Министерство высшего и среднего специального
образования Р С Ф С Р

Куйбышевский ордена Трудового Красного Знамени
авиационный институт имени академика С.П.Королева

В.Д. Е л е н е в

ЛАБОРАТОРНЫЙ ПРАКТИКУМ ПО ИСПОЛЬЗОВАНИЮ САПР

У т в е р ж д е н о
редакционно-издательским
советом института
в качестве
учебного пособия
для студентов

Куйбышев 1987

УДК 681.3

Е л е н е в В.Д. Лабораторный практикум по использованию САПР. – Куйбышев: КуАИ, 1987, с. 68.

Лабораторный практикум составлен по курсу "Системы автоматиаированного проектирования".

В нем рассматриваются вопросы разработки и практического использования подсистем САПР. Приводятся описания программных средств отображения графической информации, диалоговых и информационно-поисковых систем, элементов машинной графики.

Пособие разработано на кафедре летательных аппаратов и предназначено студентам факультета летательных аппаратов;может быть также полезно и для студентов других специальностей при выполнении лабораторных работ по данному курсу.

Ил. 10. Табл. 2. Библиогр. – 11 назв.

Рецензенты: д-р физ.-мат.наук Г.И.Б ы к о в ц е в,
канд.техн.наук А.В. С о л л о г у б

ОБЩИЕ МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ РАБОТ

Все работы выполняются с использованием ЭВМ. На первом занятии со студентами проводится инструктаж по технике безопасности. Материалы очередной лабораторной работы должны быть проработаны студентами самостоятельно во внеаудиторное время. Каждому студенту выдается индивидуальное задание.

Лабораторные работы, как правило, состоят из двух этапов. На первом этапе проводятся подготовительные работы: разработка алгоритмов и программ, решение практических задач и т.д. Второй этап связан с непосредственной работой на ЭВМ.

Каждая лабораторная работа заканчивается оформлением отчета. Если это требуется, то необходимо также продемонстрировать преподавателю работу разработанных или используемых программных средств. Студент должен уметь ответить на контрольные вопросы, приведенные в конце описания лабораторной работы.

В лабораторных работах используется программное обеспечение, разработанное В.В.Неретиным.

ОБРАБОТКА ГРАФИЧЕСКОЙ ИНФОРМАЦИИ В САПР

Цель работы: получение практических навыков построения графических изображений с использованием пакета графических программ ГРАФОР.

I. ОБЩИЕ ПОЛОЖЕНИЯ

Одной из основных задач систем автоматизированного проектирования является снижение трудоемкости выпуска чертежно-графической документации. Как показывает опыт эксплуатации подсистем САПР на ведущих предприятиях как в нашей стране, так и за рубежом, эффективность использования САПР особенно велика при выполнении таких работ, как изготовление типовых сборочных чертежей, деталировка, компоновка, внесение изменений. Во многом это объясняется тем, что для этих видов работ удается формализовать большую часть выполняемых операций.

Процесс получения графической информации может состоять из нескольких этапов. Предварительно требуемое изображение формируется на экране графического дисплея и сохраняется в базе данных. Затем, по мере необходимости, оно выбирается из базы данных и после соответствующих преобразований выводится на бумажный носитель. В настоящее время широко используется специальное программное обеспечение, позволяющее вычерчивать как на экране дисплея, так и на бумажном носителе различные примитивы: прямые, дуги окружности, кривые более высоких порядков, типовые элементы. Одним из широко используемых устройств вывода на бумажные носители графической и текстовой информации в САПР являются графопостроители. Вычерчивание изображений проводится графопостроителями с использованием специального языка описания изображений. Для графопостроителей существует два способа описа-

ния изображения: в инкрементальном режиме и на языке графических приказов. И н к р е м е н т а л ь н ы й р е ж и м - это, по сути дела, машинный язык, каждая команда которого обеспечивает выполнение элементарной операции: поднятие или опускание пишущего узла, элементарное его перемещение в одном из восьми допустимых направлений и т.д. К преимуществам такого способа следует отнести возможность вычерчивания изображения любой сложности. Недостатком является очень большой объем программы описания изображения.

Практически все изображения на бумажном носителе формируются с использованием я з ы к а г р а ф и ч е с к и х п р и к а з о в. В основу этого режима положено задание только узловых координат (или опорных точек), а их интерполяцию выполняет само устройство. Это позволяет значительно сократить трудоемкость составления программы описания изображения и ее объем, повышает наглядность программы и облегчает процесс внесения в нее изменений. В языке графических приказов предусмотрены также возможности для создания типов вычерчиваемых линий, вывода текстовой информации, масштабирования чертежа и т.д. С использованием языка графических приказов в настоящее время разработано несколько пакетов программ. Наибольшее распространение из них получил пакет программ ГРАФОР, являющийся графическим расширением алгоритмического языка Фортран.

Для получения требуемого изображения необходимо составить программу на Фортране, предусмотрев в ней обращения к подпрограммам ГРАФОРА. Используя подпрограммы ГРАФОРа, можно описать практически любое изображение будучи свободным от необходимости знать внутренний язык графопостроителей.

Обращение к подпрограммам ГРАФОРа осуществляется с использованием оператора

CALL имя (список),

где имя - имя подпрограммы;
список - список фактических аргументов.

2. ОСНОВНЫЕ ОПЕРАТОРЫ ГРАФОРа

2.1. Выбор единицы измерения

Перед формированием изображения необходимо выбрать одну из допустимых единиц измерений: миллиметры, сантиметры, дюймы. Для этого со-

ответственно используются программы *MMS* , *CMS* или *INCHES* ,не имеющие списков фактических параметров. Если ни одна из этих подпрограмм не вызывается, то по умолчанию единицей измерения устанавливается сантиметр. Все углы в ГРАФОРе определяются в градусах.

2.2. Определение страницы

Любая программа начинается с определения страницы, в которой будет формироваться изображение. При этом предполагается, что начало системы координат находится в левом нижнем углу страницы. Здесь же устанавливается начальное положение пишущего узла (пера графплоттера).

Обращение к подпрограмме:

CALL PAGE (XL, YL, NAME, N, J),

где *XL* ; *YL* – размеры страницы вдоль осей *x* и *y* ;

NAME – название страницы (удобно задавать литеральной константой);

N – количество символов в названии;

J – признак очерчивания границы: *J* = 0 – граница не очерчивается, *J* = 1 – граница очерчивается.

Для перехода к формированию новой страницы и по завершении программы на ГРАФОРе требуется закрыть доступ к странице с помощью подпрограммы *ENDPG* .

Обращение к программе:

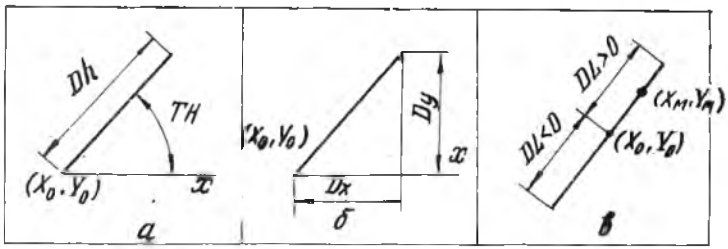
CALL ENDPG (NUM),

где *NUM* – литеральная константа, содержащая текст, разделяющий страницы.

2.3. Описание прямых линий

При вычерчивании прямых линий считается, что начальной (или текущей) опорной точкой является точка, где предварительно было остановлено перо графплоттера. Вторая опорная точка определяется соответствующими параметрами, задаваемыми пользователем. В зависимости от способа задания линии может использоваться несколько программ.

П р о г р а м м а *MOVA* . Строит отрезок по его длине *DL* и углу *TH* , образованному прямой с осью *x* (рис.I,а).



Р и с. I

Обращение к программе:

CALL MOVA(DL, TH, J).

П р о г р а м м а *MOVB* . Строит отрезок по приращениям координат *DX* и *DY* вдоль осей *x* и *y* , соответственно (рис.I,б). Обращение к программе:

CALL MOVB(DX, DY, J).

П р о г р а м м а *MOVC* . Строит отрезок по его длине *DL* и точке (*XM* , *YM*), лежащей на отрезке или его продолжении (рис.I,в). Обращение к программе:

CALL MOVC (XM, YM, DL, J)

при *DL* > 0 перо из начальной точки движется в сторону точки (*XM* , *YM*), а при *DL* < 0 – противоположную сторону.

П р о г р а м м а *MOVE* . Строит отрезок по координатам конечной точки.

Обращение к программе:

CALL MOVE(X, Y, J).

В приведенных программах параметр *J* указывает состояние пера: *J* = 0 – перо поднято (линия не вычерчивается), *J* = I – перо опущено (линия вычерчивается).

П р о г р а м м а *DASHP*. Обеспечивает вычерчивание штриховых и штрихпунктирных линий.

Обращение к программе:

CALL DASHP(X,Y,DL),

где *X*, *Y* – координаты второй опорной точки;
|DL| – длина основного штриха линии: *DL* > 0 – проводится штриховая линия; *DL* < 0 – проводится штрихпунктирная линия.

2.4. Описание окружности и ее дуг

Рассмотрим наиболее часто используемые программы построения окружности и ее дуг.

П р о г р а м м а *CIRC*. Вычерчивает окружность заданного радиуса *R* с центром в текущей точке.

Обращение к программе:

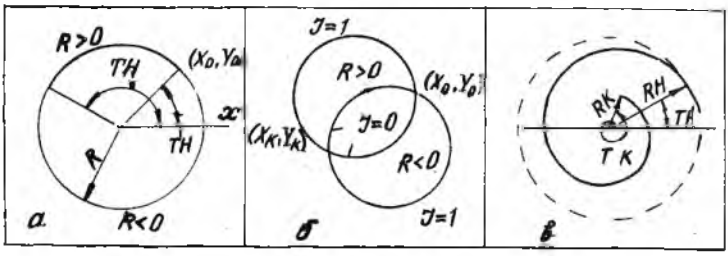
CALL CIRC(R)

После завершения процедуры перо возвращается в исходную точку.

П р о г р а м м а *ARCIA*. Проводит из текущей точки дугу окружности заданного радиуса, ограниченную лучами, образующими с осью *x* углы *TH* (начальный) и *TK* (конечный), как показано на рис.2,а.

Обращение к программе:

CALL ARCIA(R,TH,TK)



Р и с. 2

здесь $|R|$ - величина радиуса: $R > 0$ - перемещение пера из начальной точки в конечную против часовой стрелки, $R < 0$ - перемещение пера по часовой стрелке.

П р о г р а м м а *ARCIB* . Соединяет текущую точку с заданной дугой окружности заданного радиуса (рис.2,б).

Обращение к программе:

CALL ARCIB (R, XK, YK, J),

здесь $|R|$ - величина радиуса: $R > 0$ - перемещение пера из начальной точки в конечную происходит против часовой стрелки, $R < 0$ - перемещение пера по часовой стрелке; (XK , YK) - координаты конечной точки; J - признак выбора дуги: $J = 0$ - короткая дуга ($\leqslant 180^0$), $J = 1$ - длинная дуга ($\geqslant 180^0$).

П р о г р а м м а *CIRCLE*. Позволяет вычерчивать окружности, спирали и их дуги (рис. 2,в).

Обращение к программе:

CALL CIRCLE (XS, YS, TH, TK, RH, RK, L),

где XS , YS - координаты начальной точки;
TH, TK - углы наклона начального и конечного радиусов к оси x ;
RH, RK - начальный и конечный радиус;
L - признак вычерчиваемой линии: $L = 0$ - штриховая линия, $L = 1$ - сплошная линия.

2.5. Описание эллипса и его дуг

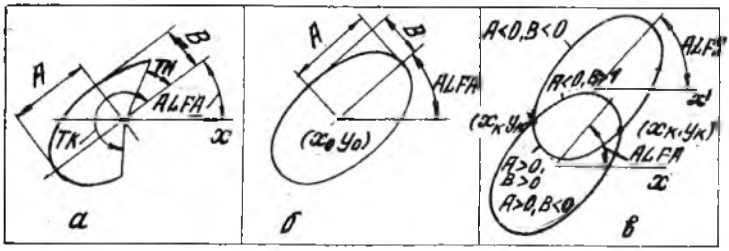
Для вычерчивания эллипсов и их дуг рассмотрим следующие программы.

П р о г р а м м а *ELIPS* . Вычерчивает эллипс или дугу эллипса.

Обращение к программе:

CALL ELIPS (X, Y, A, B, ALFA, TH, TK),

где (X, Y) - координаты начальной точки;
A, B - размеры полуосей эллипса;
$ALFA$ - угол наклона полуоси A к оси x ;
TH, TK - соответственно, углы между начальным и конечным диаметром и полуосью A (рис. 3,а).



Р и с. 3

П р о г р а м м а *ELPS* . Позволяет начертить эллипс с заданными размерами полуосей *A* и *B* с центром в текущей точке и с заданным наклоном *ALFA* полуоси *A* к оси *x* (рис.3,б).

Обращение к программе:

CALL ELPS (A, B, ALFA)

П р о г р а м м а *ARCELB*. Проводит из текущей точки в заданную точку дугу эллипса с указанными размерами полуосей и направлением рисования относительно центра (рис.3,в).

Обращение к программе:

CALL ARCELB (A,B, ALFA, XK, YK),

здесь */A/* , */B/* – размеры полуосей: *A* > 0 – перемещение пера против часовой стрелки; *A* < 0 – перемещение пера по часовой стрелке; *B* > 0 – выбирается малая дуга ($\leqslant 180^0$), *B* < 0 – выбирается большая дуга ($\geqslant 180^0$); *ALFA* – угол наклона полуоси *A* к оси *x*; *XK, YK* – координаты конечной точки.

2.6. Вычерчивание размерых линий

П р о г р а м м а *NARROW* . Чертит отрезок со стрелками на концах.

