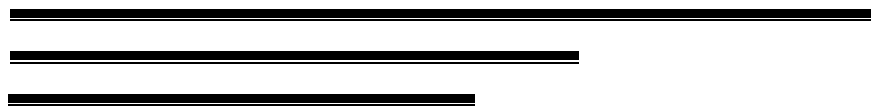




ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ ПРОЕКТИРОВАНИЯ РЭС



◆ ИЗДАТЕЛЬСТВО ТГТУ ◆

Министерство образования и науки Российской Федерации
ГОУ ВПО «Тамбовский государственный технический университет»

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ ПРОЕКТИРОВАНИЯ РЭС

Методические указания по выполнению лабораторных работ
для студентов 3, 4 курсов специальности 210201
«Информационные технологии проектирования РЭС»
дневной и заочной форм обучения



Тамбов
Издательство ТГТУ
2008

УДК 621.37/39(07)
ББК 844-02я73
М915

Рекомендовано Редакционно-издательским советом ТГТУ

Рецензент

Доктор технических наук, профессор ТГТУ
В.А. Погонин

Составители:

Ю.Л. Муромцев,
А.Н. Грибков

М915 Информационные технологии проектирования РЭС : методические указания по выполнению лабораторных работ / сост. : Ю.Л. Муромцев, А.Н. Грибков. – Тамбов : Изд-во Тамб. гос. техн. ун-та, 2008. – 32 с. – 100 экз.

Содержат рекомендации по выполнению комплекса лабораторных работ по дисциплине «Информационные технологии проектирования РЭС» с использованием CASE-технологий и программного продукта EXCEL.

Предназначены для студентов, обучающихся по специальности 210201 (направление 210200) дневного и заочного отделений.

УДК 621.37/39(07)

ББК 844-02я73

© ГОУ ВПО «Тамбовский государственный
технический университет» (ТГТУ), 2008

Учебное издание

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ ПРОЕКТИРОВАНИЯ РЭС

Методические указания

С о с т а в и т е л и:

МУРОМЦЕВ Юрий Леонидович,
ГРИБКОВ Алексей Николаевич

Редактор Т.М. Глинкина
Компьютерное макетирование Т.Ю. Зотовой

Подписано в печать 05.03.2008
Формат 60×84/16. 1,86 усл. печ. л. Тираж 100 экз. Заказ № 110

Издательско-полиграфический центр ТГТУ
392000, Тамбов, Советская, 106, к. 14

ЗНАКОМСТВО С IDEF-ТЕХНОЛОГИЯМИ И СРЕДОЙ VPWIN

Цель работы: Знакомство с интерфейсом VPwin и технологией создания функциональных моделей. Правильное заполнение каркаса и свойств диаграммы.

Методические указания

VPwin имеет достаточно простой и интуитивно понятный интерфейс пользователя, дающий возможность аналитику создавать сложные модели при минимальных усилиях.

На рис. 1 показана интегрированная среда разработки модели VPwin. Она состоит из главного меню, панелей инструментов, навигатора модели и рабочего поля модели.

Запуск VPwin осуществляется при помощи кнопки Пуск/VPWin. Если появляется диалог ModelMart Connection Manager, следует нажать на кнопку Cancel.

Для создания новой функциональной модели надо войти в меню File/New. Появляется диалог I would like to. Введите имя (Name) и тип (Type) модели. Нажмите OK. Появляется окно Properties for New Models. Во вкладке General введите автора диаграммы и нажмите OK. Автоматически создается контекстная диаграмма. Перейдите в меню Model/Model Properties. Во вкладке General диалога Model Properties следует внести имя проекта (Project). Обратите внимание на то, что по умолчанию установлен тип модели – «как есть» (Time Frame: AS-IS). Во вкладке Purpose внесите цель (Purpose) и точку зрения (Viewpoint). Нажмите OK.

Для сохранения модели войдите в меню File/Save. В появившемся окне «Сохранить как» выберите каталог и введите имя файла модели. Нажмите на кнопку «Сохранить».

Для печати диаграммы войдите в меню File/Save и в появившемся окне «Print» нажмите на кнопку OK.

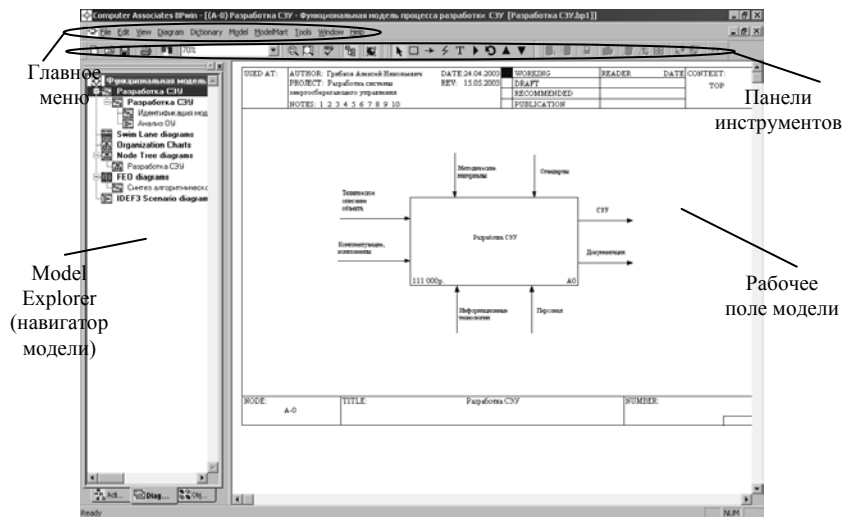


Рис. 1. Интегрированная среда разработки модели VPwin 4.0

Порядок выполнения работы

1. Запустите VPwin.
2. Ознакомьтесь с интерфейсом среды VPwin.
3. Создайте новую модель в нотации IDEF0.
4. Введите автора диаграммы, название проекта, цель моделирования и точку зрения.
5. Сохраните результат на диске.

