МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий

(наименование института полностью)

Кафедра « Прикладная математика и информатика»

(наименование кафедры)

01.03.02 Прикладная математика и информатика

(код и наименование направления подготовки, специальности)

Системное программирование и компьютерные технологии

(направленность (профиль)/специализация)

БАКАЛАВРСКАЯ РАБОТА

на тему Динамическое программирование в решении производственных задач

Студент	А.А. Марков	
	(И.О. Фамилия)	(личная подпись)
Руководитель	Н.А. Сосина	
	(И.О. Фамилия)	(личная подпись)
Консультанты	Н.В. Андрюхина	
	(И.О. Фамилия)	(личная подпись)

Допустить к защите

Заведующий кафедрой к.т.н., доцент, А.В. Очеповский _____

(ученая степень, звание, И.О. Фамилия) (личная подпись)

«_____»_____20_____г.

Тольятти 2019

АННОТАЦИЯ

Тема выпускной квалификационной работы: «Динамическое программирование в решении производственных задач»

Работа выполнена студентом Тольяттинского Государственного Университета, института математики, физики и информационных технологий, группы ПМИб-1502, Марковым Александром Александровичем.

Выпускная квалификационная работа посвящена реализации алгоритмов решения производственных задач методом динамического программирования.

Целью выпускной квалификационной работы является обоснование метода динамического программирования для решения производственных задач, решение задач аналитически и с помощью программных средств.

Объект исследования – производственные задачи,решаемые методом динамического программирования.

Предмет исследования – алгоритм метода динамического программирования.

Задачи работы:

1) изучить общий подход динамического программирования;

2) выявить производственные задачи, решаемые методом динамического программирования;

3) продемонстрировать применение метода динамического программирования при решении производственных задач аналитически;

4) выполнить программную реализацию разработанного алгоритма.

Актуальность, выполненной работы можно обосновать тем, чторешение ряда задач производственного управления можно упростить, если процесс управления осуществлять поэтапно, заменивнахождение точек экстремума целевой функции многих переменных многократным нахождением точек экстремума функции одного или небольшого числа

переменных, что возможно, если воспользоваться методом динамического программирования.

Результатом работы является программа решения производственных задач, таких как задачи о замене оборудования, задачи о распределении ресурсов и инвестиций и другие.

Бакалаврская работа содержит пояснительную записку объемом 43 страниц, включая 9 иллюстраций, 12 таблиц, список литературы.

ABSTRACT

The title of the graduation work is "Dynamic programming in solving production problems"

The final qualification paper is devoted to the implementation of the algorithm for solving the problem of optimal resource allocation.

The aim of this work is to substantiate the dynamic programming method for solving production problems, such as: resource allocation problem, investment distribution problem, equipment replacement task, knapsackproblem analytically and using software tools.

The object of study - production problems.

The subject of research is an algorithm for solving production problems.

In the first part the theoretical basis of the dynamic programming method, Bellman's principle of optimality and the main recurrence relations are discussed. The application of the method of dynamic programming in applied problems is also considered.

In the second part, we will consider the types of production tasks and give appropriate attention to the analytical solution of each task.

The third part is about the design and implementation of the algorithm and interface for a program.

The result of the work is a program for solving production problems.

The bachelor's work contains an explanatory note with a volume of 43 pages, including 9 illustrations, 12 tables, a list of references.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Большое количество производственных задач можно сформулировать в виде математической задачи на экстремум. К некоторым из этих задач применимы классические методы математического анализа и вариационное исчисление. Но для значительной части задач производства и управления подобные методы не применимы или малоэффективны. Нахождение экстремума классическими методами зачастую приводит к тому, что на следующем этапе решения приходится решать новые задачи, еще более сложные, чем поставленные первоначально. Даже в тех случаях, когда задача формулируется, как задача математического программирования, применение необходимых условий для построения оптимального решения очень часто не приводит к нужному результату. В производственных задачах оптимизации к максимуму целевой функции легче бывает подойти поэтапно, чем аналитически решить систему уравнений, определяемую возможными ограничениями поставленной задачи. И даже, в том случае, когда задача математического программирования может быть решена, проверка найденного решения может оказаться довольно сложной и тем сложнее, чем больше аргументов у функции.

Актуальностьвыполненной работы можно обосновать тем, чторешениеряда задач производственного управления можно упростить, если процесс управления осуществлять поэтапно, заменивнахождение точек экстремума целевой функции многих переменных многократным нахождением точек экстремума функции одного или небольшого числа переменных, что возможно, если воспользоваться методом динамического программирования.