Обращение к программе:

CALL NARROW (XH, YH, XK, YK, S, IC) ,

здесь (*XH* , *YH*), (*XK* , *YK*) – соответственно координаты начальной и конечной точек; *S* – длина стрелки; *IC* – параметр, управляющий стрелкой: *IC* = I – стрелка обращена к начальной точке, *IC* = 2 – стрелка обращена к конечной точке, *IC* = 3 – стрелка указывает на оба конца.

П р о г р а м м а *DIMEN* . Позволяет изобразить размерную линию заданной длины.

Обращение к программе:

CALL DIMEN(XH, YH, DL, TH),

здесь *XH* , *YH* – координаты начальной точки;
DL – длина отрезка;
TH – угол наклона отрезка к оси *x* .

П р о г р а м м а *DARC* . Вычерчивает дугу со стрелками на концах.

Обращение к программе:

CALL DARC(XC, YC, XH, YH, A, IC),

здесь *XC* , *YC* – координаты центра;
XH, *YH* – координаты начальной точки дуги;
|A| – угловая величина дуги: *A* > 0 – дуга вычерчивается против часовой стрелки, *A* < 0 – дуга вычерчивается по часовой стрелке;
IC – параметр, управляющий стрелкой:
IC = I – стрелка обращена к начальной точке,
IC = 2 – стрелка обращена к конечной точке,
IC = 3 – стрелки указывают на оба конца.

2.7. Вывод текстовой информации

Перед выводом текстовой информации необходимо выбрать соответствующий шрифт, используя обращение

CALL SET(J),

где *J* – вариант шрифта: *J* = 0 – прописные русские и латинские буквы, цифры и знаки; *J* = I – строчные русские и прописные буквы, цифры и знаки.

Угол наклона шрифта можно задать, используя обращение

CALL ITALIC(J) ,

где J – признак курсива: $J = 0$ – прямой шрифт, $J = I$ – правый наклон, $J = -I$ – левый наклон.

Непосредственно текстовая информация выводится программой *SYMBOL*.

Обращение к программе:

CALL SYMBOL (X, Y, SIZE, TEXT, N, THETA),

где X, Y – координаты левого нижнего угла первой литеры текста или приращения к конечным координатам текста, предшествующего данному;

SIZE – высота символов;

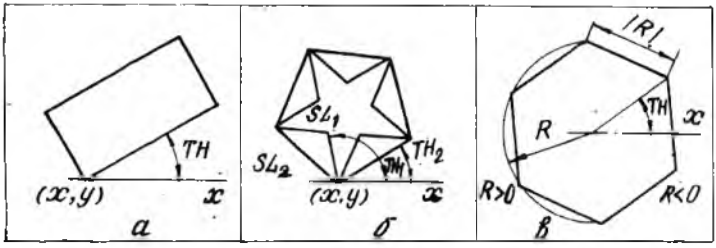
TEXT – заданный текст;

|N| – количество литер в строке: $N > 0$ (X, Y) считается координатами на странице; $N < 0$ (X, Y) считается приращениями;

THETA – угол наклона строки текста к оси x.

2.8. Описание многоугольников

Прямоугольник с поворотом на заданный угол можно вычертить с помощью программы *RECT* (рис.4,а).



Р и с. 4

Обращение к программе:

CALL RECT (X, Y, H, W, TH),

где X, Y – координаты левого нижнего угла прямоугольника;

H, W – длина строк прямоугольника;

TH – угол поворота вокруг точки (*X, Y*) относительно оси *x* ;

Правильный многоугольник вычерчивается с использованием программы *POLYG* (рис. 4,б).

Обращение к программе:

CALL POLYG(X, Y, S.L, M, TH),

где *X, Y* – координаты начальной точки;
SL – размер стороны многоугольника;
|M| – число сторон многоугольника: *M* > 0 – рисуется выпуклый многоугольник, *M* < 0 – рисуется звездчатый многоугольник;
TH – угол наклона к оси *x* стороны, с которой начинается рисование.

Выпуклый правильный многоугольник с центром в текущей точке вычерчивается программой *POLG* (рис. 4,в).

Обращение к программе:

CALL POLG (R, M, TH) ,

где *|R|* – радиус описанной окружности (*R* > 0) или сторона многоугольника (*R* < 0);
M – число сторон многоугольника;
TH – угол от оси *x* до ближайшего луча, проведенного в вершину.

2.9. Штриховка

Заштриховать участок страницы, ограниченный замкнутой кусочно-линейной ориентированной кривой, можно с помощью программы *SHADE* .

Обращение к программе:

CALL SHADE(X, Y, N, STEP, EPS, BETA),

где *X, Y* – массивы длины *N* абсцисс и ординат соответственно, задающие вершины границы;
N – число вершин границы;
STEP – расстояние между линиями штриховки в выбранных единицах измерения;

EPS – сдвиг линий штриховки относительно основной штриховки по нормали к ней;

BETA – угол наклона штриховки к оси x .

3. ИНСТРУКЦИЯ ПО РАБОТЕ НА ЭВМ

Лабораторная работа выполняется на ЭВМ СМ1420. Пакет графических программ ГРАФОР подключается к программе пользователя на этапе построения образа задачи через библиотеку *GRAFOR* , расположенную на устройстве *DK4* : в каталоге [1,1].

Пример: *> TKB FTSK=FOBJ, DK4: [1,1] GRAFOR/LB.*

здесь *FOBJ* – имя файла объектного модуля программы пользователя, полученного транслятором ФОРТРАН-IУ;

FTSK – имя файла образа задачи.

В результате выполнения программы пользователя на магнитном диске сформируется файл, содержащий программу работы графопостроителя. Имя этого файла будет совпадать с названием страницы, указанным в подпрограмме *PAGE*. Расширение файла *XY* .

Для поочередного запуска всех этапов выполнения задачи в системе (трансляция, построение образа задачи, выполнение) можно воспользоваться командой *> GRE* , на запрос которой необходимо указать имя исходного модуля программы. От того, в каком режиме используется графопостроитель: в автономном или централизованном, зависит дальнейший порядок построения изображения на бумажном носителе.

3.I. Работа графопостроителя в автономном режиме

При работе графопостроителя в автономном режиме необходимо полученный файл вывести на промежуточный носитель информации – перфоленту. Для вывода файла с магнитного диска на перфоленту используется команда: *> GRP.* При этом, на запрос системы необходимо ввести имя используемого файла.

Пример: Укажите имя файла, выводимого на перфоленту [*S*]: *<GR/S.XY>*,

здесь в символах < > приведен возможный ответ пользователя.

Затем полученная перфолента устанавливается на фотосчитыватель графопостроителя, и после включения устройства происходит формирование изображения на бумажном носителе.

3.2. Работа графопостроителя в централизованном режиме

При работе графопостроителя в централизованном режиме вывод изображения на бумажный носитель осуществляется с помощью команды *>GRS* . Как и в первом варианте, на запрос системы необходимо ввести имя используемого файла. После этого работа графопостроителя происходит под управлением ЭВМ.

4. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Продолжительность лабораторной работы - 4 ч. Работа выполняется по следующей схеме.

1. Самостоятельное изучение теоретического материала лабораторной работы во внеаудиторное время.
2. Получение индивидуального задания у преподавателя.
3. Составление и отладка программы для вычерчивания чертежа заданной детали.
4. Запуск программы на выполнение и получение файла программы работы графопостроителя.
5. Получение чертежа детали на бумажном носителе.
6. Отчет о выполненной работе.

5. СОДЕРЖАНИЕ ОТЧЕТА

1. Задание на работу.
2. Чертеж детали на бумажном носителе.
3. Листинг составленной программы с использованием пакета графических программ ГРАФОР.

Контрольные вопросы

1. Какие существуют способы изображения для графопостроителя?
2. В чем заключается преимущество языка графических приказов?
3. Для чего нужно определять страницу?
4. Какими способами может задаваться прямая линия?

5. Какими способами может описываться дуга окружности?

6. Какие есть варианты изображения отрезка со стрелками?

7. Можно ли изменять размер и наклон символов при использовании программы *SYMBOL* ?

8. Какую программу можно использовать для отображения пятиугольной звезды?

Библиографический список

Баяновский Ю.Ш., Галактионов В.А., Михайлова Т.Н. Графор. Графическое расширение Фортрана. - М.: Наука, 1985.

Горелик А.Г. Автоматизация инженерно-графических работ с помощью ЭВМ. - Минск: Вышейшая школа, 1980.

Тодорой Д.Н., Романчук Л.И., Перетятков С.М. Языки машинной графики. Справочник. - Кишинев: Картя молдовенска, 1980.

ОРГАНИЗАЦИЯ ДИАЛОГА В СИСТЕМЕ "ПОЛЬЗОВАТЕЛЬ-ЭВМ"

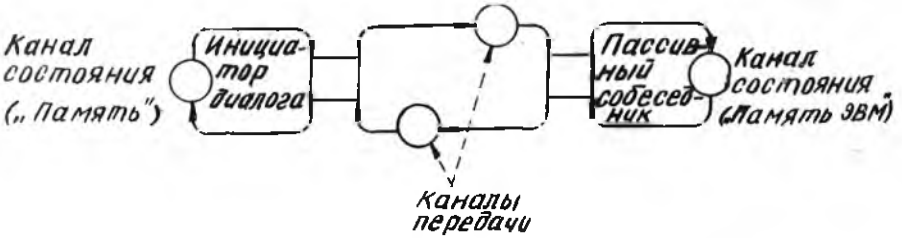
Цель работы: закрепление и углубление теоретических знаний по организации диалога в системе "пользователь-ЭВМ"; преобретение практических навыков по программной реализации диалога типа "предложение для выбора".

I. ОБЩИЕ ПОЛОЖЕНИЯ

Одним из условий широкого применения систем автоматизированного проектирования является их ориентация на пользователя - не профессионала в области вычислительной техники и программных систем.

С приходом на ЭВМ массового пользователя - прикладного специалиста - появились соответствующие требования к организации работы с программными системами, в частности, это касается и организации диалога. Под диалогом пользователя с ЭВМ мы будем понимать интерактивный обмен посланиями между пользователем и диалоговой системой в соответствии с установленным языком и формой диалога для достижения определенной задачи.

Диалоговая система состоит из двух информационных центров: инициатора диалога (пользователь) и пассивного собеседника (ЭВМ), как показано на рисунке. Каждый информационный центр имеет канал состояния



Рис

для хранения информации. Например, каналом состояния может быть мозг пользователя и его письменные заметки. Для ЭВМ канал состояния - это ее оперативная и внешняя память.

Для обмена сообщениями два информационных центра используют два канала односторонней передачи, которые обеспечиваются средствами ввода-вывода ЭВМ.

Наименьший элемент диалога между двумя участниками будем называть шагом диалога. Шаг диалога состоит из действия (выдачи сообщений одного из участников диалога) и ответа (сообщений второго участника диалога). Действие всегда составляет первую часть диалога, а ответ - вторую часть.

П р и м е р.
Действие: Будете продолжать работу?
Ответ: Да.

I.I. Требования к диалогу

В литературе можно встретить немало примеров, когда системы автоматизированного проектирования, имеющие большие возможности в своей прикладной области, не находили должного применения из-за плохо организованного диалога пользователя с ЭВМ. В основном эти диалоговые системы не учитывали психологические особенности пользователя - прикладного специалиста. Специалист в прикладной области не может тратить много времени на изучение специфики ЭВМ, языков программирования и диалоговых языков. Поэтому современные диалоговые системы должны учитывать такие особенности человека, как ограниченный объем внимания, забывчивость, потерю внимания при длительном ожидании ответа и др.

Из всей совокупности требований, обычно предъявляемых к диалогу, выделим следующие:

гибкость диалога,
ясность,
легкость пользования,
простота обучения,
надежность.

Гибкость диалога. Это требование достигается в том случае, если форма диалога не является жесткой и неизменной, когда диалоговая система учитывает разнообразие потребностей и уровней квалификации поль-

пользователей. Например, диалоговая система может предлагать различные режимы диалога с различными уровнями детализации сообщений в зависимости от квалификации пользователя и его пожеланий.

Ясность. У пользователя должна быть полная ясность в поведении системы и организации диалога. Например, это требование обеспечивается в случаях, когда:

система предоставляет список функций с хорошей структурой;

аналогичные задачи требуют аналогичных действий пользователя;

система не дает неожиданных эффектов, она проявляет одинаковое поведение в одинаковых ситуациях и т.д.

Легкость пользования. Легкость пользования складывается по существу из трех основных составляющих.

Легкость в обращении, которая обеспечивается:

расширением функций ввода-вывода;

предоставлением помощи пользователю в случае необходимости;

малым временем ожидания ответа (работа с диалоговой системой должна вестись в так называемом реальном масштабе времени, когда время реакции системы не превышает нескольких секунд);

возможностью работы с системой с небольшим количеством дополнительных объяснений и т.д.

Учет свойств пользователя, его способностей, знаний и изменчивости со временем этих качеств:

общение с системой должно проходить на языке, близком к терминам предметной области исследований;

система не вынуждает пользователя к необдуманным действиям;

система учитывает, что пользователь приобретает опыт работы с ней;

система учитывает возможность допущения пользователем ошибок, причем учитывается, что их число зависит от продолжительности работы с системой.

Устойчивость к ошибкам, которая достигается в том случае, когда:

система страхует пользователя в особенно ответственных случаях (уничтожение, преобразование данных и т.д.);

система обеспечивает диагностику ответов пользователя и предлагает ясные сообщения об обнаруженных ошибках;

сообщение об ошибках содержит указание, как их исправить;

исправление ошибок должно выполняться только в тех частях ответа, где они обнаружены.

Простота обучения. Пользователь должен иметь возможность обучаться работе с системой в процессе работы с ней. При этом простые задачи не должны требовать специальной подготовки пользователя. Желательно, чтобы система представляла пользователю инструкции по работе прямо на экран дисплея.

Надежность. Диалоговая система может считаться надежной, когда:

аварии системы происходят редко;

система предоставляет возможности защиты данных и обеспечение их секретности;

при возникновении сбойных ситуаций сохраняется "предыстория" диалога;

нет никаких побочных и скрытых эффектов.