Содержание отчета

1. Название и цель лабораторной работы.
2. Название моделируемого процесса.
3. Название проекта, автора работы, цели моделирования и точки зрения.
4. Распечатка каркаса диаграммы.

ПОСТРОЕНИЕ КОНТЕКСТНОЙ ДИАГРАММЫ



Цель работы: Ознакомление с технологией и методикой построения диаграммы нулевого уровня. Построение контекстной диаграммы бизнес-процесса в среде VPwin.

Методические указания


Эта и все последующие лабораторные работы связаны друг с другом и выполняются последовательно в одном и том же файле. Для загрузки файла модели необходимо в главном меню выбрать пункт File/Open в появившемся окне «Открыть» выбрать файл, сохраненный в ходе выполнения предыдущей работы и нажать на кнопку «Открыть».

Для задания наименования работы на контекстной диаграмме щелкните правой кнопкой мыши по работе. В контекстном меню выберите Name. Во вкладке Name введите имя работы.

Для создания граничной стрелки входа:

- щелкните по кнопке с символом стрелки  в палитре инструментов и перенесите курсор к левой стороне экрана, пока не появится начальная темная полоска;
- щелкните один раз по полоске и еще раз в левой части работы со стороны входа;
- вернитесь на палитру инструментов и выберите опцию редактирования стрелки ;
- щелкните правой кнопкой мыши на линии стрелки, во всплывающем меню выберите Name и добавьте имя стрелки во вкладке Name диалога Arrow Properties. Нажмите ОК.

Граничные стрелки выхода, управления и механизма изображаются аналогично.

Для создания текста на поле диаграммы нажмите на кнопку с символом  на панели инструментов и щелкните на свободное место диаграммы. В появившемся окне Text Block Properties внесите точку зрения и цель моделирования. Нажмите ОК.

Пример контекстной диаграммы приведен в приложении А.

Последовательность выполнения работы

1. Загрузите модель, полученную в результате предыдущей работы.
2. Введите наименование работы на диаграмме.
3. Создайте граничные стрелки входов, выходов, управлений и механизмов.
4. В левом нижнем углу диаграммы введите точку зрения и цель моделирования.
5. Сохраните результат на диске.

Содержание отчета

1. Название и цель лабораторной работы.
2. Таблица стрелок контекстной диаграммы (табл. 1).
3. Распечатка контекстной диаграммы.

1. Стрелки контекстной диаграммы


Наименование (Arrow Name)	Описание (Arrow Definition)	Тип (Arrow Type)

Лабораторная работа 3


ПОСТРОЕНИЕ ДИАГРАММЫ ДЕКОМПОЗИЦИИ ПЕРВОГО УРОВНЯ


Цель работы: Изучение методики построения диаграммы декомпозиции первого уровня.

Методические указания

Для декомпозиции диаграммы выберите в палитре инструментов кнопку перехода на нижний уровень  и в диалоге Activity Box Count установите число работ на диаграмме нижнего уровня и нажмите ОК. Автоматически будет создана диаграмма декомпозиции первого уровня.

Для внесения наименований работ правой кнопкой мыши щелкните по первой работе, выберите свойство Name, внесите имя работы и нажмите ОК. Повторите операцию для всех работ.

Если окажется, что количество работ недостаточно, то работу можно добавить в диаграмму, щелкнув сначала по кнопке  на палитре инструментов, а затем по свободному месту на диаграмме.

Для связи работ между собой используются внутренние стрелки, т.е. стрелки, которые не касаются границы диаграммы, начинаются у одной и кончаются у другой работы. Для рисования внутренней стрелки необходимо в режиме рисования стрелок (кнопка  на палитре инструментов) щелкнуть по сегменту одной работы (например, выхода) и затем по сегменту (например, входа) другой работы. Для разветвления стрелки нужно в режиме редактирования стрелки щелкнуть по фрагменту стрелки и по соответствующему сегменту работы. Для именованной разветвляющихся стрелок необходимо щелкнуть правой кнопкой мыши по ветви стрелки, которой надо дать название, в контекстном меню выбрать пункт Name, ввести наименование ветви и нажать на кнопку ОК. Для слияния двух стрелок выхода нужно в режиме редактирования стрелки сначала щелкнуть по сегменту выхода работы, а затем по соответствующему сегменту стрелки.

Пример диаграммы декомпозиции первого уровня приведен в приложении Б.

Последовательность выполнения работы

1. Загрузите модель, полученную в результате предыдущей работы.
2. Декомпозируйте контекстную диаграмму.
3. Внесите названия работ на диаграмме первого уровня.
4. Свяжите граничные стрелки с работами и задайте их наименования.
5. Свяжите работы внутренними стрелками и задайте их наименования.
6. Сохраните результат на диске.

Содержание отчета

1. Название и цель лабораторной работы.
2. Таблица работ диаграммы первого уровня (табл. 1).
3. Таблица стрелок диаграммы декомпозиции первого уровня (табл. 2).

1. Работы диаграммы декомпозиции первого уровня

Номер работы	Наименование (Activity Name)	Описание (Definition)

2. Стрелки диаграммы декомпозиции первого уровня


Наименование (Arrow Name)	Описание (Arrow Definition)	Тип (Arrow Type)

Лабораторная работа 4

ПОСТРОЕНИЕ ДИАГРАММЫ ДЕКОМПОЗИЦИИ ВТОРОГО УРОВНЯ

Цель работы: Изучение методики построения диаграммы декомпозиции второго уровня.

Методические указания

Для получения диаграммы декомпозиции второго уровня перейдите на диаграмму первого уровня, выберите в палитре инструментов кнопку перехода на нижний уровень  и выберите работу для декомпозиции, щелкнув по ней левой кнопкой мыши. В появившемся диалоге Activity Box Count установите число работ на диаграмме второго уровня и нажмите ОК. Автоматически будет создана диаграмма декомпозиции второго уровня.

