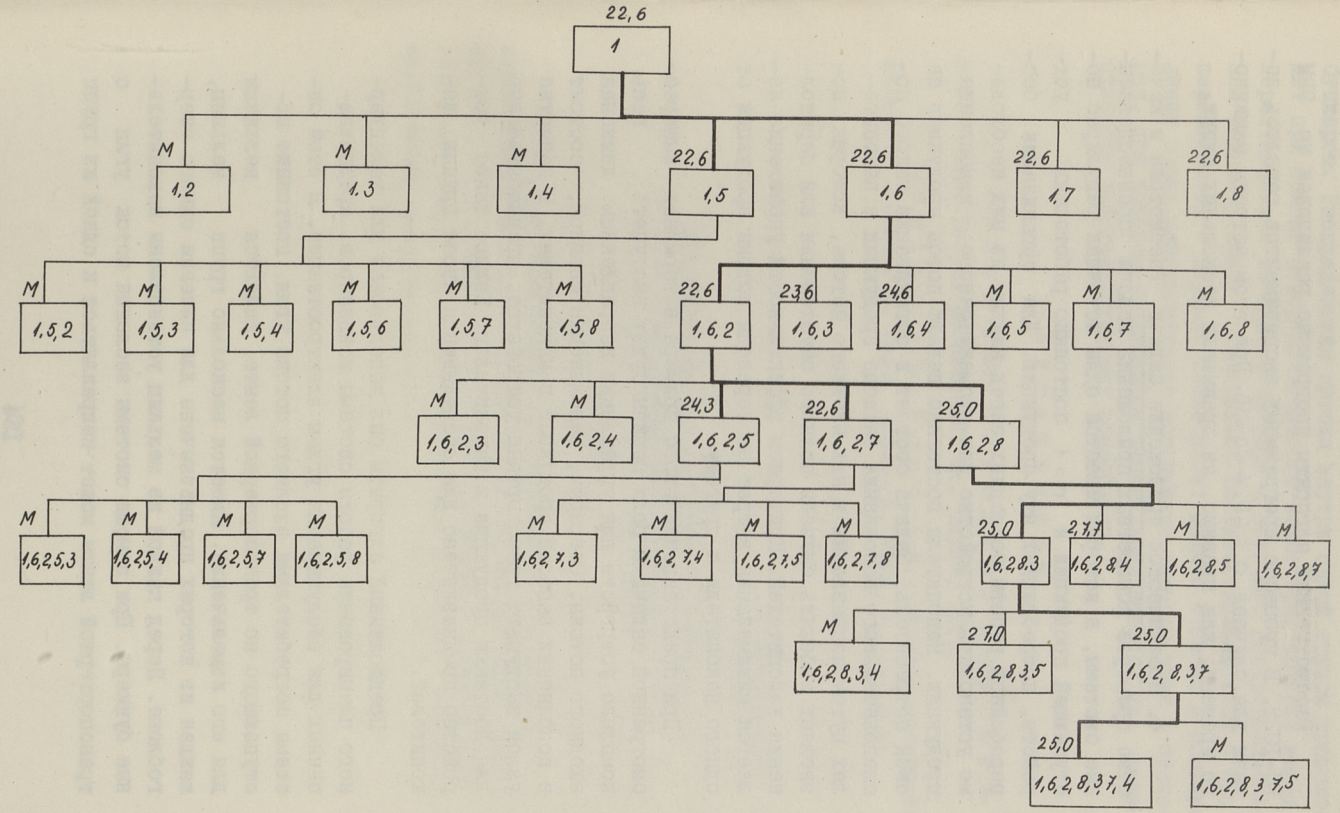
Таблица I

Исходная информация для числового примера

№ работы	I	2	3	4	5	6	7	8
Продолжительность обслуживания	8,0	8,0	8,0	8,0	2,2	6,0	2,2	6,0

Применяемую в МВГ схему поиска оптимального решения можно представить состоящей из двух этапов. Первый этап реализует нахождение решения, являющегося начальным приближением к оптимальному решению. На фиг. I изображено дерево поиска, сформулированное в результате первого этапа. Прямоугольники на фигуре соответствуют множествам расписаний, цифры внутри прямоугольников - фиксированной последовательности работ $i_1, i_2, \ldots, i_k$ цифры над прямоугольниками - оценкам нижних границ соответствующих множеств расписаний. Жирными линиями выделены пройденные на первом этапе пути поиска оптимального решения. Как видно из фигуры, найденное начальное приближение к оптимальному решению соответствует последовательности обслуживания работ I-6-2-8-3-7-4-5, при этом затрачивается 25,0 единиц времени на необходимые перенастройки.