I.2. Типы диалога

При проектировании диалоговых систем принято выделять типы используемого диалога. Рассмотрим наиболее употребительные типы диалога.

Простой запрос. Система запрашивает необходимые данные. У пользователя нет выбора, он может только ввести требуемые от него объекты. Этот тип диалога обычно используется для последовательного ввода данных.

П р и м е р.

Действие: Задайте толщину обечайки в миллиметрах.

Ответ: I,5.

Предложение для выбора. Система предлагает несколько альтернативных задач, из которых пользователь должен выбрать только одну. Существует два варианта такого типа диалога: меню, вопросы, требующие ответа: да/нет. В меню шаг диалога состоит из демонстрации возможностей выбора (действие) и указания выбранного варианта (ответ).

П р и м е р.

ДЕЙСТВИЕ: Что вы желаете?

I. Ввести данные.

2. Изменить данные.

3. Исключить данные.

4. Получить доступ к данным.

5. Закончить работу.

В зависимости от грамматики используемого языка диалога ответ может быть оформлен следующими способами:

указанием позиции выбранной задачи,

ОТВЕТ: 2

(пользователем выбран режим изменения данных);

указанием ключевого слова выбранной задачи [чаще всего ключевое слово состоит из начальных букв альтернатив (число символов обусловливается однозначным определением задачи)],

ОТВЕТ: В[ести данные]

(пользователем выбран режим ввода данных; для ответа достаточно одного символа),

ОТВЕТ: ИЗ [менить данные]

(выбран режим изменения данных; для ответа требуется два символа - "ИЗ", т.к. есть еще одна задача, начинающаяся с буквы "И" - исключить данные).

Если требуется принять или отвергнуть единственный вариант, то используется второй вариант диалога: вопросы, требующие ответа: да/нет.

П р и м е р.

ДЕЙСТВИЕ: Повторить вычисления при новых данных?

ОТВЕТ: Д [а].

Запрос с указанием синтаксиса ответа. Этот тип диалога по сути дела включает в себя несколько функций простого запроса. На запрос системы пользователь должен реагировать синтаксически ограниченным входным сообщением.

П р и м е р.

ДЕЙСТВИЕ: Какая дата Вас интересует? (мм/дд/гг)

ОТВЕТ: 12/02/86

Заполнение форм (бланков). При этом типе диалога за одно действие может быть задано несколько вопросов. Выходное сообщение системы (действие) состоит из форм на естественном языке, представленных на экране дисплея. Пользователь должен заполнить соответствующие поля конкретными значениями (именами, числами, датами и т.д.).

Когда пользователь заканчивает ввод одного элемента, система автоматически перемещает курсор к началу следующего поля.

П р и м е р:

Анкетные данные студента

фамилия имя отчество

xxxxxxxxxxxxxxx xxxxxxxxxxxxxx xxxxxxxxxxxxxx

дата рожд. место рождения

xx.xx.xx xxxxxxxxxxxxx

домашний адрес: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

окончил xxxxxxxxxxxxxxxxxxxxxxxxxxxx в xxxx году

поступил на первый курс

факультета xxxxxxxxxxxxxxxxxxxxxxxxxxx в xxxx году

Здесь позиции на экране, отмеченные символами "x", являются полями, которые должен заполнить пользователь. Как правило, для использования форм в диалоговых системах применяется специальное программное обеспечение.

Запрос свободного ответа. Работа с этим типом диалога ведется на так называемом квазиестественном языке, т.е. языке, близком к терминам предметной области и реализованном в данной системе. Пользователю предоставляется неограниченный выбор среди всех функций, доступных в текущем состоянии диалога.

Примером такого типа диалога может служить выбор команд языка управления задания при работе с операционной системой ОС РВ на семействе СМ ЭВМ.

ДЕЙСТВИЕ: > (этот символ означает, что система готова принять команду)

ОТВЕТ: *FOR ALFA, BETA = GAMMA*

(пользователем транслируется модуль *GAMMA.FTN*, в результате чего формируются файл объектного модуля *ALFA.OBJ* и листинг-файл *BETA.LST* ; расширения файлов *.FTN* ,*.OBJ* и *.LST* выбираются по умолчанию).

2. ОРГАНИЗАЦИЯ ДИАЛОГА С ИСПОЛЬЗОВАНИЕМ ФАЙЛА СИНТАКСИЧЕСКОГО ДЕРЕВА ДИАЛОГА

Рассмотрим применение специального программного обеспечения, которое на СМ ЭВМ на алгоритмическом языке Фортран позволяет организовать диалог пользователя с ЭВМ по типу предложения для выбора.

Предварительно весь диалог пользователя с ЭВМ описывается с помощью дерева диалога. В узлах дерева указываются действия системы, а ветви описывают ответы пользователя. Весь диалог целесообразно разбить на этапы или уровни. Как правило, с увеличением уровня увеличивается степень детализации и конкретизации решаемой задачи. Ответ пользователя представляется в виде командных строк диалога, которые состоят из последовательных наборов слов двух типов: ключевых и переменных. Ключевые слова – это слова, которые во всех командных строках имеют один и тот же вид. Переменные слова могут принимать различные значения в одинаковых по назначению командных строках и характеризуются только их допустимой длиной и местом появления в командной строке. Первым словом командной строки обязательно должно быть ключевое слово. В командной строке не может быть двух переменных слов рядом.

Совокупность ключевых слов, которые в дереве диалога описываются ветвями, исходящими из одного и того же узла, называется группой ключевых слов.

Любое ключевое слово этой группы может быть выбрано пользователем при вводе очередного слова командной строки.

Например, задача, обрабатывающая такие структуры данных, как фонд, подсхема, файл, может иметь следующие группы ключевых слов:

1. Создать, удалить, изменить, открыть;
2. Фонд, подсхему, файл;
3. Без подтверждения, с условием.

В соответствии с некоторым синтаксическим деревом диалога могут быть, например, использованы следующие командные строки:

создать фонд < имя фонда > ;

удалить файл < имя файла > без подтверждения и т.д.

(Здесь символом < ... > обозначены переменные слова).

2.I. Ввод командных строк с терминала и управление диалогом

Ключевые слова командных строк вводятся с терминала в режиме довывода символов. Это означает, что для задания ключевого слова требуется указать столько символов, сколько необходимо для однозначного распознавания этого слова в текущей группе. Переменные слова режима довывода не имеют и должны вводиться полностью.

Для управления диалогом используются функциональная клавиатура дисплея и специальные символы (таблица).

№№ п/п	Используемая клавиатура дисплея		Пояснения
	СМ7209	I5И3-00I3	
I	*CR*	ВК	Признак окончания ввода
2	*CTRL/Z*	СУ/*Z*	Аварийное прерывание ввода командной строки
3	*CTRL/U*	СУ/*U*	Аварийное прерывание ввода командной строки
4	*DEL*	ЗБ	Посимвольное удаление набранных слов или их частей
5	*LF*	ПС	Удаление всего слова или набранной части слова командной строки
6	?	?	При нажатии клавиши "?" на экране терминала печатается справочная информация для слова, ввод которого ожидается
7	Пробел	Пробел	Является ограничителем переменного слова командной строки

2.2. Формирование файла синтаксического дерева диалога

Файл синтаксического дерева диалога представляет собой текстовый файл на магнитном диске со строками фиксированной длины (78 символов), который может быть подготовлен любым текстовым редактором (например *TED*). В файл синтаксического дерева диалога могут входить:

группы ключевых слов командных строк;

описание переменных слов (указание максимально допустимых длин);

справочная информация о группах ключевых слов и о переменных словах;

служебная информация: ссылки ключевых слов на следующие группы, т.е. описание путей прохождения по узлам синтаксического дерева.

Группы ключевых слов. Группа ключевых слов представляет собой набор строк файла синтаксического дерева, содержащих ключевые слова, номера связанных с ними переменных слов и служебную информацию для получения справок и перехода к следующим группам ключевых слов.

Набор строк группы может располагаться с любой строки файла. Ограничителем группы является строка минусов ('—...').

Каждая строка группы ключевых слов содержит следующую информацию:

текст ключевого слова;

номер переменного слова, следующего в командной строке за указанным ключевым;

номер строки *HELP* (справки) для переменного слова;

номер строки *HELP* для следующей группы ключевых слов;

номер строки файла, с которой начинается следующая группа ключевых слов (указатель на следующий узел).

Текст ключевого слова задается с 14 колонки строки и заканчивается символом "." (точка). Для отделения ключевых слов друг от друга на экране дисплея в тексте ключевого слова должен содержаться символ-разделитель (для этого удобно использовать пробел). Максимальная длина ключевого слова – 52 символа, если за ним не следует переменное слово, и 42 символа – в противном случае.

Номер переменного слова задается, если за указанным в строке группы ключевым словом должно следовать переменное слово. Номер переменного слова указывается с 57 колонки строки и заканчивается нецифровым символом. Номер строки *HELP* для переменного слова указывается с 62 колонки строки, а номер строки *HELP* с информацией о следующей группе ключевых слов – с 68 колонки.

Номер строки файла, с которой начинается следующая группа ключевых слов, задается с 74 колонки. Для указания того, что описываемое ключевое слово является последним в командной строке, в 74 колонке необходимо поставить "0" или пробел.

Если некоторое ключевое слово может быть как последним в командной строке, так и нет, то тогда группа ключевых слов, на которую это слово ссылается, должна содержать стандартное ключевое слово"<НК>".

Описание переменных слов. Описание переменных слов начинается после служебной строки, состоящей из символов " $ " и располагаемой не ниже 10 строки файла синтаксического дерева.

Описание переменного слова включает:

максимальную длину слова, указываемую с 9-й колонки строки;

текст комментария, располагаемого после указателя длины и разделителя (пробел, символ, знак и т.д.).

Порядок следования строк описания переменных слов соответствует их порядковым номерам. Описание переменных слов заканчивается строкой минусов ("-").

Задание строк справочной информации. Строки *HELP* могут размещаться в произвольном месте файла синтаксического дерева. Текст справки, выводимой на терминал по одной команде "?", может занимать несколько последовательных строк файла и в каждой строке должен начинаться с 8 колонки.

Ограничителем строк является строка минусов.

Примечание: Первые семь колонок строк файла могут использоваться по усмотрению пользователя. Здесь обычно указывают порядковые номера строк, порядковые номера ключевых и переменных слов и т.д.

2.3. Обращение к программе

Программа, обеспечивающая диалог с использованием файла синтаксического дерева, носит название *GUIDE*.

Обращение к ней на Фортране имеет вид

CALL GUIDE(LUNF, NAMFIL, RESP, KEYW1, IHELP1, ITERM, IER),

где *LUNF* - логический номер для файла синтаксического дерева (целая константа);

NAMFIL - полная спецификация имени файла синтаксического дерева (литеральная константа);

RESP - символьный отзыв пользовательской программы, на который будут вводиться командные строки (литеральная константа);

KEYW1 - номер строки в файле синтаксического дерева, с которой начинается первая группа ключевых слов (целая константа или переменная);

IHELP1 - номер строки в файле синтаксического дерева, с которой начинается справка по первой группе ключевых слов;

ITERM – возвращаемый код клавиши завершения командной строки (целая переменная);
0 – клавиша <*CR*> , 1 – *CTRL/U* ,
2 – *CTRL/Z*;

IER – код завершения программы *GUIDE* (целая переменная):
= 0 – нормальное завершение,
> 0 – ошибка директивы ввода-вывода.

Для передачи вызывающей программе результатов работы программы *GUIDE* используются общие области

COMMON/CSTRIN/STR ,

где *STR* – байтовый массив, размером 80, содержащий полную командную строку, введенную с терминала;

COMMON/REF/NUM, NUMWDS, LENVFL ,

где *NUM* – количество введенных слов командной строки (ключевых и переменных);

NUMWDS – байтовый массив, размерностью 10, в который помещается список введенных слов; для ключевых слов передаются их номера в тех группах ключевых слов, где они описаны; для переменных слов – их номера со знаком минус;

LENVFL – байтовый массив размерностью 4, в котором передаются максимальные размеры переменных слов.

COMMON/CMLCOM/CMLIN,

где *CMLIN* – байтовый массив, размерностью 78, в котором передаются значения переменных слов, введенных в командной строке; значения переменных слов передаются в позициях массива, соответствующим номерам слов и их максимально возможным длинам и заканчиваются кодом 0, если длина введенного слова меньше максимальной.

2.4. Подключение программы *GUIDE* к программе пользователя

Подключение программы *GUIDE* к программе пользователя производится на этапе построения образа задачи через библиотеку *BSPLIB*, расположенную на устройстве *DK* : в разделе [1,1].

П р и м е р.

```
>TKB  FTSK=FOBJ, DK:[1,1]BSPLIB/LB ,
```

где *FOBJ* - имя объектного модуля программы пользователя;

FTSK - имя файла образа задачи.

Пример использования программы *GUIDE* приведен в приложении.

ПОРЯДОК ПРОВЕДЕНИЯ РАБОТЫ

До начала аудиторного занятия, во время самостоятельной работы, студенты должны изучить теоретические материалы данной лабораторной работы. Работа выполняется бригадами студентов по 3-4 человека в два этапа, по следующей примерной схеме.

Первый этап (длительность аудиторного занятия - 4 ч):

выдача индивидуальных заданий;

составление файла синтаксического дерева диалога;

составление диалоговой программы, использующей файл синтаксического дерева диалога;

проверка готовности студентов к выполнению следующего этапа.

Второй этап. (длительность аудиторного занятия - 4 ч):

подготовка на ЭВМ файла синтаксического дерева диалога;

подготовка диалоговой программы;

отладка диалоговой программы и запуск ее на выполнение на ЭВМ;

демонстрация работы разработанной диалоговой программы преподавателю;

оформление отчета.

ОФОРМЛЕНИЕ ОТЧЕТА

В отчете необходимо привести:

вариант задания;
разработанную схему диалога;
распечатку файла синтаксического дерева диалога;
распечатку диалоговой программы.