Задание названий работ и расстановка стрелок производится точно так же, как и на диаграмме первого уровня.

Пример диаграммы декомпозиции второго уровня приведен в приложении В.

Последовательность выполнения работы

1. Загрузите модель, полученную в результате предыдущей работы.
2. Декомпозируйте одну из работ диаграммы первого уровня.
3. Внесите названия работ на диаграмме второго уровня.
4. Свяжите граничные стрелки с работами и задайте их наименования.
5. Свяжите работы внутренними стрелками и задайте их наименования.
6. Сохраните результат на диске.

Содержание отчета

1. Название и цель лабораторной работы.
2. Таблица работ диаграммы второго уровня (табл. 1).
3. Таблица стрелок диаграммы декомпозиции второго уровня (табл. 2).

4. Распечатка диаграммы второго уровня.

1. Работы диаграммы декомпозиции второго уровня

Номер работы	Наименование (Activity Name)	Описание (Definition)

2. Стрелки диаграммы декомпозиции второго уровня

Наименование (Arrow Name)	Описание (Arrow Definition)	Тип (Arrow Type)

Лабораторная работа 5

ПОСТРОЕНИЕ ДИАГРАММЫ УЗЛОВ

Цель работы: Изучение методики построения диаграммы узлов.

Методические указания

Для создания диаграммы узлов выберите в меню пункт Diagram/Add Node Tree. В диалогe Node Tree Wizard – Step 1 of 2 внесите имя диаграммы (Node Tree Name), укажите диаграмму корня дерева (по умолчанию – A0) и количество уровней (Number of Levels). Нажмите кнопку «Готово». Создается диаграмма дерева узлов.

Пример диаграммы узлов приведен в приложении Г.

Последовательность выполнения работы

1. Загрузите модель, полученную в результате предыдущей работы.
2. Создайте диаграмму узлов.
3. Сохраните результат на диске.

Содержание отчета

1. Название и цель лабораторной работы.
2. Распечатка диаграммы узлов.

Лабораторная работа 6


ПОСТРОЕНИЕ FEO-ДИАГРАММЫ

Цель работы: Изучение методики построения FEO-диаграмм.

Методические указания

Для создания FEO-диаграммы выберите в меню пункт Diagram/Add FEO Diagram. В диалогe Add New FEO Diagram внесите имя FEO-диаграммы (Name of new diagram) и выберите исходную диаграмму (Source Diagram Name). Нажмите ОК.

Удалите лишние стрелки и работы на FEO-диаграмме. Для удаления щелкните левой кнопкой мыши по стрелке или работе и нажмите на клавишу Delete. После удаления всех лишних стрелок и работ создание диаграммы завершено.

Для перехода между стандартной диаграммой, деревом узлов и FEO используйте кнопку  на панели инструментов.

Пример FEO-диаграммы приведен в приложении Д.

Последовательность выполнения работы

1. Загрузите модель, полученную в результате предыдущей работы.
2. Создайте диаграмму FEO.
3. Удалите лишние стрелки и работы.
4. Сохраните результат на диске.

Содержание отчета


1. Название и цель лабораторной работы.
2. Распечатка FEO-диаграммы.

Лабораторная работа 7


ПОСТРОЕНИЕ ДИАГРАММЫ В СТАНДАРТЕ IDEF3

Цель работы: Изучение методики построения диаграммы в нотации IDEF3.

Методические указания

Для создания диаграммы декомпозиции в стандарте IDEF3 перейдите на диаграмму, которую вы хотите декомпозировать, нажмите в палитре инструментов на кнопку перехода на нижний уровень  и выберите работу для декомпозиции, щелкнув по ней левой кнопкой мыши. В появившемся диалоге Activity Box Count выберите нотацию – IDEF3 и установите число работ на диаграмме. Нажмите ОК. Автоматически будет создана диаграмма декомпозиции в стандарте IDEF3, содержащая работы (UOW).

Для внесения наименования работы правой кнопкой мыши щелкните по работе, выберите в контекстном меню свойство Name и внесите имя работы. Затем во вкладке Definition внесите определение. Во вкладке UOW внесите свойства работы (Objects, Facts, description, Constrains). Нажмите ОК.

Для создания перекрестка нажмите на кнопку  на палитре инструментов, щелкните левой кнопкой мыши на поле модели и в появившемся окне выберите нужный тип перекрестка, нажав по нему левой кнопкой мыши.

Связи работ и перекрестков стрелками осуществляются аналогичным образом, как и в диаграммах IDEF0.

Пример диаграммы IDEF3 приведен в приложении Е.

Последовательность выполнения работы

1. Загрузите модель, полученную в результате предыдущей работы.
2. Создайте диаграмму IDEF3.
3. Введите названия, определения и свойства работ (UOW).
4. Создайте перекрестки.
5. Соедините работы и перекрестки стрелками.
6. Сохраните результат на диске.

Содержание отчета

1. Название и цель лабораторной работы.
2. Таблица работ диаграммы IDEF3 (табл. 1).
3. Распечатка диаграммы IDEF3.

1. Работы диаграммы декомпозиции IDEF3


Наименование (Activity Name)	Описание (Definition)	Свойства UOW

Лабораторная работа 8


ПОСТРОЕНИЕ ДИАГРАММЫ ПОТОКОВ ДАННЫХ DFD

Цель работы: Изучение методики построения диаграмм потоков данных в нотации DFD.

Методические указания

Для создания диаграммы декомпозиции в стандарте DFD перейдите на диаграмму, которую вы хотите декомпозировать, нажмите в палитре инструментов на кнопку перехода на нижний уровень  и выберите работу для декомпозиции, щелкнув по ней левой кнопкой мыши. В появившемся диалоге Activity Box Count выберите нотацию – DFD и установите число работ на диаграмме. Нажмите ОК. Автоматически будет создана диаграмма декомпозиции в стандарте DFD.