На втором этапе процедуры осуществляется попытка улучшения найденного решения путем обратного хода на вышестоящие уровни с последующим, если это целесообразно, ветвлением подмножеств этих уровней и повторным спуском вниз по ветвям. Найденное в результате этой процедуры оптимальное решение соответствует последовательности I-8-2-6-3-5-4-7, при этом затраты на перенастройки составят 24,6 единиц времени.



Фиг. 1. Фрагмент дерева поиска оптимального расписания.

Рассмотренный алгоритм программно реализован на ЭВМ ЕС-1022. В процессе программных экспериментов решались задачи для случаев $n \leqslant 40, 4 \leqslant q \leqslant 10$. При этом затраты машинного времени, как правило, не превышали нескольких минут.

4. Практическое приложение задачи. Рассмотрим в качестве примеров конкретных приложений задачи производственные системы, в которых имеется один источник регулярно поступающей продукции и $n > 1$ постоянно работающих устройств, потребляющих эту продукцию. Для организации бесперебойной работы последних перед каждым из них необходимо установить устройство для промежуточного накопления продукции. Накопители постоянно выдают поток продукции на свой потребитель. Задача состоит в организации такого последовательного пополнения запасов продукции в накопителях путем направления в них входного потока, которая позволяет избежать случаев полного опустошения или переполнения накопителей. Критерием эффективности управления является минимизация затрат на переводы потока продукции от одного накопителя к другому.

При представлении этого примера в терминах вышерассмотренной задачи в роли машины будет выступать весь комплекс устройств для управления направлением движения входного потока. Операции, выполняемые машиной, состоят в пополнении запасов продукции в накопителях. В качестве узлов выступают транспортные устройства, предназначенные для доставки продукции к накопителям. Каждое такое устройство предназначено для обслуживания своей группы накопителей.

Предложенный алгоритм был использован для календарного планирования работы системы конвейеров, предназначенных для обслуживания углем электростанции. В этой системе потребителями входного потока угля, постоянно поступающего по транспортерной ленте, являются мельницы для его измельчения. Имеется несколько групп мельниц, каждая из которых предназначена для питания одного энергоблока. Перед каждой из мельниц установлены накопительные бункеры. При помощи системы заслонок поток угля с транспортерной ленты может направляться к одной из групп

бункеров. Распределение потока внутри одной группы происходит при помощи распределительного конвейера, который может переставляться от одного бункера к другому. Распределительный конвейер можно представить соответствующим узлу, а его перестановку - перенастройке узла. Задача состоит в составлении плана своевременного пополнения запасов угля в бункерах, при котором достигаются минимальные затраты на перестановки распределительных конвейеров.

Литература

I. Конвей Р.В., Максвелл В.П., Миллер Л.В. Теория расписаний. М., Наука, 1975.

2. Lawler, E.L., Wood, D.E., Branch- and Bound Methods. - Survey A. Oper. Research, 1966, V. 14, N 4.

A. Mishchenko

Sequencing n Jobs on a Single Machine: a Branch-and-bound Solution

Summary

This paper deals with the generalized one-machine scheduling problem, where the criterion is the minimization of the time for machine adjusting. A branch-and-bound algorithm and simple numerical example are proposed. Practical application examples are given.

Содержание

I. А.О. Вооглайд, Д.Б. Лийб. Древовидная переменная в обработке данных. 3

2. А.О. Вооглайд, Д.Б. Лийб. Полная оптимизация памяти анализаторов предшествования I5

3. М.В. Лепп. Схема трансляции, реализуемая при помощи СПТ ТПИ. 3I

4. Л.К. Выханду. О некоторых методах упорядочения объектов и признаков в системе данных . . 43

5. Л.К. Выханду, Т.Ф. Лучковский, Я.Я. Тепанди. Система хранения и обработки дискретной информации (СХОДИ). 5I

6. Ю.Г. Лааст-Лаас. О проектировании логической структуры баз данных экономического анализа . 63

7. Т.В. Арак. Об аппроксимации n-кратных сверток распределений, имеющих неотрицательную характеристическую функцию, безгранично делимыми распределениями. 8I

8. Х.Ф. Симанова. Имитационное моделирование поточного производства на банке данных. 87

9. В.А. Йокк. Обзор технологии программирования. I0I

I0. Л.К. Выханду, Э.Х.-Т. Кунапуу. Обработка социально-экономических показателей в диалоговом режиме. II3

II. Э.А.-Ю. Юби. О задаче минимизации функции вероятности. I2I

I2. П. Пуусемп. Об определяемости полупрямого произведения циклических групп своей полугруппой эндоморфизмов. I3I

I3. А.В. Мищенко. Об одной задаче упорядочения обслуживания работ на одной машине. I49

руб. 1.30