Контрольные вопросы

1. Что называется диалогом пользователя с ЭВМ?
2. Из каких информационных центров состоит диалоговая система?
3. Какие требования предъявляются к диалогу?
4. Какие типы диалога вы знаете?
5. Как реализуется диалог типа предложение для выбора?
6. Что такое синтаксическое дерево диалога?
7. Какими программными средствами может быть подготовлен файл синтаксического дерева диалога?

Библиографический список

Деннинг В., Эссиг Г., Маас С. Диалоговые системы "человек-ЭВМ". Адаптация к требованиям пользователя. - М.: Мир, 1984.

Диалог пользователя и ЭВМ. Основы проектирования и реализация /Довгялло А.Н. - Киев: Наукова думка, 1981.

Мартин Дж. Системный анализ передачи данных. - М.: Мир, 1975.

МЕТОДЫ СОРТИРОВКИ И ПОИСКА ИНФОРМАЦИИ

Цель работы: закрепление и углубление теоретических знаний по методам поиска и сортировки; получение практических навыков сортировки таблиц и использования поисковых систем.

I. ПОИСК ИНФОРМАЦИИ

В системах автоматизированного проектирования часто приходится иметь дело с обработкой большого объема информации. Эту информацию удобно представлять в виде таблиц (или файлов), состоящих из однородных записей, описывающих некоторую совокупность элементов данных.

Предположим, что при работе с таблицами необходимо среди большого количества записей отыскать нужную. Как это сделать? Для этого необходимо выяснить, по какому признаку мы собираемся проводить поиск. В каждой записи выделяется элемент, называемый ключом, который сравнивается со специальным аргументом поиска x. Очевидно, что существуют две возможности окончания поиска: либо поиск оказывается удачным и запись отыскивается, либо поиск оказывается неудачным, т.е. в таблице нет записи, содержащей ключ, совпадающий с заданным аргументом.

Рассмотрим возможные алгоритмы поиска информации в таблице.

I.I. Линейный поиск

Этот алгоритм обычно используется для поиска в таблицах, в которых записи не упорядочены по ключам. Например, в телефонном справочнике мы хотим найти абонента, которому принадлежит телефон с интересующим нас номером (известно, что такие справочники упорядочены по фамилиям абонентов, а не по номерам телефонов). В этом случае единственный путь для поиска нужной записи состоит в последовательном сравнении соответствующего ключа каждой записи с заданным аргументом поиска.

При линейном поиске, согласно теории вероятности, в среднем необходимо просмотреть половину таблицы для нахождения нужной записи. При этом среднее время поиска оценивается соотношением:

$$T_{ср} = [\text{время одного просмотра}] \times \frac{N}{2},$$

где N - число записей в таблице.

I.2. Поиск в упорядоченной таблице

Изменим условия в приведенном выше примере и поставим задачу определения по телефонному справочнику номера телефона интересующего нас абонента.

Как мы обычно поступаем в этом случае? Мы делаем приблизительную оценку расположения нашей записи в справочнике и открываем справочник на предполагаемой странице. Если нужной записи нет, переворачиваем несколько страниц вперед или назад и проверяем снова. Нам известно, как идти к цели, так как мы знаем о важном свойстве телефонного справочника, а именно о том, что он упорядочен. Такая упорядоченность называется лексиграфическим порядком.

Существует большое число алгоритмов поиска записи в упорядоченной таблице. Упорядоченной будем называть таблицу, для ключей которой выполняется соотношение

$$K_1 < K_2 < \ldots < K_N,$$

где N - количество записей в таблице;

K_i - значение ключа i -й записи.

Мы же рассмотрим только один, так называемый бинарный поиск. Иногда можно встретить и другие названия этого алгоритма поиска: двоичный поиск, метод деления пополам, логарифмический поиск.

Суть алгоритма бинарного поиска заключается в следующем. Поиск начинают с середины таблицы, сравнивая аргумент поиска K с ключом $\left(\frac{N+1}{2}\right)$ -й записи.

Сравниваемый ключ K_i может быть равен, больше или меньше аргумента поиска [для лексиграфического порядка при сравнении символьных данных используются числовые эквиваленты (коды каждого символа)]. Результат сравнения позволяет определить, в какой половине таблицы необходимо продолжить поиск. Если $K_i = K$, то искомая запись найдена и поиск прекращается. Если $K_i > K$, то выбирается верхняя половина

таблицы в качестве новой таблицы для поиска. Если же $K_i < K$, то в дальнейшем используется нижняя половина таблицы. Проанализированная запись при этом в новую таблицу не включается.

При бинарном поиске требуется сделать в среднем $log_2(N)$ проверок.

Приведем пример использования бинарного поиска. В этом примере в таблице, состоящей из 15 записей, требуется найти запись, у которой значение ключа равно "ЗАЙЦЕВ".

Искомая запись находится за три просмотра.

Пример работы алгоритма бинарного поиска

Номер записи	Ключ	1-й просмотр	2-й просмотр	3-й просмотр
1	Абрамов			
2	Бородин			
3	Бурмистров			
4	Давыдов		< Зайцев	
5	Жуков			
6	Зайцев			= Зайцев
7	Иванов			
8	Колесов	> Зайцев		
9	Леонтьев			
10	Медведев			
11	Николаев			
12	Петров			
13	Решетов			
14	Степанов			
15	Яковлев			

1.3. Выборка по вторичным ключам

В приведенном выше материале мы рассмотрели так называемый поиск по первичным ключам, т.е. ключам, однозначно определяющим запись в файле. Но в практических задачах часто требуется вести поиск не по первичным ключам, а по значениям других полей записи, которые мы будем называть вторичными ключами.

Например, в файле статистических данных студентов института может потребоваться найти всех студентов третьего курса, проживающих в общежитии, не моложе 20 лет и т.д.

Спецификацию искомых записей обычно называют запросом. Принято выделять следующие три типа запросов.

Простой запрос. При этом типе запроса указывается конкретное значение аргумента поиска, а сам поиск проводится по единственному ключу. (По сути дела, это будет поиск по первичному ключу).

Пример запроса:

ИМЯ = АЛЕКСАНДР

Запрос по области значений. Для ключа задается конкретная область значений, например:

Рост < 180 или $17 \leqslant$ возраст $\leqslant 21$

Булев запрос. Этот тип запроса состоит из двух первых типов, соединенных логическими операциями

И, ИЛИ, НЕТ

Пример запроса:

(КУРС = 3) И (ИЗУЧАЕМЫЙ ЯЗЫК = АНГЛИЙСКИЙ)
И НЕ((СПЕЦИАЛИЗАЦИЯ = ПРОЧНОСТЬ) ИЛИ
(СПЕЦИАЛИЗАЦИЯ = ДИНАМИКА ПОЛЕТА))

Наиболее простой способ выборки по вторичным ключам заключается в следующем: вначале накапливаются все запросы и только после этого начинается их обработка последовательным поиском во всем файле. Этим способом можно пользоваться, когда просматриваемый файл не слишком большой и когда не требуется быстрого ответа на введенный запрос.

Если же требования к поиску таковы, что приведенным способом пользоваться нельзя (например, при поиске в файле, представляющим собой объект непрерывных запросов), то необходимо перейти к другим, более сложным, схемам поиска: с использованием инвертированных файлов, комбинаторного хэширования и т.д. (см.работу [I]).

2. СОРТИРОВКА

Очевидным является то, что поиск целесообразнее проводить в упорядоченных таблицах. Для получения таких упорядоченных таблиц используются методы сортировки.

Задача сортировки заключается в следующем. Имеется таблица из N записей R_1, R_2, ..., R_N. Каждая запись R_i имеет ключ K_i, который управляет процессом сортировки. Помимо ключа, запись может содержать дополнительную сопутствующую информацию, которая не влияет на сортировку, но всегда остается в этой записи. (Например, при сортировке списка абонентов телефонной сети ключом является фамилия абонента, а сопутствующей информацией будет адрес абонента и номер его телефона).

Требуется найти такую перестановку записей $P(1), P(2), \ldots, P(N)$, после которой ключи расположатся в неубывающем порядке:

$K_{P(1)} \leqslant K_{P(2)} \leqslant \ldots \leqslant K_{P(N)}$.

2.1. Методы внутренней сортировки

Внутренними называются сортировки, которые применяются к таблицам, полностью размещаемым в оперативной памяти ЭВМ. В зависимости от цели сортировки в одних случаях может понадобиться физически переразмещать записи в памяти так, чтобы их ключи были упорядочены, в других можно обойтись вспомогательной таблицей, которая определяет перестановки. Например, если ключи короткие, а сопутствующая информация велика, то для повышения скорости сортировки ключи можно вынести в специальную таблицу адресов (или ссылок), которые указывают на записи.

Другая схема сортировки записей, имеющих большую сопутствующую информацию, заключается в том, что в каждой записи выделяется вспомогательное поле связи, в котором указывается номер следующей по порядку записи из исходной таблицы. Эта схема сортировок называется сортировкой списков.

После сортировки таблицы адресов или сортировки списка записи могут быть легко обработаны и, если нужно, расположены в требуемом порядке.

Для понимания механизма работы алгоритмов сортировок отвлечемся от необходимости сортировать большие по объему записи и рассмотрим только сортировку ключей. Все методы внутренних сортировок могут быть разделены на три основных типа: сравнительные, распределительные и сортировки вычислением адреса.

2.I.I. Сравнительные сортировки

Сравнительные сортировки основаны на сравнении пары ключей на каждом этапе. Рассмотрим несколько широко известных сравнительных сортировок.

О б м е н н а я с о р т и р о в к а. Алгоритм обменной сортировки основан на сравнении пары соседних ключей и перестановки их в требуемом порядке. В зависимости от того, в каком направлении происходит сравнение пар ключей, обменную сортировку называют также пузырьковой сортировкой (меньшие по значению ключи подобно пузырькам поднимаются вверх) или сортировкой погружением (большие по значению ключи подобно тяжелым шарикам опускаются вниз).

Алгоритм обменной сортировки (сортировки погружением) рассмотрим на примере упорядочивания I2 чисел. Для большей наглядности файл сортируемых чисел представим в вертикальном виде. Результаты каждого очередного просмотра (или прохода по алгоритму) также будем записывать в виде столбцов таблицы.

Пример обменной сортировки

Неотсортированные числа	I-й проход	2-й проход	3-й проход	4-й проход	5-й проход	6-й проход	7-й проход
I9	I3	05	05	0I	0I	0I	0I
I3	05	I3	0I	05	05	05	02
05	I9	0I	I3	I3	I3	02	05
27	0I	I9	I9	I6	02	09	09
0I	26	26	I6	02	09	II	II
26	27	I6	02	09	II	I3	I3
3I	I6	02	09	II	I6	I6	I6
I6	02	09	II	I9	I9	I9	I9
02	09	II	2I	2I	2I	2I	2I
09	II	2I	26	26	26	26	26
II	2I	27	27	27	27	27	27
2I	3I	3I	3I	3I	3I	3I	3I

Черточками ограничены ключи, которые встали на свои места.

Рассмотренная таблица сортируется за семь просмотров. При этом нетрудно заметить, что в каждом просмотре после последней операции сравнения, по крайней мере, один ключ встает на свое окончательное

место и его можно исключить из сравнений в последующих проходах. Следовательно, эффективность сортировки может быть повышена за счет сокращения на каждом просмотре размерности сортируемой таблицы и за счет проверки на досрочное завершение работы алгоритма (если во время прохода по алгоритму не было перестановок – сортировка закончена).

Такая оптимизированная сортировка по грубой оценке требует $N \times (N-1)/2$ сравнений. Необходимо отметить, что обменная сортировка кроме простоты реализации и наглядности не имеет никаких других преимуществ, поэтому в практических задачах используются другие виды сортировки.

Ш е й к е р – с о р т и р о в к а. В обменной сортировке за один просмотр ключ не может переместиться более чем на одну позицию вверх. Так, если наименьший ключ в исходной таблице находился внизу, то сортировка выполняется при максимальном числе сравнений. Это наводит на мысль о шейкер–сортировке, в которой таблица просматривается попеременно в обоих направлениях. При таком подходе среднее число сравнений несколько сокращается, но тем не менее, так же, как и обменная сортировка, шейкер–сортировка не дает существенного повышения эффективности.

С о р т и р о в к а Ш е л л а. Основной недостаток рассмотренных выше сортировок заключается в том, что в них меняться местами могут лишь соседние элементы, т.е. каждая запись может переместиться только на одну позицию. Алгоритм сортировки, в которой происходит сравнение пар, расположенных на некотором расстоянии друг от друга, был предложен Д.Шеллом и носит его имя. Сортировка Шелла близка к оптимальной для сравнительных сортировок. Ее идея заключается в следующем.

Сортировка начинается со сравнения ключей, находящихся на расстоянии d друг от друга. Это приводит к тому, что ключи, которые находятся не на месте, будут перемещаться быстрее, чем при простой обменной сортировке.

При каждом очередном просмотре значение d уменьшается:

$$d_{i+1} = 0{,}5(d_i + 1).$$

В сортировке Шелла на первом этапе файл делится на $N/2$ групп по два ключа в каждой, на втором этапе файл делится на $N/4$ групп по четыре ключа в каждой и т.д. На каждом этапе в каждой группе ключей проводятся сравнения с соответствующими перестановками. После того как

сравнения перестают приводить к перестановкам, при заданном значении d процесс возобновляется с новым значением d .

При сортировках в группах участвуют либо сравнительно короткие таблицы, либо сравнительно хорошо упорядоченные таблицы.

Пример сортировки Шелла

Исходная таблица	I-й просмотр (d_1 = 6)	2-й просмотр (d_2 = 3)	3-й просмотр (d_3 = 2)	4-й просмотр (d_4 = I)
I9	— I9	— * 09	— * 02	* 0I
I3	I3	* 0I	0I	* 02
05	* 02	02	— * 09	* 05
27	09	— * I9	* 05	* 09
0I	0I	** II	— II	II
26	* 2I	* 05	** I3	I3
3I	— 3I	— * 27	— ** I6	I6
I6	I6	** I3	* I9	I9
02	* 05	* 2I	— *** 2I	2I
09	* 27	— * 3I	* 26	26
II	II	* I6	— * 27	27
2I	* 26	26	** 3I	3I

Для каждого просмотра черточками отмечены элементы первой группы

* – обмен
** – двойной обмен
*** – тройной обмен.