Для внесения наименования работы правой кнопкой мыши щелкните по работе, выберите в контекстном меню свойство Name и внесите имя работы. Затем во вкладке Definition внесите определение. Нажмите ОК.

Для создания хранилища данных выберите кнопку  на палитре компонентов и щелкните левой кнопкой мыши на свободном месте диаграммы. В появившемся окне Data Store введите наименование хранилища данных и нажмите на кнопку ОК.

Связи работ и хранилищ данных стрелками осуществляются аналогичным образом, как и в диаграммах IDEF0.

Пример диаграммы DFD приведен в приложении Ж.

Последовательность выполнения работы

1. Загрузите модель, полученную в результате предыдущей работы.
2. Создайте диаграмму DFD
3. Внесите хранилища данных.
4. Удалите лишние граничные стрелки.
5. Создайте внутренние ссылки.
6. Сохраните результат на диске.

Содержание отчета

1. Название и цель лабораторной работы.
2. Таблица работ диаграммы DFD (табл. 1).
3. Таблица хранилищ данных (табл. 2).
4. Распечатка диаграммы DFD.

1. Работы диаграммы DFD

Номер работы	Наименование (Activity Name)	Описание (Definition)

2. Хранилища данных диаграммы DFD

Номер	Наименование (Name)	Описание (Definition)

Лабораторная работа 9

СТОИМОСТНОЙ АНАЛИЗ

Цель работы: Изучение методики стоимостного анализа (ABC).

Методические указания

Сначала необходимо установить единицы измерения времени и денег, для этого выберите в главном меню пункт Model/Model Properties и во вкладке ABC Units установите следующие значения параметров: Currency description – Рубль, Symbol placement – 1 p., Symbol – p. Time Unit – Days. Нажмите ОК.

Для определения центров затрат выберите пункт меню Dictionary/Cost Center и в диалоге Cost Center Dictionary внесите названия (Name) и описания (Definition) центров затрат.

Для задания стоимости работы (для каждой работы на диаграмме декомпозиции) щелкните правой кнопкой мыши по работе и во всплывающем меню выберите свойство Costs. Во вкладке Costs диалог Activity Properties укажите частоту проведения данной работы в рамках общего процесса (Frequency) и ее продолжительность (Duration). Затем выберите в списке один из центров затрат и задайте его стоимость. Аналогично назначьте суммы по каждой статье расходов. Нажмите ОК. Если в процессе назначения стоимости возникает необходимость внесения дополнительных центров затрат, диалог Cost Center Editor вызывается прямо из диалога Activity Cost соответствующей кнопкой.

Для формирования отчета по стоимостному анализу выберите в главном меню Tools/Reports/Activity Cost Report. Выберите стандартный отчет Standart Report – Activity Cost / Cost Center (landscape). Для просмотра отчета нажмите кнопку Preview, для печати – Print. После завершения просмотра и печати отчета нажмите на кнопку Close.

Последовательность выполнения работы

1. Загрузите модель, полученную в результате предыдущей работы.
2. Установите единицы измерения стоимости и времени.
3. Внесите названия и определения центров затрат.
4. Внесите стоимости и длительности каждой работы модели.
5. Создайте отчет Activity Cost Report.
6. Сохраните результат на диске.

Содержание отчета

1. Название и цель лабораторной работы.
2. Таблица центров затрат (табл. 1).
3. Отчет Activity Cost Report.
4. Распечатка контекстной диаграммы.

1. Таблица центров затрат

Наименование (Name)	Описание (Definition)

ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ

Цель работы: освоение методов автоматизированного решения задач линейного программирования с использованием пакетов прикладных программ.

Методические указания

Для решения задачи методом линейного программирования необходимо, чтобы описанная в ней ситуация отвечала пяти основным условиям.

Во-первых, она должна быть связана с ограниченными ресурсами (т.е. ограниченное число рабочих, оборудования, финансов и т.д.), в противном случае этой задачи просто бы не существовало. Во-вторых, необходимо сформулировать точную цель (максимальная прибыль или минимальные затраты). В-третьих, задача должна характеризоваться линейностью (например, если на изготовление детали требуются три часа, то на изготовление двух будет затрачено шесть часов, на выпуск трех – девять и т.д.) В-четвертых, задача должна характеризоваться однородностью (изделия, изготовленные на станке, идентичны; все часы, в течение которых рабочий выполняет ту или иную операцию, используются им с одинаковой продуктивностью). Пятое условие заключается в делимости: метод линейного программирования строится на допущении, что результат и ресурсы можно разделить на доли. Если такое деление невозможно, например, полет половины самолета или прием на работу одной четвертой служащего, аналитику лучше всего воспользоваться специальной модификацией линейного программирования – дискретным (целочисленным) программированием.

Методы линейного программирования могут применяться, если построена только одна цель: максимизировать (например, прибыль) или минимизировать (например, издержки). Когда целей несколько, используют целевое программирование. Если же задача эффективнее всего решается поэтапно или по временным интервалам, аналитику следует использовать метод динамического программирования. В еще более сложных задачах при решении могут понадобиться другие варианты данного метода, например нелинейное или квадратичное программирование.

Модель линейного программирования. Формально выражаясь, задача линейного программирования связана с оптимизацией процесса, в ходе которого отбираются неотрицательные искомые переменные X_1, X_2, \dots, X_n , используемые затем для максимизации (или минимизации) целевой функции в следующей форме.

Максимизировать (минимизировать) целевую функцию

$$F = C_1X_1 + C_2X_2 + \dots + C_nX_n \quad (1)$$

при условии ограничений на количество ресурсов, выраженных в таком виде:

$$\begin{cases} A_{11}X_1 + A_{12}X_2 + \dots + A_{1n}X_n \leq B_1 ; \\ A_{21}X_1 + A_{22}X_2 + \dots + A_{2n}X_n \leq B_2 ; \\ \dots \\ A_{m1}X_1 + A_{m2}X_2 + \dots + A_{mn}X_n \leq B_m , \end{cases} \quad (2)$$

где C_n, A_{mn}, B_m – заданные постоянные величины.

В зависимости от типа задачи ограничения могут указываться также с использованием знака «равенство» (=) или знака «больше или равно» (\geq).

Допустимым решением или планом задачи линейного программирования называется вектор $P = (X_1, X_2, \dots, X_n)$, удовлетворяющий условиям (2).

Множество планов в геометрической интерпретации представляет собой выпуклый многогранник.

План, который обращает в равенство хотя бы n независимых ограничений задачи, называется опорным планом. Опорный план соответствует некоторой вершине многогранника планов.

Опорный план содержит не свыше m положительных компонент (m – число независимых ограничений задачи). Опорный план, содержащий менее m положительных компонент, называют вырожденным.

Оптимальный план является опорным (если оптимум достигается на двух опорных планах, то оптимальны все планы, соответствующие отрезку, соединяющему соответствующие вершины).

Основным численным методом решения задач линейного программирования является симплексный метод.

Доказано, что оптимальное решение задачи линейного программирования связано с угловыми точками многоугольника решений, т.е. с опорными планами. Они определяются системой m линейно независимых векторов, содержащихся в системе из n векторов A_1, \dots, A_n . Количество опорных планов меньше C_n^m , где n – число неизвестных, а m – число ограничений. При больших n и m найти все их перебором очень трудно, поэтому необходим упорядоченный перебор, такой схемой является симплексный метод, который позволяет исходя из известного опорного плана задачи за конечное число шагов получить ее оптимальный план. Каждый из шагов (итераций) состоит в нахождении нового плана, которому соответствует меньшее для задачи на минимум (большее для задачи на максимум) значение линейной формы, чем ее значение в предыдущем плане. Процесс продолжается до получения оптимального плана. Если задача не обладает планами или ее линейная форма неограничена на многограннике решений, то симплексный метод позволяет установить это в процессе решения.

При использовании симплексного метода пользуются симплексными таблицами следующего вида [2]:

C_0	Базис плана	План X	C_1	C_2	...	C_m	C_{m+1}	...	C_k	...	C_n
			P_1	P_2	...	P_m	P_{m+1}	...	P_k	...	P_n
C_1	P_4	B_1	1	0	...	0	$Z_{1,m+1}$...	Z_{1k}	...	Z_{1n}
C_2	P_5	B_2	0	1	...	0	$Z_{2,m+1}$...	Z_{2k}	...	Z_{2n}
...
C_m	P_m	B_m	0	0	...	1	$Z_{m,m+1}$...	Z_{mk}	...	Z_{mn}
	Z_k	$F(X) = 0$	Z_1	Z_2	...	Z_m	Z_{m+1}	...	Z_k	...	Z_n
	Δ_k		Δ_1	Δ_2	...	Δ_m	Δ_{m+1}	...	Δ_k	...	Δ_n

В первом столбце таблицы C_0 записывают коэффициенты целевой функции (C_1, C_2, \dots, C_m) при базисных переменных (в базис входят только векторы, образующие единичную подматрицу, как в нашем случае P_1/P_m). Во втором столбце (Базис плана) должны находиться базисные векторы данного опорного плана, а в третьем столбце (План X) – правая часть ограничений задачи (базисные компоненты плана). Таким образом, перемножая элементы первого столбца таблицы со столбцом «План X » и суммируя эти произведения, мы получаем значение целевой функции (ячейка – $F(X) = C_1B_1 + C_2B_2 + \dots + C_mB_m$).

Первая строка симплексной таблицы содержит коэффициенты линейной функции нашей задачи и остается неизменной на протяжении всего решения (C_1, C_2, \dots, C_n).

В центральной части таблицы записывают коэффициенты при неизвестных в ограничениях исходной задачи ($Z_{11}, \dots, Z_{1n}, \dots, Z_{m1}, \dots, Z_{mn}$). При этом следует заметить, что коэффициенты при базисных переменных в ограничениях задачи составляют единичную подматрицу.

Строку Z_k рассчитывают по формуле

$$Z_k = \sum_{j=1}^m C_j Z_{jk} . \quad (3)$$

Строку Δ_k рассчитывают по формуле

$$\Delta_k = Z_k - C_k . \quad (4)$$

Известно, что при переходе к новому опорному плану $X(\Theta)$, в котором k -я компонента равна $\Theta > 0$ (станет базисной), значение целевой функции изменится и будет равно:

$$F(X(\Theta)) = F(X^0) - \Theta \Delta_k . \quad (5)$$

Выводы относительно данного опорного плана делают на основании содержимого последней строки таблицы по следующим критериям.

Критерий 1 (критерий оптимальности). Если все $\Delta_k \geq 0$, то выбранный план для задачи максимизации является оптимальным (для задачи на минимум признак оптимальности – неположительность всех Δ_k).

Критерий 2. Если обнаруживается некоторое $\Delta_k < 0$ (для задачи минимизации $\Delta_k > 0$) и хотя бы одно из значений $Z_{jk} > 0$, то переход к новому опорному плану увеличит (уменьшит) значение целевой функции. При этом в базис будет вводиться новый вектор, выбор которого определяется по критерию максимального по модулю произведения $\Theta \Delta_k$. Если к тому же учесть, что число положительных (базисных) компонент опорного плана должно оставаться равным m , то одну из m базисных компонент исходного плана обращаем в нуль выбором:

$$\Theta = \min \frac{X_j^0}{Z_{jk}} , \quad (6)$$

где X_j^0 – компоненты опорного плана; Z_{jk} – коэффициенты при k -й переменной в ограничениях задачи.