Эмпирические исследования показали, что сортировка Шелла требует приблизительно $BN(log_2(N))^2$ единиц времени для N – элементного вектора, где B – коэффициент пропорциональности.

2.I.2. Распределительная сортировка

В распределительной сортировке упорядочивание ключей происходит путем анализа числового значения ключа по одной цифре за этап.

Рассмотрим одну из наиболее простых схем распределительных сортировок – сортировку поразрядным группированием. Все сортируемые ключи распределяются по группам, число которых равно основанию используемой системы счисления. Для десятичной системы счисления таких групп будет десять: 0, I, 2,...,9.

Сортировка начинается с анализа самых младших разрядов ключей. Все ключи, имеющие в младшем разряде ноль, помещаются в нулевую группу, имеющую в младшем разряде единицу, - в первую группу и т.д. После того как все элементы будут распределены по такому правилу, содержимое групп объединяется в новый файл, в котором вначале будут располагаться ключи нулевой группы, затем первой группы и т.д. Процесс повторяется для следующего разряда и так далее, до тех пор, пока не будут проанализированы все разряды.

Сортировка поразрядным группированием обладает высоким быстродействием, но имеет такой существенный недостаток, как большой дополнительный объем памяти, требуемый для выделяемых групп.

Пример сортировки поразрядным группированием

Исходная таблица	I-е распределение	I-е объединение	2-е распределение	Окончательное объединение
I9	0)	0I	0) 0I,02,05,09	0I
I3	I) 0I,3I,II,2I	3I	I) II,I3,I6,I9	02
05	2) 02	II	2) 2I,26,27	05
27	3) I3	2I	3) 3I	09
0I	4)	02	4)	II
26	5) 05	I3	5)	I3
3I	6) 26,I6	05	6)	I6
I6	7) 27	26	7)	I9
02	8)	I6	8)	2I
09	9) I9,09	27	9)	26
II		I9		27
2I		09		3I

2.I.3. Сортировка вычислением адреса

Сортировка вычислением адреса преобразует ключ в адрес, близкий к тому месту, где ожидается его окончательное расположение. Рассмотрим один из наиболее простых алгоритмов этого вида сортировок.

Предположим, что значения ключей распределены довольно равномерно в диапазоне $[K_{min}, K_{max}]$. Начальные приближения адресов ключей определяются из соотношений

$$A = int(K_i/m), \quad i = \overline{1,N},$$

где оператор int обозначает выделение целой части числа, N - размерность таблицы,

$$m = int(K_{max}/N) + 1.$$

Начиная с первого, ключи последовательно располагаются по вычисленным адресам (таким образом, сортировка будет состоять из N - этапов).

Нетрудно убедиться в том, что в некоторых случаях ключи могут иметь одинаковые адреса. Поэтому до помещения ключа по вычисленному адресу необходимо проверить, не занято ли уже это место. Если это место занято, то по данному адресу располагается тот ключ, значение которого меньше. Ключу, у которого значение оказалось больше, присваивается следующий по порядку адрес, и уже для него повторяется процедура анализа на возможность размещения по новому адресу и т.д.Если "конфликта из-за места" нет, ключ перемещается по вычисленному адресу и начинается новый этап сортировки.

Приведенный алгоритм будет работать эффективнее, если для сортировки использовать таблицы большей(приблизительно в два раза), чем это требуется для размещения данных,размерности. Время сортировки оценивается величиной CN , где C - коэффициент пропорциональности.

Пример сортировки вычислением адреса

Номер этапа	I	2	3	4	5	6	7	8	9	IO	II	I2
Ключи	I9	I3	05	27	0I	26	3I	I6	02	09	II	2I
Вычисленный адрес	6	4	I	9	0	8	IO	5	0	3	3	7
Таблица												
0					0I	0I	0I	0I	*0I	0I	0I	0I
I			05	05	05	05	05	05	02	02	02	02
2									05	*05	05	05
3										09	*09	09
4		I3	I3	I3	I3	I3	I3	I3	I3	I3	II	II
5										I6	I3	I3
6	I9	I9	I9	I9	I9	I9	I9	I9	I9	I9	I6	I6
7											I9	*I9
8						26	26	26	26	26	26	2I
9				27	27	27	27	27	27	27	27	26
IO							3I	3I	3I	3I	3I	27
II												3I

Здесь звездочка "*" указывает на конфликт между ключами, а стрелка - на необходимость и направление перемещения ключей.

2.2. Методы внешней сортировки

При сортировке часто возникают ситуации, когда объем оперативной памяти ЭВМ не позволяет разместить в нем все записи файла. Предположим, например, что предназначенный для сортировки файл состоит из 5000 записей $R_1, R_2, \ldots, R_{5000}$, а в оперативной памяти ЭВМ можно разместить одновременно только 1000 этих записей.

Как провести сортировку в этом случае?

Один из наиболее популярных в настоящее время подходов к решению этой задачи состоит в следующем:

весь файл разбивается на подфайлы, целиком умещающиеся в памяти ЭВМ (для нашего примера $\{R_1, R_2, \ldots, R_{1000}\}$, $\{R_{1001}, R_{1002}, \ldots, R_{2000}\}, \ldots, \{R_{4001}, R_{4002}, \ldots, R_{5000}\}$;

каждый из подфайлов упорядочивается одним из методов внутренней сортировки;

полученные подфайлы объединяются или, как это принято называть, сливаются в один упорядоченный файл, расположенный на внешнем носителе информации.

2.3. Сортировка слиянием

Слияние означает объединение двух и более упорядоченных файлов в один упорядоченный файл.

Рассмотрим метод сортировки на слиянии упорядоченных файлов $x: \{x_1, x_2, \ldots, x_n / x_i \leqslant x_{i+1},\ i = \overline{1, (n-1)}\}$ и $y: \{y_1, y_2, \ldots, y_m / y_i \leqslant y_{i+1},\ i = \overline{1, (m-1)}\}$ в один файл $z: \{z_1, z_2, \ldots, z_{m+n} / z_i \leqslant z_{i+1},\ i = \overline{1, (m+n-1)}\}$.

Наиболее простой способ слияния - сравнить два наименьших элемента каждого из сливаемых файлов, вывести наименьший из них в результирующий файл z и повторить процедуру.

Проиллюстрируем этот алгоритм следующим примером слияния двух упорядоченных файлов:

$$x : \{503,\ 703,\ 765\}$$
$$y : \{087\ \ 512,\ 677\}$$

Выделяя наименьший элемент из первой пары, получим

$$087 \left\{ \begin{array}{l} 503,\ 703,\ 765 \\ 512,\ 677 \end{array} \right.$$

Затем из сравнения новой первой пары определим второй элемент файла z :

$$087,\ 503 \left\{ \begin{array}{l} 703,\ 765 \\ 512,\ 677 \end{array} \right.$$

Повторяя выделения наименьшего элемента из первых пар, в конечном виде получим следующий файл:

$$z : \{\ 087,\ 503,\ 512,\ 677,\ 703,\ 765\ \}$$

3. ИНСТРУКЦИЯ ПО РАБОТЕ НА ЭВМ

На ЭВМ студенты работают с использованием пакета сменных магнитных дисков с именем *STUD* , расположенного на устройстве *DK1* : После регистрации в системе необходимо смонтировать используемое устройство командой:

```
> MOU DK1 : STUD
```

Работа на ЭВМ состоит из двух этапов. На первом этапе требуется сформировать файл статистических данных, для чего используется программа *STATS* из раздела [130, 100].

Обращение к программе:

```
> RUN [130,100] STATS
```

Программа *STATS* обеспечивает ввод статистических данных в интерактивном режиме, при котором ответы вводятся в ЭВМ после соответствующих подсказок и комментариев, не требующих дополнительной расшифровки в описании работы.

На втором этапе в сформированном файле статистических данных требуется произвести поиск информации, соответствующей выданному индивидуальному заданию. Для этого используется готовое программное обеспечение, вызываемое командой

```
> RUN [130,100] SEARCH
```

Программа *SEARCH* обеспечивает обработку всех трех типов запросов, описанных в п. I.3. Особенности программы требуют, чтобы запросы третьего типа (булевы запросы) подразделялись на простые запросы и запросы по области значений, а последовательность их ввода соответствовала приоритету выполнения используемых логических операций. Правила ввода запросов каждого типа (синтаксис, последовательность и т.д.) выводятся при выполнении программы на экран дисплея и в дополнительных пояснениях не нуждаются.

По завершении операции поиска информации требуется произвести сортировку выделенных записей по заданному ключу. Эту процедуру также выполняет программа *SEARCH* .

В заключение требуется распечатать на АЦПУ упорядоченные записи.

4. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

I. Изучение теоретического материала работы во внеаудиторное время занятий.

2. Краткий рассказ преподавателя о порядке проведения аудиторного занятия, рассчитанного на 4 академических часа.

3. Выдача индивидуальных заданий.

4. Проведение сортировки заданной таблицы данных "вручную" методами, указанными в индивидуальном задании.

Следующие этапы работы выполняются в дисплейном классе

5. Заполнение файла статистических данных.

6. Поиск в файле статистических данных информации согласно индивидуальному заданию.

7. Сортировка выделенных записей с последующей их распечаткой на АЦПУ.

8. Оформление отчета по работе.

5. ОФОРМЛЕНИЕ ОТЧЕТА

Отчет оформляется на отдельных листах бумаги с указанием фамилии и группы студента, а также выданного варианта задания. Отчет состоит из иллюстрации этапов "ручной" сортировки заданных таблиц данных и распечатки записей, выбранных из файла статистических данных.

Контрольные вопросы

1. Как называется элемент, по которому ведется поиск?
2. В каких случаях целесообразно использовать линейный поиск?
3. Какие таблицы называются упорядоченными?
4. Что такое выборка по "вторичным ключам"?
5. Какие типы запросов вы знаете?
6. В чем заключается задача сортировки?
7. Какие методы сортировки называются внутренними?
8. На какие основные типы можно подразделить методы внутренней сортировки?
9. На чем основаны алгоритмы обменных сортировок?
10. Каким образом происходит упорядочивание ключей в распределительных сортировках?
11. Как работает алгоритм сортировки вычислением адреса?
12. Для каких файлов может быть выполнена сортировка слиянием?

Библиографический список

1. К н у т Д. Искусство программирования для ЭВМ. Сортировка и поиск. Т. 3. - М.: Мир, 1978.

ОПИСАНИЕ КРИВЫХ В ПАРАМЕТРИЧЕСКОМ ВИДЕ

Цель работы: закрепление и углубление теоретических знаний построения обводов; приобретение практических навыков описания кривых методом Безье.

I. ОБЩИЕ СВЕДЕНИЯ

Одно из важных мест в системе автоматизированного проектирования занимают подсистемы формирования технической поверхности объекта. Создание таких подсистем предусматривает построение математических моделей поверхностей, отвечающих некоторым наперед заданным требованиям различного характера. Частным случаем разработки математических моделей поверхностей является построение обводов объекта проектирования. Обвод - это кривая, составленная из дуг различных кривых, состыкованных между собой определенным образом (совпадение точек стыка, равенство первых и вторых производных в стыке кривых и т.п.). Существует большое количество методов построения обводов. Среди них рассмотрим методы интерполяции.

Задача интерполяции заключается в следующем: по координатам узловых точек $x_i, y_i, i=\overline{1,n}$ некоторой кривой требуется определить коэффициенты интерполирующей функции $y(x)=\sum_{i=1}^{n} c_i \varphi_i(x)$. Интерполирующими функциями могут быть: прямая, дуга окружности, кривые второго и более высокого порядка, степенные функции и т.д.

Ограничимся рассмотрением интерполирующих функций, заданных в параметрическом виде.

При движении точки M в трехмерном пространстве относительно прямоугольной системы координат $OXYZ$ (рис.I) ее текущее положение может быть определено радиус-вектором $\bar{r}=\bar{r}(u)$, где параметр u служит меткой точки или ее координатной. В частном случае, в качестве

параметра u может использоваться время: $\bar{r} = \bar{r}(t)$ или в координатной форме: $x = x(t)$, $y = y(t)$, $z = z(t)$.

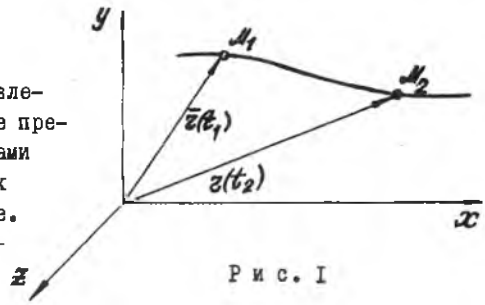


Рис. I

Параметрическое представление кривых имеет определенные преимущества перед другими формами записи кривых. Среди основных преимуществ отметим следующие.

I. При построении графического изображения кривой или поверхности упрощается определение координат промежуточных точек.

2. Упрощается воспроизведение кривых на графическом дисплее или графопостроителе.

3. Упрощаются различные преобразования исходной кривой (сдвиг, масштабирование, поворот и т.д.).

4. Широкая распространенность для описания гладких обводов в различных методах (кривые второго порядка, Фергюссона, Безье, Кунса и др.) и системах автоматизированного проектирования.

I.I. Параметрические кривые третьего порядка в форме Фергюссона

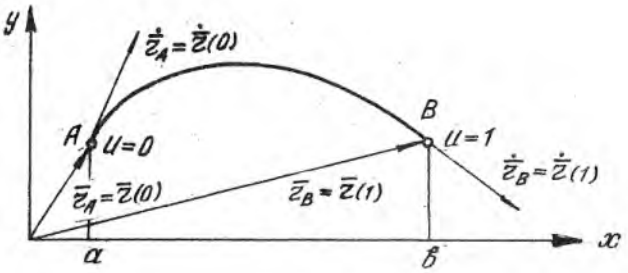
В общем случае уравнение кривой третьего порядка на отрезке $[a,b]$ имеет вид

$$\bar{r} = \bar{r}(u) = \bar{a}_0 + \bar{a}_1 u + \bar{a}_2 u^2 + \bar{a}_3 u^3 . \qquad (I)$$

Параметр u зададим таким образом, чтобы для точки A $u = 0$ и для точки B $u = I$. Форму кривой будем характеризовать координатами начальной A и конечной B точек кривой, а также значениями производных $\dot{\bar{r}}_A$ и $\dot{\bar{r}}_B$ в этих точках (рис.2).

Продифференцировав уравнение (I), будем иметь

$$\frac{d\bar{r}}{du} = \dot{\bar{r}} = \dot{\bar{r}}(u) = \bar{a}_1 + 2\bar{a}_2 u + 3\bar{a}_3 u^2 . \qquad (2)$$



Р и с. 2

Для определения векторов $\bar{a}_i$ ($i = \overline{0,3}$) запишем граничные условия. Подставляя в уравнения (I) и (2) граничные условия для точки A ($u = 0$), получим

$$\begin{aligned} &\bar{z}(0) = \bar{a}_0 \\ &\dot{\bar{z}}(0) = a_1 \end{aligned} \quad \text{или} \quad \begin{aligned} &\bar{a}_0 = \bar{z}(0), \\ &\bar{a}_1 = \dot{\bar{z}}(0). \end{aligned} \tag{3}$$

Аналогично для точки B ($u = 1$)

$$\begin{aligned} &\bar{z}(1) = \bar{a}_0 + \bar{a}_1 + \bar{a}_2 + \bar{a}_3, \\ &\dot{\bar{z}}(1) = \bar{a}_1 + 2\bar{a}_2 + 3\bar{a}_3. \end{aligned} \tag{4}$$

Решая систему уравнений (4) с учетом (3), находим

$$a_2 = 3[\bar{z}(1) - \bar{z}(0)] - 2\dot{\bar{z}}(0) - \dot{\bar{z}}(1),$$

$$a_3 = 2[\bar{z}(0) - \bar{z}(1)] + \dot{\bar{z}}(0) + \dot{\bar{z}}(1).$$

Подставляя найденные значения $\bar{a}_0$, $\bar{a}_1$, $\bar{a}_2$, $\bar{a}_3$ в уравнение(I) получим уравнение кривой в форме Фергюссона:

$$\begin{aligned} \bar{z}(u) = {} & \bar{z}(0)[1 - 3u^2 + 2u^3] + \dot{\bar{z}}(0)[u - 2u^2 + u^3] + \\ & + \bar{z}(1)[3u^2 - 2u^3] + \dot{\bar{z}}(1)[-u^2 + u^3]. \end{aligned} \tag{5}$$

Для машинной реализации это уравнение целесообразно записать в матричном виде:

$$\bar{z} = uCS,$$

где

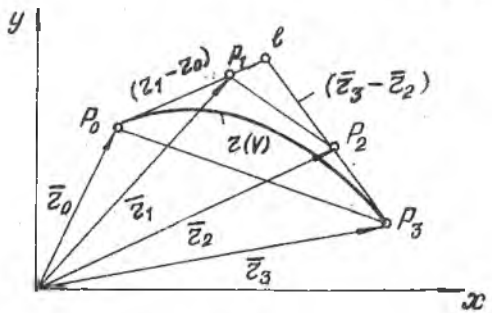
$$u = [1\ u\ u^2\ u^3],$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix}, \quad \bar{S} = \begin{bmatrix} \bar{z}(0) \\ \bar{z}(1) \\ \dot{\bar{z}}(0) \\ \dot{\bar{z}}(1) \end{bmatrix}.$$

I.2. Параметрические кривые в форме Безье

Для повышения наглядности представления кривой Безье перегруппировал члены параметрического уравнения Фюргюссона таким образом, чтобы можно было управлять формой кривой с помощью характеристической ломаной.

В этом случае форма кривой задается соприкасающейся в граничных точках ломаной, образованной прямыми P_0P_1, P_1P_2, P_2P_3 (рис.3).



Р и с. 3

Обозначим

$$\begin{aligned} \bar{z}(0) &= \bar{z}_0, \\ \bar{z}(1) &= \bar{z}_3, \\ \dot{\bar{z}}(0) &= 3(\bar{z}_1 - \bar{z}_0), \\ \dot{\bar{z}}(1) &= 3(\bar{z}_3 - \bar{z}_2). \end{aligned} \qquad (6)$$

Кривая в форме Безье, оставаясь четырехпараметрической, проходит через точки P_0 и P_3 и имеет касательные, идущие соответственно от точек P_0 к P_1 и от P_2 к P_3.

Подставим зависимости (6) в уравнение Фергюссона (5):

$$\bar{z} = \bar{z}(u) = \bar{z}_0[1-3u^2+2u^3] + \bar{z}_3[3u^2-2u^3] + \\ + 3(\bar{z}_1-\bar{z}_0)[u-2u^2+u^3] + 3(\bar{z}_3-\bar{z}_2)[-u^2+u^3]. \quad (7)$$

Упростив выражение (7), получим уравнение для записи кубической кривой Безье

$$\bar{z} = \bar{z}(u) = \bar{z}_0(1-u)^3 + 3u(1-u)^2\bar{z}_1 + 3u^2(1-u)\bar{z}_2 + \\ + u^3\bar{z}_3. \quad (8)$$

В матричной форме уравнение (8) имеет вид

$$\bar{z} = u\,C\,\bar{S}, \quad (9)$$

где

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}, \quad S = \begin{bmatrix} \bar{z}_0 \\ \bar{z}_1 \\ \bar{z}_2 \\ \bar{z}_3 \end{bmatrix}.$$

Для плоских кривых уравнение (9) в координатной форме запишется в следующем виде:

$$z = \begin{bmatrix} x_z \\ y_z \end{bmatrix},$$

$$x_z = u\,C\,S_x$$

$$y_z = u\,C\,S_y,$$

где

$$S_x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}, \quad S_y = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix},$$

а u и C имеют тот же смысл и значения, что и в уравнении (9).

Управление формой кривой осуществляется изменением длин отрезков P_0P_1 и P_2P_3 (или величинами $\bar{z}_0, \bar{z}_1, \bar{z}_2, \bar{z}_3$). При одновременном

увеличении отрезков увеличивается полнота кривой. Если увеличивать только один из отрезков, то кривая будет располагаться ближе к этому отрезку.

Для того, чтобы плоская кривая образовывала петлю, необходимо выполнение условий

$$P_0P_1 > P_0C \quad \text{и} \quad P_2P_3 > CP_3 .$$

Примеры формы плоской кривой, в зависимости от выбранной характеристической ломаной, приведены на рис. 4.

Безье было получено также уравнение полиноминальной кривой более высокого порядка:

$$\bar{r} = \bar{r}(u) = \sum_{i=0}^{n} \frac{n!}{(n-i)!\,i!}\, u^i (1-u)^{n-i} \bar{r}_i ,$$

где $\bar{r}_0$, $\bar{r}_1$, $\bar{r}_2$, ..., $\bar{r}_n$ – радиус-векторы n +I вершин $P_0, P_1, P_2, \ldots, P_n$ некоторой обобщенной характеристической ломаной.

Форма записи кубической кривой Безье (8) является частным случаем этой обобщенной записи.

Необходимо отметить, что с увеличением степени полиномов увеличивается непрерывность большого числа производных, т.е. увеличивается гладкость обводов. Но в то же время увеличение степени полинома снижает наглядность метода и усложняет процесс конструирования кривой требуемой формы.

2. ИНСТРУКЦИЯ ПО РАБОТЕ НА ЭВМ

В лабораторной работе реализован метод Безье, позволяющий описывать кривые полиномом n -й степени. Для этого задается некоторая характеристическая ломаная с точками P_0 , P_1 , P_2 ,..., P_n (где P_0 и P_n являются, соответственно, начальной и конечной точками кривой). Каждая точка задается координатами x и y на растровой матрице размером 512x256, т.е. значения x и y должны лежать в интервалах $1 \leqslant x \leqslant 512$, $1 \leqslant y \leqslant 256$. При этом предполагается, что начало системы координат располагается в левом нижнем углу экрана дисплея. Ось Ox направлена по горизонтали вправо, а

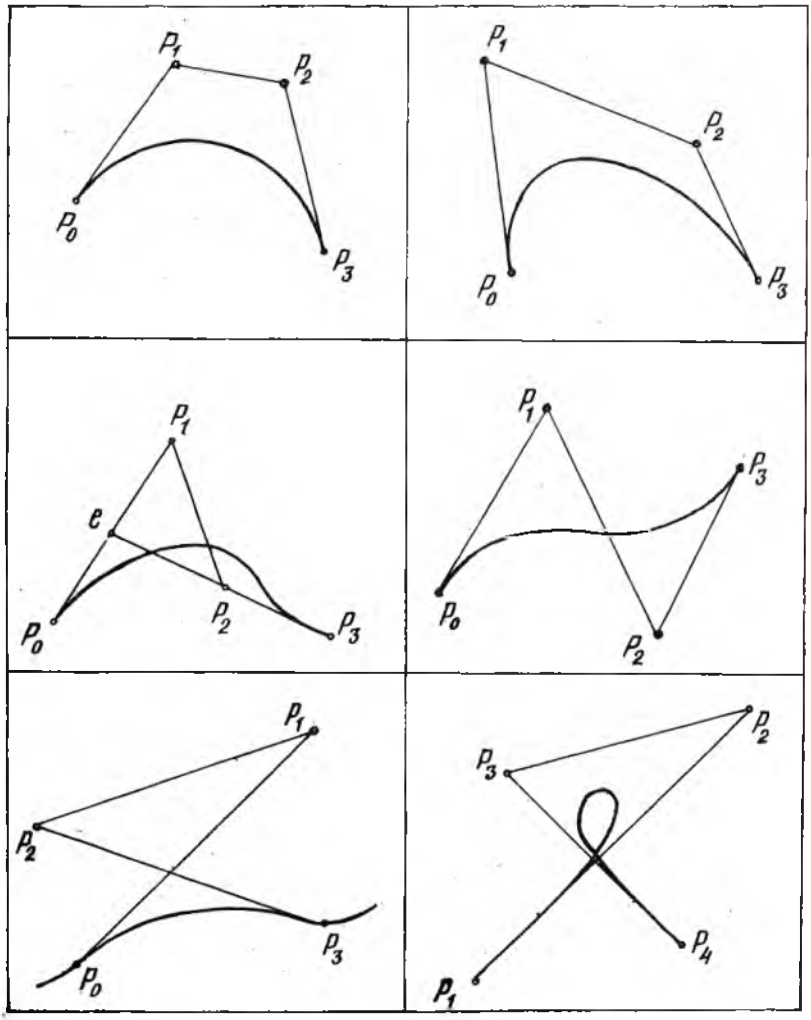


Рис. 4

ось OY – по вертикали вверх. Подбор точек $P_1, P_2, \ldots, P_{n-1}$ характеристической ломаной носит, как правило, итерационный характер.

Работа ведется в интерактивном режиме, при котором ответы пользователя вводятся в ЭВМ после соответствующих подсказок, не требующих дополнительной расшифровки в описании работы. Каждому студенту на экране дисплея предлагается вариант кривой, которую он должен повторить, подбирая координаты точек $P_1, P_2, \ldots, P_{n-1}$. Для каждого варианта данных на дисплее отображается исходная кривая, характеристическая ломаная, текущая интерполирующая кривая.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Изучить материалы лабораторной работы.
2. Получить у преподавателя вариант задания.
3. Ввести программу в ЭВМ.
4. С использованием ЭВМ подобрать параметры кривой, соответствующие выданному заданию.
5. Оформить отчет по работе.

4. ОФОРМЛЕНИЕ ОТЧЕТА

Отчет оформляется на отдельных листах бумаги с указанием фамилии и группы студента. Отчет включает в себя графическую иллюстрацию этапов конструирования кривой Безье с указанием значений координат точек характеристической ломаной. Для каждого этапа рисуются исходная кривая, характеристическая ломаная, текущая интерполирующая кривая Безье.

Контрольные вопросы

1. Что называется обводом?
2. Каким образом кривые описываются в параметрическом виде?
3. Каковы преимущества использования параметрического способа описания кривых?
4. Какими граничными параметрами характеризуется кривая в форме Фергюссона?
5. Каковы преимущества описания кривых в форме Безье?
6. Чем характеризуются кривые при описании их в форме Безье?

7. Как нужно изменить характеристическую ломаную Безье, чтобы увеличить полноту интерполирующей кривой?

8. Какое условие должно выполняться, чтобы кривая Безье имела петли?

Библиографический список

Теоретические основы формирования моделей поверхностей /Под ред. В.И.Я к у н и н а. - М.: МАИ, 1985.

Я к у н и н В.И. Геометрические основы систем автоматизированного проектирования технических поверхностей. - М.: МАИ, 1980.

Ф о к с А., П р а т т М. Вычислительная геометрия. - М.: Мир, 1982.

АППРОКСИМАЦИЯ ТАБЛИЧНЫХ ДАННЫХ КРИВЫМИ

Цель работы: закрепление и углубление теоретических знаний, а также приобретение практических навыков аппроксимации табличных данных кривыми по методу наименьших квадратов.

I. ОБЩИЕ ПОЛОЖЕНИЯ

В своей работе инженеру часто приходится иметь дело с данными, представленными в виде таблиц. Это может быть связано, например, с тем, что данные определяются экспериментально и при этом лишь для некоторых дискретных значений аргументов. Такой табличный способ задания данных неудобен для использования в системах автоматизированного проектирования, так как в этом случае бывает трудно формализовать вычислительные процедуры или же эти процедуры оказываются очень громоздкими. Поэтому на практике табличные данные часто стремятся описать некоторыми функциональными зависимостями, или, как это принято называть, аппроксимировать кривыми.