Критерий 3. Если обнаруживается некоторое $\Delta_k < 0$, но все $Z_{jk} \leq 0$, то линейная форма задачи не ограничена по максимуму (неограниченность по минимуму устанавливается аналогично при $\Delta_k > 0$ и всех $Z_{jk} \leq 0$).

Переход к очередной симплексной таблице сводится к тому, чтобы выразить X_k из уравнения, соответствующего минимуму для Θ , и исключить из остальных уравнений (в итоге получаем единичный вектор при новой базисной компоненте и тем самым сохраняется единичная подматрица при базисных векторах):

1. Строку, соответствующую вектору, выводимому из базиса (главная строка), делим на коэффициент (главный элемент), находящийся на пересечении этой строки и столбца, соответствующего вектору, вводимому в базис (главный столбец).

2. От всех остальных строк вычитаем преобразованную главную строку, умноженную на элемент, находящийся на пересечении искомой строки и главного столбца.

Получив новую симплексную таблицу для улучшенного опорного плана, действуем по той же схеме, т.е. проверяем новый план на оптимальность и при необходимости переходим к очередному опорному плану. Этот процесс продолжаем до тех пор, пока найденный опорный план не окажется оптимальным.

Итак, нахождение оптимального плана симплексным методом включает следующие этапы:

1. Находят опорный план.
2. Составляют симплекс-таблицу.

3. Выясняют, имеется ли хотя бы одно отрицательное число Δ_j . Если нет, то найденный опорный план оптимален. Если же среди чисел Δ_j имеются отрицательные, то либо устанавливают неразрешимость задачи, либо переходят к новому опорному плану.

4. Находят направляющие столбец и строку. Направляющий столбец определяется наибольшим по абсолютной величине отрицательным числом Δ_j , а направляющая строка – минимальным из отношений компонент столбца вектора X к положительным компонентам направляющего столбца.

5. Определяют положительные компоненты нового опорного плана, коэффициенты разложения векторов P_j по векторам нового базиса и числа F'_0, Δ'_j . Все эти числа записываются в новой симплекс-таблице.

6. Проверяют найденный опорный план на оптимальность. Если план не оптимален и необходимо перейти к новому опорному плану, то возвращаются к этапу 4, а в случае получения оптимального плана или установления неразрешимости процесс решения задачи заканчивают.

Последовательность выполнения работы

1. Математическая постановка задачи.
2. Численное решение задачи с использованием функции «Поиск решения» пакета анализа данных.
3. Оформление отчета.

Содержание отчета

1. Название и цель лабораторной работы.
2. Постановка задачи линейного программирования.
3. Последовательность операций решения задачи с использованием программы Excel.
4. Результаты решения задачи.

Лабораторная работа 11

НЕЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ

Цель работы: Освоение методов автоматизированного решения задач нелинейного программирования с использованием пакетов прикладных программ.

Методические указания

Задачами нелинейного программирования называются задачи, в которых нелинейны и (или) целевая функция, и (или) ограничения в виде равенств и неравенств и для которых методы математического анализа оказываются непригодными.

Класс задач нелинейного программирования шире класса задач линейного программирования. Подробное изучение практических задач, которые условились считать линейными, показывает, что они в действительности являются нелинейными.

Нелинейное программирование представляет собой наиболее характерный метод оптимизации при проектировании машин и технологических процессов и служит для выбора наилучшего плана распределения ограниченных материальных, финансовых и трудовых ресурсов.

В евклидовом пространстве система ограничений определяет область допустимых решений задачи. В отличие от задач линейного программирования она не всегда является выпуклой. Если определена область допустимых решений, то нахождение решения задачи (1), (2) сводится к определению такой точки этой области, через которую проходит гиперповерхность наивысшего (наинизшего) уровня $f(x_1, x_2, \dots, x_n) = h$. Указанная точка может находиться как на границе области допустимых решений, так и внутри нее.

Метод множителей Лагранжа. Рассмотрим частный случай общей задачи нелинейного программирования, предполагая, что система ограничений содержит только уравнения, отсутствуют условия неотрицательности переменных, $f(x_1, x_2, \dots, x_n)$ и $g_i(x_1, x_2, \dots, x_n)$ – функции, непрерывные вместе со своими частными производными. Ограничения в задаче заданы уравнениями, поэтому для ее решения можно воспользоваться классическим методом отыскания условного экстремума функций нескольких переменных. Вводят набор переменных, называемых множителями Лагранжа, и составляют функцию Лагранжа:

$$F(x_1, x_2, \dots, x_n) = f(x_1, x_2, \dots, x_n) + \sum_{i=1}^m \lambda_i [b_i - g_i(x_1, x_2, \dots, x_n)]. \quad (7)$$

Находят частные производные

$$\frac{\partial F}{\partial x_j} \quad (j = 1, \bar{n}), \quad \frac{\partial F}{\partial \lambda_i} \quad (i = 1, \bar{m}) \quad (8)$$

и рассматривают систему $n + m$ уравнений

$$\begin{cases} \frac{\partial F}{\partial x_j} = \frac{\partial f}{\partial x_j} - \sum_{i=1}^m \lambda_i \frac{\partial g_i}{\partial x_j} = 0, & j = 1, \bar{n}; \\ \frac{\partial F}{\partial \lambda_j} = b_j - g_j(x_1, x_2, \dots, x_n) = 0, & i = 1, \bar{m}. \end{cases} \quad (9)$$

с $n + m$ неизвестными $x_1, x_2, \dots, x_n, \lambda_1, \lambda_2, \dots, \lambda_m$. Решив систему уравнений (9), получают все точки, в которых функция может иметь экстремальные значения. Дальнейшее исследование найденных точек проводят так же, как и в случае безусловного экстремума. Метод множителей Лагранжа имеет ограниченное применение, так как система (9), как правило, имеет несколько решений.