Задача аппроксимации состоит в том, что по координатам n узловых точек $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, взятых из таблиц, определяются коэффициенты a_j, $j = \overline{1,K}$ некоторой выбранной аппроксимирующей функции $y = g(a, x)$.

В задачах аппроксимации не требуется обязательного прохождения аппроксимирующей кривой через узловые точки, поэтому для решения этой задачи необходимо выбрать некоторый критерий приближения.

В математике доказано, что наилучшим критерием аппроксимации является тот, в котором оценивается минимум суммы квадратов отклонений между значениями, определяемыми выбранной кривой и таблицей. В литературе этот метод называется методом наименьших квадратов (МНК).

Фактический выбор функции $g(a,x)$ должен осуществляться с учетом специфики табличных данных, под которой понимается их периодичность, симметрия, наличие асимптотики и т.д. В некоторых случаях, когда бывает трудно отдать предпочтение той или иной функции, можно построить несколько аппроксимирующих кривых и для дальнейшего использования выбрать из них ту, у которой наименьший критерий приближения.

I.I. Метод наименьших квадратов

Пусть нам задана таблица, описывающая n точек:

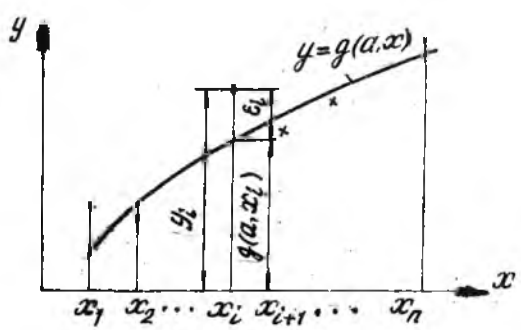
$$(x_1,y_1),(x_2,y_2),\dots,(x_n,y_n),$$

и требуется найти аппроксимирующую кривую $g(a,x)$ в диапазоне

$$x_1 \leqslant x \leqslant x_n .$$

Погрешность вычисления по функции $g(a,x)$ в каждой точке таблицы (рис.) будет составлять:

$$e_i = g(a,x_i) - y_i \ , \ i = \overline{1,n} .$$



Р и с.

Тогда сумма квадратов погрешностей определяется выражением

$$F = \sum_{i=1}^{n} [g(a,x_i) - y_i]^2 . \qquad (I)$$

В задаче необходимо определить значения коэффициентов a_j, $j=\overline{1,K}$, при которых достигается минимум функции (I).

Из курса высшей математики известно, что функция нескольких переменных достигает минимума в точке, в которой обеспечивается равенство нулю ее первых частных производных. Для нашей задачи это условие можно записать как

$$\frac{\partial E}{\partial a_1}=\frac{\partial E}{\partial a_2}=\cdots=\frac{\partial E}{\partial a_K}=0$$

или

$$\frac{\partial E}{\partial a_1}=2\sum_{i=1}^{n}\left[g(a,x_i)-y_i\right]g(a,x_i)'_{a_1}=0 \qquad (2)$$

$$\cdots$$

$$\frac{\partial E}{\partial a_K}=2\sum_{i=1}^{n}\left[g(a,x_i)-y_i\right]g(a,x_i)'_{a_K}=0.$$

Ограничимся рассмотрением случая, когда функция $g(a,x)$ представима в виде линейной комбинации элементарных функций:

$$g(a,x)=a_1g_1(x)+a_2g_2(x)+\cdots+a_Kg_K(x)=\sum_{j=1}^{K}a_jg_j(x). \qquad (3)$$

С учетом функции (3) уравнения (2) запишем в следующем виде:

$$\sum_{i=1}^{n}\left[\sum_{j=1}^{K}a_jg_j(x_i)-y_i\right]g_1(x_i)=0, \qquad (4)$$

$$\cdots$$

$$\sum_{i=1}^{n}\left[\sum_{j=1}^{K}a_jg_j(x_i)-y_i\right]g_K(x_i)=0.$$

Преобразуем уравнения (4):

$$\sum_{i=1}^{n}\sum_{j=1}^{K}a_jg_j(x_i)g_1(x_i)=\sum_{i=1}^{n}g_1(x_i)y_i, \qquad (5)$$

$$\cdots$$

$$\sum_{i=1}^{n}\sum_{j=1}^{K}a_jg_j(x_i)g_K(x_i)=\sum_{i=1}^{n}g_K(x_i)y_i.$$

Система уравнений (5) может быть записана в матричной форме

$$GA = B, \tag{6}$$

где

$$G = \begin{bmatrix} \sum_{i=1}^{n} g_1^2(x_i) & \sum_{i=1}^{n} g_1(x_i)g_2(x_i) \dots \sum_{i=1}^{n} g_1(x_i)g_K(x_i) \\ \dots \\ \sum_{i=1}^{n} g_K(x_i)g_1(x_i) & \sum_{i=1}^{n} g_K(x_i)g_2(x_i) \dots \sum_{i=1}^{n} g_K^2(x_i) \end{bmatrix},$$

$$A = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_K \end{bmatrix}, \quad B = \begin{bmatrix} \sum_{i=1}^{n} g_1(x_i)y_i \\ \dots \\ \sum_{i=1}^{n} g_K(x_i)y_i \end{bmatrix}.$$

Так как элементы матриц G и B определяются только табличными данными, то система K линейных уравнений с K неизвестными (6) может быть разрешена.

Рассмотрим частный случай задачи, когда функцию $g(a,x)$ можно представить в виде суммы двух элементарных:

$$g(a,x) = a_1 g_1(x) + a_2 g_2(x). \tag{7}$$

Как это будет показано ниже, аппроксимация табличных данных функцией с двумя коэффициентами a_1 и a_2 во многих задачах может быть сведена к случаю (7). Как правило, для этого к исходной функции требуется применить некоторый оператор преобразования (чаще всего это логарифмирование). При этом следует помнить, что и к табличным значениям функций необходимо применить тот же оператор преобразования.

Для случая (7) система уравнений (5) примет вид

$$\sum_{i=1}^{n} \left[a_1 g_1^2(x_i) + a_2 g_1(x_i)g_2(x_i)\right] = \sum_{i=1}^{n} g_1(x_i)y_i$$

$$\sum_{i=1}^{n} \left[a_1 g_1(x_i)g_2(x_i) + a_2 g_2^2(x_i)\right] = \sum_{i=1}^{n} g_2(x_i)y_i$$

или

$$a_1 \sum_{i=1}^{n} g_1^2(x_i) + a_2 \sum_{i=1}^{n} g_1(x_i)g_2(x_i) = \sum_{i=1}^{n} g_1(x_i)y_i \tag{8}$$

$$a_1 \sum_{i=1}^{n} g_1(x_i)g_2(x_i) + a_2 \sum_{i=1}^{n} g_2^2(x_i) = \sum_{i=1}^{n} g_2(x_i)y_i \,. \qquad (8)$$

Решая систему уравнений (8), определим коэффициенты a_1 и a_2:

$$a_2 = \frac{\sum_{i=1}^{n} g_1(x_i)y_i \sum_{i=1}^{n} g_1(x_i)g_2(x_i) - \sum_{i=1}^{n} g_1^2(x_i) \sum_{i=1}^{n} g_2(x_i)y_i}{\left[\sum_{i=1}^{n} g_1(x_i)g_2(x_i)\right]^2 - \sum_{i=1}^{n} g_1^2(x_i) \sum_{i=1}^{n} g_2^2(x_i)},$$

$$a_1 = \frac{\sum_{i=1}^{n} g_1(x_i)y_i - a_2 \sum_{i=1}^{n} g_1(x_i)g_2(x_i)}{\sum_{i=1}^{n} g_1^2(x_i)} \,. \qquad (9)$$

I.2. Аппроксимация прямой линией

При аппроксимации табличных данных прямой линией функция $g(a,x)$ записывается в следующем виде:

$$g(a,x) = a_1 + a_2 x, \qquad (10)$$

т.е. в этой задаче элементарные функции описываются уравнениями

$$g_1(x) = 1, \quad g_2(x) = x \,.$$

Подставляя уравнения (10) в систему уравнений (8) и учитывая, что $\sum_{i=1}^{n} g_1^2(x_i) = n$, получим формулы для вычисления коэффициентов a_1 и a_2:

$$a_2 = \frac{\sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i - n \sum_{i=1}^{n} x_i y_i}{\left(\sum_{i=1}^{n} x_i\right)^2 - n \sum_{i=1}^{n} x_i^2},$$

$$a_1 = \frac{1}{n}\left(\sum_{i=1}^{n} y_i - a_2 \sum_{i=1}^{n} x_i\right). \qquad (11)$$

Рассмотрим пример построения аппроксимирующей прямой для четырех точек (табл.)

i	I	2	3	4
x_i	-I	I	4	7
y_i	-I	-I	2	3

Подставляя табличные значения x_i, y_i, $i=\overline{1,4}$ в формулы (II), получим значения коэффициентов a_1 =-0,806 и a_2 =0,565. Следовательно, уравнение искомой прямой будет иметь вид

$$g(a,x) = -0{,}806 + 0{,}565x .$$

I.3. Аппроксимация гиперболической функцией

Уравнение гиперболы имеет вид $g(a,x)=a_1+a_2/x$. Для данной задачи элементарные функции описываются уравнениями

$$g_1(x)=1,\ g_2(x)=1/x. \qquad (I2)$$

Подставляя уравнения (I2) в систему уравнений (8), после соответствующих преобразований получим

$$a_2=\frac{\sum_{i=1}^{n} y_i \sum_{i=1}^{n}\frac{1}{x_i} - n\sum_{i=1}^{n}\frac{y_i}{x_i}}{\left(\sum_{i=1}^{n}\frac{1}{x_i}\right)^2 - n\sum_{i=1}^{n}\frac{1}{x_i}},$$

$$a_1=\frac{1}{n}\left(\sum_{i=1}^{n} y_i - a_2\sum_{i=1}^{n}\frac{1}{x_i}\right).$$

I.4. Аппроксимация степенной функцией

Рассмотрим аппроксимацию табличных данных функцией вида

$$g(b,x)=b_1 x^{b_2}. \qquad (I3)$$

Для представления функции (I3) в виде суммы двух элементарных функций прологарифмируем ее

$$\ln g(b,x) = \ln b_1 + b_2 \ln x$$

и преобразуем

$$\ln g(b,x) = a_1 g_1(x) + a_2 g_2(x), \qquad (I4)$$

где

$$a_1 = \ln b_1 \ , \ a_2 = b_2 \ , \qquad (I5)$$

$$g_1(x) = 1, \ g_2(x) = \ln x .$$

Учитывая вид функции (I4), вместо табличных значений функций y_i, $i = \overline{1,n}$ необходимо брать соответствующие значения их логарифмов

$$y_i^* = \ln y_i \ , \ i = \overline{1,n} . \qquad (I6)$$

Подставив зависимости (I5) и (I6) в систему уравнений (8), найдем искомые значения коэффициентов

$$b_2 = a_2 = \frac{\sum_{i=1}^{n} \ln x_i \sum_{i=1}^{n} \ln y_i - n \sum_{i=1}^{n} \ln x_i \ln y_i}{\left(\sum_{i=1}^{n} \ln x_i\right)^2 - n \sum_{i=1}^{n} (\ln x_i)^2} ,$$

$$b_1 = e^{a_1} = \exp\left[\frac{1}{n}\left(\sum_{i=1}^{n} \ln y_i - b_2 \sum_{i=1}^{n} \ln x_i\right)\right].$$

I.5. Аппроксимация экспоненциальной функцией

Для определения коэффициентов a_1 и a_2 аппроксимирущей функции $g(a,x) = e^{(a_1 + a_2 x)}$ преобразуем ее к сумме двух элементарных функций. Для этого прологарифмируем эту функцию:

$$\ln g(a,x) = a_1 + a_2 x. \qquad (I7)$$

Учитывая выражение (I7), в данной задаче будем использовать следующие зависимости:

$$g_1(x)=1,\ g_2(x)=x,\ y_i^*=\ln y_i,\ i=\overline{1,n}. \qquad (I8)$$

Подставив уравнение (I8) в систему уравнений (8), определим выражения для вычисления коэффициентов a_1 и a_2:

$$a_2=\frac{\sum_{i=1}^{n} x_i \sum_{i=1}^{n} \ln y_i - n\sum_{i=1}^{n} x_i \ln y_i}{\left(\sum_{i=1}^{n} x_i\right)^2 - n\sum_{i=1}^{n} (x_i)^2},$$

$$a_1=\frac{1}{n}\left(\sum_{i=1}^{n} \ln y_i - a_2 \sum_{i=1}^{n} x_i\right).$$

I.6. Аппроксимация показательной функцией

Для сведения данной задачи к условию (7) показательную функцию $g(b,x)=b_1 b_2^x$ необходимо прологарифмировать $\ln g(b,x)=\ln b_1+x\ln b_2$. Следовательно, в дальнейшем мы должны использовать зависимости

$$a_1=\ln b_1,\ a_2=\ln b_2,$$

$$g_1(x)=1,\ g_2(x)=x,\ y_i^*=\ln y_i,\ i=\overline{1,n}. \qquad (I9)$$

С учетом зависимостей (I9) выражения для вычисления коэффициентов b_1 и b_2 примут вид

$$b_2=e^{a_2}=\exp\left[\frac{\sum_{i=1}^{n} \ln y_i \sum_{i=1}^{n} x_i - n\sum_{i=1}^{n} x_i \ln y_i}{\left(\sum_{i=1}^{n} x_i\right)^2 - n\sum_{i=1}^{n} (x_i)^2}\right],$$

$$b_1=e^{a_1}=\exp\left[\frac{1}{n}\left(\sum_{i=1}^{n} \ln y_i - \ln b_2 \sum_{i=1}^{n} x_i\right)\right].$$

I.7. Аппроксимация логарифмической функцией

Логарифмическая функция $g(a,x)=a_1+a_2 \ln x$ соответствует условию (7), поэтому в данной задаче можно использовать следующие зависимости для элементарных функций:

$$g_1(x)=1,\ g_2(x)=\ln x. \qquad (20)$$

Подставляя зависимости (20) в систему уравнений (8), получим следующие выражения для вычисления коэффициентов a_1 и a_2:

$$a_2=\frac{\sum_{i=1}^{n} y_i \sum_{i=1}^{n} \ln x_i - n\sum_{i=1}^{n} x_i \ln y_i}{\left(\sum_{i=1}^{n} \ln x_i\right)^2 - n\sum_{i=1}^{n} (\ln x_i)^2},$$

$$a_1=\frac{1}{n}\left(\sum_{i=1}^{n} y_i - a_2 \sum_{i=1}^{n} \ln x_i\right).$$

2. ИНСТРУКЦИЯ ПО РАБОТЕ С ЭВМ

Работа выполняется на ЭВМ СМ I420 с использованием программы *APROX*, находящейся в разделе [I30,I00] тома (пакета сменных магнитных дисков) с именем *STUD*, устанавливаемого на устройство *DK1*:

После регистрации в системе необходимо смонтировать устройство *DK1*: командой

```
> MOU  DK1 : STUD
```

Обращение к программе

```
> RUN  DK1 : [130, 100] APROX
```

Программа *APROX* обеспечивает определение коэффициентов для нескольких аппроксимирующих функций. Работа с программой ведется в интерактивном режиме по следующей схеме:

I. Ввод исходных данных (размерность таблицы и сами табличные данные).

2. В случае необходимости коррекция введенных данных.