Выпуклое программирование. Функция $f(x_1, x_2, \dots, x_n)$, заданная на выпуклом множестве X , называется выпуклой, если для любых двух точек $X^{(1)}$ и $X^{(2)}$ из X и любого $0 \leq \lambda \leq 1$ выполняется соотношение:

$$f[\lambda X^{(2)} + (1-\lambda)X^{(1)}] \leq \lambda f(X^{(2)}) + (1-\lambda)f(X^{(1)}). \quad (10)$$

Функция $f(x_1, x_2, \dots, x_n)$, заданная на выпуклом множестве X , называется вогнутой, если для любых двух точек $X^{(1)}$ и $X^{(2)}$ из X и любого $0 \leq \lambda \leq 1$ выполняется соотношение:

$$f[\lambda X^{(2)} + (1-\lambda)X^{(1)}] \geq \lambda f(X^{(2)}) + (1-\lambda)f(X^{(1)}). \quad (11)$$

Если неравенства (10) и (11) считать строгими и они выполняются при $0 \leq \lambda \leq 1$, то функция $f(x_1, x_2, \dots, x_n)$ является строго выпуклой (строго вогнутой). Выпуклость и вогнутость функций определяется только относительно выпуклых множеств.

Если $f(x) = \sum_{j=1}^k f_j(x)$, где $f_j(x)$ – выпуклые (вогнутые) функции на некотором выпуклом множестве $X \subset E$, то функция $f(x)$ – также выпуклая (вогнутая) на X .

Рассмотрим задачу нелинейного программирования:

$$f(x_1, x_2, \dots, x_n) \rightarrow \max \quad (12)$$

при ограничениях

$$g_j(x_1, x_2, \dots, x_n) \leq b_j, \quad i = 1, \bar{m}, \quad (13)$$

$$x_j \geq 0, \quad j = 1, \bar{n}. \quad (14)$$

Для решения сформулированной задачи в такой общей постановке не существует универсальных методов. Однако для отдельных классов задач, в которых сделаны дополнительные ограничения относительно свойств функций $f(x)$ и $g(x)$, разработаны эффективные методы их решения.

Говорят, что множество допустимых решений задачи (12) – (14) удовлетворяет условию регулярности, или условию Слейтера, если существует, по крайней мере, одна точка $X^{(0)}$, принадлежащая области допустимых решений такая, что $g_j(X^{(0)}) < b_j, \quad i = 1, \bar{m}$. Задача (12) – (14) называется задачей выпуклого программирования, если функция $f(x_1, x_2, \dots, x_n)$ является вогнутой (выпуклой), а функции $g_j(x_1, x_2, \dots, x_n) (i = 1, \bar{m})$ – выпуклыми. Функцией Лагранжа задачи выпуклого программирования (12) – (14) называется функция

$$L(x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_m) = f(x_1, x_2, \dots, x_n) + \sum_{i=1}^m [b_i - g_i(x_1, x_2, \dots, x_n)] y_i, \quad (15)$$

где y_1, y_2, \dots, y_m – множители Лагранжа.

Точка $(X^{(0)}, Y^{(0)}) = (x_1^0, x_2^0, \dots, x_n^0, y_1^0, y_2^0, \dots, y_m^0)$ называется седловой точкой функции Лагранжа, если $L(x_1, x_2, \dots, x_n, y_1^0, y_2^0, \dots, y_m^0) \leq L(x_1^0, x_2^0, \dots, x_n^0, y_1^0, y_2^0, \dots, y_m^0) \leq L(x_1^0, x_2^0, \dots, x_n^0, y_1, y_2, \dots, y_m)$ для всех $x_j \geq 0 (j = 1, \bar{n})$ и $y_j \geq 0 (j = 1, \bar{m})$.

Теорема Куна–Таккера. Для задачи выпуклого программирования (12) – (14) множество допустимых решений которой обладает свойством регулярности, $X^{(0)} = (x_1^0, x_2^0, \dots, x_n^0)$ является оптимальным решением тогда и только тогда, когда существует такой вектор $Y^{(0)} = (y_1^0, y_2^0, \dots, y_m^0), y_i^0 \geq 0, i = 1, \bar{m}$, что $(X^{(0)}, Y^{(0)})$ – седловая точка функции Лагранжа.

Если предположить, что функции f и g непрерывно дифференцируемы, то теорема Куна–Таккера может быть дополнена аналитическими выражениями, определяющими необходимые и достаточные условия того, чтобы точка $(X^{(0)}, Y^{(0)})$ была седловой точкой функции Лагранжа, т.е. являлась решением задачи выпуклого программирования:

$$\left\{ \begin{array}{l} \frac{\partial L_0}{\partial x_j} \leq 0, \quad j = 1, \bar{n}; \\ x_j^0 \frac{\partial L_0}{\partial x_j} = 0, \quad j = 1, \bar{n}; \\ x_j^0 \geq 0, \quad j = 1, \bar{n}; \\ \frac{\partial L_0}{\partial y_i} \geq 0, \quad i = 1, \bar{m}; \\ y_i \frac{\partial L_0}{\partial y_i} = 0, \quad i = 1, \bar{m}; \\ y_i^0 \geq 0, \quad i = 1, \bar{m}, \end{array} \right. \quad (16)$$

где $\frac{\partial L_0}{\partial x_j}$ и $\frac{\partial L_0}{\partial y_i}$ – значения соответствующих частных производных функции Лагранжа, вычисленные в седловой точке.