3. Выбор аппроксимирующей функции из числа предлагаемых.

4. Распечатка результатов аппроксимации (коэффициентов аппроксимирующих функций, величины критерия приближения по методу МНК, значений аппроксимирующих функций для соответствующих узловых точек табличных данных).

Работа на каждом этапе сопровождается подробными комментариями на экране дисплея и в дополнительных пояснениях не нуждается.

3. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Продолжительность лабораторной работы - 4 ч. Работа выполняется по следующей примерной схеме.

1. Изучение материалов лабораторной работы во внеаудиторное время занятий.

2. Получение у преподавателя индивидуального задания.

3. Вывод формул для вычисления коэффициентов заданной аппроксимирующей функции.

4. Определение на ЭВМ коэффициентов аппроксимирующих функций для заданных табличных данных.

5. Из нескольких рассмотренных аппроксимирующих функций выбор наилучшей по минимуму критерия МНК.

6. Оформление отчета по работе.

4. ОФОРМЛЕНИЕ ОТЧЕТА

Отчет оформляется на отдельных листах бумаги с указанием фамилии и учебной группы студента, а также варианта индивидуального задания. В отчете требуется описать процедуру вывода формулы для определения коэффициентов аппроксимирующих функций. Необходимо также графически проиллюстрировать аппроксимации табличных данных исследуемыми функциями и указать наилучшую из них по минимуму критерия МНК.

Контрольные вопросы

1. В чем заключается задача аппроксимации?

2. Почему для решения задачи аппроксимации необходимо выбирать критерий приближения?

3. Какой критерий аппроксимации используется в методе наименьших квадратов?

4. Чем вызвано стремление представить исследуемую аппроксимирующую функцию в виде линейной комбинации элементарных функций?

Библиографический список

Я к у н и н В.И. Геометрические основы систем автоматизированного проектирования технических поверхностей. – М.: МАИ, 1980.

Ш у п п Т. Решение инженерных задач на ЭВМ. Практическое руководство / Пер. с англ. – М.: Мир, 1982.

ПРИМЕР ОРГАНИЗАЦИИ ДИАЛОГА С ИСПОЛЬЗОВАНИЕМ ФАЙЛА СИНТАКСИЧЕСКОГО ДЕРЕВА

В данном примере рассматривается вариант диалога для информационно-поисковой системы, ориентированной на обработкустатистических данных самолетов: просмотр, добавление новых и изменение уже существующих данных. Предполагается, что эта система строится в соответствии с некоторой принятой классификацией самолетов.

Для файла синтаксического дерева диалога выделяется четыре группы ключевых слов:

1. Просмотреть, добавить, изменить.

2. Военный, гражданский.

3. Истребитель, штурмовик, бомбардировщик, ракетоносец, военно-транспортный.

4. Пассажирский, транспортный, сельхозавиации.

Переменным словом является название фирмы изготовителя.

ПРИМЕР ОФОРМЛЕНИЯ ФАЙЛА СИНТАКСИЧЕСКОГО
ДЕРЕВА ДИАЛОГА

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678
        10        20        30        40        50        60        70
                  ЗАДАНИЕ ДЛИНЫ ПЕРЕМЕННОГО СЛОВА
005 ! 20 ! ПЕРЕМЕННОЕ СЛОВО ДЛЯ НАЗВАНИЯ ФИРМЫ-ИЗГОТОВИТЕЛЯ
007 !   УКАЖИТЕ РЕЖИМ РАБОТЫ СО СТАТИСТИЧЕСКИМИ ДАННЫМИ САМОЛЕТОВ :
008 !
009 !       ПРОСМОТРЕТЬ      ДОБАВИТЬ       ИЗМЕНИТЬ
011 !   П Е Р В А Я   Г Р У П П А   К Л Ю Ч Е В Ы Х   С Л О В
        10        20        30        40        50        60        70
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678
015 ! 01 ! ПРОСМОТРЕТЬ .....! 021 ! 027 !
016 ! 02 ! ДОБАВИТЬ .....! 021 ! 027 !
017 ! 03 ! ИЗМЕНИТЬ .....! 021 ! 027 !
019 !          СПРАВКА ПО ВТОРОЙ ГРУППЕ КЛЮЧЕВЫХ СЛОВ
020 !==========================================================
021 !              УКАЖИТЕ НАЗНАЧЕНИЕ САМОЛЕТА :
022 !
023 !                ВОЕННЫЙ       ГРАЖДАНСКИЙ
025 !                ВТОРАЯ ГРУППА КЛЮЧЕВЫХ СЛОВ
027 ! 01 ! ВОЕННЫЙ .....! 032 ! 037 !
028 ! 02 ! ГРАЖДАНСКИЙ .....! 045 ! 049 !
030 !          СПРАВКА ПО ТРЕТЬЕЙ ГРУППЕ КЛЮЧЕВЫХ СЛОВ
031 !==========================================================
032 !            УКАЖИТЕ ТИП ВОЕННОГО САМОЛЕТА :
033 !
034 !      ИСТРЕБИТЕЛЬ    ШТУРМОВИК    БОМБАРДИРОВЩИК
035 !          РАКЕТОНОСЕЦ    ВОЕННО-ТРАНСПОРТНЫЙ
037 ! 01 ! ИСТРЕБИТЕЛЬ .....! 02 ! 053
038 ! 02 ! ШТУРМОВИК .....! 02 ! 053
039 ! 03 ! БОМБАРДИРОВЩИК .....! 02 ! 053
040 ! 04 ! РАКЕТОНОСЕЦ .....! 02 ! 053
041 ! 05 ! ВОЕННО-ТРАНСПОРТНЫЙ .....! 02 ! 053
043 !          СПРАВКА ПО ЧЕТВЕРТОЙ ГРУППЕ КЛЮЧЕВЫХ СЛОВ
044 !==========================================================
045 !            УКАЖИТЕ ТИП ГРАЖДАНСКОГО САМОЛЕТА :
046 !
047 !    ПАССАЖИРСКИЙ    ТРАНСПОРТНЫЙ  СЕЛЬХОЗАВИАЦИИ
049 ! 01 ! ПАССАЖИРСКИЙ .....! 01 ! 053
050 ! 02 ! ТРАНСПОРТНЫЙ .....! 01 ! 053
051 ! 03 ! СЕЛЬХОЗАВИАЦИИ .....! 01 ! 053
053 !  УКАЖИТЕ НАЗВАНИЕ ФИРМЫ - ИЗГОТОВИТЕЛЯ (НЕ БОЛЕЕ 20 СИМВОЛОВ)
```

```
C  ============================================================
C    ПРИМЕР ОБРАЩЕНИЯ К ПРОГРАММЕ GUIDE ДЛЯ ОРГАНИЗАЦИИ      !
C  ДИАЛОГА С ИСПОЛЬЗОВАНИЕМ ФАЙЛА СИНТАКСИЧЕСКОГО ДЕРЕВА     !
C  ============================================================

       CHARACTER*11 FILNAM
       BYTE    CMLIN(78),NUMWDS(10),LENVFL(4),CSTR(78)
       COMMON  /CMLCOM/ CMLIN /REF/ KOL,NUMWDS,LENVFL
       COMMON  /CSTRIN/ CSTR

C ==========================================================
C                                                          !
C              НАЗНАЧЕНИЕ COMMON ОБЛАСТЕЙ:                 !
C             ------------------------------               !
C CMLCOM:       БУФЕР ПЕРЕМЕННЫХ ПОЛЕЙ КОМАНДНЫХ СТРОК     !
C REF    :      KOL     - КОЛИЧЕСТВО ВВЕДЕННЫХ СЛОВ        !
C              NUMWDS  - НОМЕРА ВВЕДЕННЫХ СЛОВ            !
C              LENVFL  - ДЛИНЫ ПЕРЕМЕННЫХ ПОЛЕЙ            !
C CSTRIN:      БУФЕР КОМАНДНОЙ СТРОКИ                      !
C                                                          !
C ==========================================================

  10    DO 20   L=1,78
           CSTR(L) =0
  20       CMLIN(L)=0
       FILNAM='AIRPLAN.TRE'
C
C          ОБРАЩЕНИЕ К ПРОГРАММЕ GUIDE:
C           ____ НОМЕР ПАРАМЕТРА GUIDE
C          /
C          !       Н А З Н А Ч Е Н И Е:
C          V
C ВХОД:    1 - НОМЕР LUN ДЛЯ ФАЙЛА СИНТАКСИЧЕСКОГО ДЕРЕВА
C          2 - СПЕЦИФИКАЦИЯ ФАЙЛА СИНТАКСИЧЕСКОГО ДЕРЕВА
C          3 - ОТЗЫВ
C          4 - НОМЕР СТРОКИ НАЧАЛА ПЕРВОЙ ГРУППЫ КЛ.СЛОВ
C          5 - НОМЕР СТРОКИ HELP ДЛЯ ПЕРВОЙ ГРУППЫ СЛОВ
C ВЫХОД:   6 - НОМЕР ТЕРМИНАТОРА ( ВК,CTRL/U,CTRL/Z )
C          7 - КОД ЗАВЕРШЕНИЯ ПРОГРАММЫ GUIDE
C
C                   1    2        3     4 5   6    7
       CALL  GUIDE(1,FILNAM,'КОМАНДА)',15,7,ITERM,IER)
C
C  АНАЛИЗ РЕЖИМОВ ЗАВЕРШЕНИЯ РАБОТЫ ПРОГРАММЫ
C
C
       IF ( IER  .NE.0 ) GOTO 40        ! ПО ОШИБКЕ
       IF ( ITERM.EQ.2 ) STOP           ! ПО CTRL/Z
       IF ( ITERM.EQ.1 ) GOTO 10        ! ПО CTRL/U
       IF (  KOL .EQ.0 ) GOTO 10        ! КЛ. СЛОВА
C                                       ! НЕ ВВОДИЛИСЬ
```

```
C     ВЫВОД НА ТЕРМИНАЛ  ВОЗВРАЩАЕМЫХ П/П  GUIDE ПАРАМЕТРОВ
C     ==================================================
C
C
C          ЗАМЕНА 0 НА "_" В БУФЕРАХ  CSTR И CMLIN
C          ---------------------------------------
C
        DO 30 L= 1,78
           IF ( CSTR(L)  .EQ. 0 ) CSTR(L) ='_'
  30       IF ( CMLIN(L) .EQ. 0 ) CMLIN(L)='_'

C       ВЫВОД БУФЕРА ПЕРЕМЕННЫХ ПОЛЕЙ И КОМАНДНОЙ СТРОКИ
C       ------------------------------------------------

        TYPE 900,  CSTR,CMLIN
900     FORMAT(//1X,78('-'),
     &   //' ПРИМЕЧАНИЕ :',
     &     ' В ВОЗВРАЩАЕМЫХ МАССИВАХ 0 ЗАМЕНЕН НА  "_"',
     &   //' ВВЕДЕНА КОМАНДНАЯ СТРОКА ( COMMON CSTRIN ) :'
     &    /1X,78A1
     &   //' ЗНАЧЕНИЯ ПЕРЕМЕННЫХ СЛОВ ( COMMON CMLCOM ) :'
     &    /1X,78A1)

C            ВЫВОД НОМЕРОВ СЛОВ И ДЛИН  ПОЛЕЙ
C            --------------------------------

        TYPE 910 ,  NUMWDS,LENVFL
910     FORMAT (//' COMMON REF :'/
     &   ' НОМЕРА СЛОВ : ',10I4/1X,'ДЛИНЫ ПОЛЕЙ : ',4I4)
        STOP
C              ЗАВЕРШЕНИЕ ПРОГРАММЫ С ОШИБКОЙ
C              ------------------------------

  40    TYPE 920,IER
920     FORMAT(' КОД  ОШИБКИ = ',I6)
        STOP
        END
```

Содержание

Св.план 1987, поз. 89

Валерий Дмитриевич Еленев

ЛАБОРАТОРНЫЙ ПРАКТИКУМ
ПО ИСПОЛЬЗОВАНИЮ САПР

Редактор Е.Д.Антипова
Техн.редактор Н.М.Каленюк
Корректор Н.С.Куприянова

Подписано в печать 24.09.87 г. Е00314.
Формат 60x84 I/I6. Бумага оберточная белая.
Печать оперативная. Усл.п.л. 3,9. Уч.-изд.л. 4,0.
Т. 300 экз. Заказ 7397. Цена I5 к.

Куйбышевский ордена Трудового Красного Знамени авиационный институт имени академика С.П.Королева, г. Куйбышев, ул. Молодогвардейская, I5I.

Полиграфическое объединение, г.Куйбышев, ул. Венцека,60.