Квадратичное программирование. Частным случаем задачи нелинейного программирования является задача квадратичного программирования, в которой ограничения:

$$g_j(x_1, x_2, \dots, x_n) = b_j - \sum_{i=1}^n a_{ij} x_i, \quad i = 1, \bar{m} \quad (17)$$

являются линейными, а функция $f(x_1, x_2, \dots, x_n)$ представляет собой сумму линейной и квадратичной функции (квадратичной формы):

$$f(x_1, x_2, \dots, x_n) = c_1 x_1 + c_2 x_2 + \dots + c_n x_n + d_{11} x_1^2 + d_{22} x_2^2 + \dots + d_{nn} x_n^2 + 2d_{12} x_1 x_2 + 2d_{13} x_1 x_3 + \dots + 2d_{n-1} x_{n-1} x_n. \quad (18)$$

Как и в общем случае решения задач нелинейного программирования, для определения глобального экстремума задачи квадратичного программирования не существует эффективного вычислительного метода, если не известно, что любой локальный экстремум является одновременно и глобальным. Так как в задаче квадратичного программирования множество допустимых решений выпукло, то, если целевая функция вогнута, любой локальный максимум является глобальным; если же целевая функция – выпуклая, то любой локальный минимум также и глобальный.

Функция f представляет собой сумму линейной функции (которая является и выпуклой, и вогнутой) и квадратичной формы. Если последняя является вогнутой (выпуклой), то задачи отыскания максимума (минимума) целевой функции могут быть успешно решены. Вопрос о том, будет ли квадратичная форма вогнутой или выпуклой, зависит от того, является ли она отрицательно-определенной, отрицательно-полуопределенной, положительно-определенной, положительно-полуопределенной или вообще неопределенной.

Последовательность выполнения работы

1. Математическая постановка задачи (целевая функция, ограничения).
2. Запись выражения для функции Лагранжа и условий Куна–Таккера.
3. Численное решение задачи с использованием функции «Поиск решения» пакета анализа данных.
4. Оформление отчета.

Содержание отчета

1. Название и цель лабораторной работы.
2. Постановка задачи нелинейного программирования.
3. Последовательность операций решения задачи с использованием программы Excel.
4. Результаты решения задачи.

СПИСОК ЛИТЕРАТУРЫ

К лабораторным работам 1 – 9

1. Маклаков, С.В. Моделирование бизнес-процессов с BPwin 4.0 / С.В. Маклаков. – М. : ДИАЛОГ-МИФИ, 2002. – 224 с.
2. Маклаков, С.В. BPwin и ERwin. CASE-средства разработки информационных систем / С.В. Маклаков. – 2-е изд., испр. и доп. – М. : ДИАЛОГ-МИФИ, 2001. – 304 с.
3. Маклаков, С.В. Создание информационных систем AllFusion Modeling Suite / С.В. Маклаков. – М. : ДИАЛОГ-МИФИ, 2003. – 432 с.
4. Черемных, С.В. Структурный анализ систем: IDEF-технологии / С.В. Черемных, И.О. Семенов, В.С. Ручкин. – М. : Финансы и статистика, 2001. – 208 с.
5. Федотова, Д.Э. CASE-технологии : практикум / Д.Э. Федотова, Ю.Д. Семенов, К.Н. Чижик. – М. : Горячая линия-Телеком, 2003. – 160 с.
6. Калянов, Г.Н. CASE-технологии. Консалтинг в автоматизации бизнес-процессов / Г.Н. Калянов. – 3-е изд. – М. : Горячая линия-Телеком, 2002. – 320 с.

К лабораторным работам 10 – 11

1. Акулич, И.Л. Математическое программирование в примерах и задачах : учеб. пособие для студентов экон. спец. вузов / И.Л. Акулич. – М. : Высш. шк., 1986. – 319 с.
2. Таха, Хэмди А. Введение в исследование операций / А. Таха Хэмди. – М. : Издательский дом «Вильямс», 2001. – 912 с.
3. Гельман, В.Я. Решение математических задач средствами Excel / В.Я. Гельман. – СПб. : Питер, 2003. – 240 с.
4. Долголаптева, В. Работа в Excel 7.0 для Windows 95 на примерах / В. Долголаптева. – М. : Бином, 1997. – 384 с.
5. Решение задач оптимизации в Excel // www.allmath.ru/appliedmath/operations/excel/excel.htm. 22.12.2006.
6. Курейчик, В.М. Математическое обеспечение конструкторского и технологического проектирования с применением САПР : учебник для вузов / В.М. Курейчик. – М. : Радио и связь, 1990. – 352 с.
7. Экономико-математические методы // www.math.mrsu.ru/programs/ivt/e-learn/index.htm. 24.12.2006.
8. Саймон, Джинджер. Анализ данных в Excel : наглядный курс создания отчетов и сводных таблиц : пер. с англ. / Джинджер Саймон. – М. : Издательский дом «Вильямс», 2004. – 528 с.

СОДЕРЖАНИЕ

Лабораторная работа 1	ЗНАКОМСТВО С IDEF-ТЕХНОЛОГИЯМИ И СРЕДОЙ VRWIN	3
Лабораторная работа 2	ПОСТРОЕНИЕ КОНТЕКСТНОЙ ДИАГРАММЫ	4
Лабораторная работа 3	ПОСТРОЕНИЕ ДИАГРАММЫ ДЕКОМПОЗИЦИИ ПЕРВОГО УРОВНЯ ...	6
Лабораторная работа 4	ПОСТРОЕНИЕ ДИАГРАММЫ ДЕКОМПОЗИЦИИ ВТОРОГО УРОВНЯ ...	7
Лабораторная работа 5	ПОСТРОЕНИЕ ДИАГРАММЫ УЗЛОВ	9
Лабораторная работа 6	ПОСТРОЕНИЕ FEO-ДИАГРАММЫ	9
Лабораторная работа 7	ПОСТРОЕНИЕ ДИАГРАММЫ В СТАНДАРТЕ IDEF3	10
Лабораторная работа 8	ПОСТРОЕНИЕ ДИАГРАММЫ ПОТОКОВ ДАННЫХ DFD	11
Лабораторная работа 9	СТОИМОСТНОЙ АНАЛИЗ	12
Лабораторная работа 10	ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ	13
Лабораторная работа 11	НЕЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ ...	18
Список литературы		23
Приложения		24