Объект исследования – производственные задачи, решаемые методом динамического программирования.

Предмет исследования – алгоритм метода динамического программирования.

Целью выпускной квалификационной работы является обоснование метода динамического программирования для решения производственных задач, решение задач аналитически и с помощью программных средств.

Для достижения поставленной цели необходимо решить следующие задачи:

1) изучить общий подход динамического программирования;

2) выявить производственные задачи, решаемые методом динамического программирования;

3) продемонстрировать применение метода динамического программирования при решении производственных задач аналитически;

4) выполнить программную реализацию разработанного алгоритма.

Выпускная квалификационная работа состоит из введения, трёх глав, заключения, списка используемых источников.

В главе 1 рассматривается общая постановка задачи динамического программирования.

В главе 2 приводятся производственные задачи, решаемые методом динамического программирования.

В главе 3 разрабатываются алгоритм программы. В заключении представлены результаты и выводы о выполненной работе.

ГЛАВА 1 МЕТОД ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

1.1 Общая постановка задачи динамического программирования

Основоположником динамического программирования можно считать нашего соотечественника, русского математика А. А. Маркова(начало XX века). Его исследовательские работы были продолжены в 1940-х годах американским математиком А. Вальдом, одним из основателей так называемого исследовательского анализа.

Однако наиболее полно и систематизировано сформулировать основные принципы оптимального управления многошаговыми процессами впервые удалось американскому математику Р. Беллману (Беллман Ричард Эрнест (1920– 1984), основные труды которого посвящены вычислительной математике и теории оптимального управления).Задачи управления запасами были первыми задачами, которые решались этим методом.

Динамическое программирование является разделом математической науки, в основе которой лежит изучение экстремальных задач управления (методы вычислительной математики, которые применяются для поиска экстремумов функций), планирование и разработка методов их решения, в котором процесс принятия решения разбивается на отдельные этапы.

В общей постановке задачу динамического программирования можно сформулировать следующим образом. Управляемая физическая системаS, характеризуется определенным набором параметров. Требуется построить оптимальное управлениеu^*(на множестве допустимых управлений), переводящее систему из начального состоянияx_0 в конечное состояние x_N, обеспечив целевой функции (показателю эффективности управления) экстремум.

Алгоритмы динамического программирования используются для поиска решения не сразу для всей сложной задачи, а для поиска

оптимального решения для нескольких более простых задач аналогичного содержания, на которые распадается исходная задача. Также данные алгоритмы могут применятся для подсчета количества этих решений.

Т.е. в динамическом программировании задача разделяется на подзадачи, и решения этих подзадач объединяются вместе для достижения общего решения основной задачи.

Если подзадачи могут быть рекурсивно вложены в более крупные задачи, тогда методы динамического программирования применимы, и существует связь между значением более крупной проблемы и значениями подзадач. В литературе по оптимизации это соотношение называется уравнением Беллмана.

При использовании подходов, таких как «разделяй и властвуй», подзадача может быть решена несколько раз. Динамическое программирование решает каждую из этих подзадач только один раз, тем самым уменьшив количество вычислений, а затем сохраняет результат, избегая повторного вычисления на более позднем этапе, когда требуется решение для этой подзадачи.

Динамическое программирование используется для решения задач оптимизации (например, поиск кратчайшего пути), где может существовать множество решений, но интерес представляет только поиск оптимального решения для одного и того же.

Эти проблемы должны иметь свойства перекрывающихся подзадач. Иными словами, решаемая задача может быть разбита на подзадачи, которые многократно используются, причем рекурсивный алгоритм решает одну и ту же подзадачу много раз, а не создает новую подзадачу. Например, числа Фибоначчи и оптимальная подструктура. В конечном итоге, оптимальное решение может быть построено из оптимальных решений подзадач.

Следует отметить, что во многих случаях алгоритмы перебора могут работать быстрее, чем алгоритмы динамического программирования, но они не гарантируют оптимальное решение задачи.

1.2 Уравнение Беллмана

Рассмотрим следующую постановку задачи

$$V(x) = \sup_{y} F(x, y) + \beta V(y), \tag{1.1}$$

где $y \in U(x)$ –уравнение функции (1.1) называется уравнением Беллмана;

x – переменная состояния.

$G(x) = \{y \in U(x): V(x) = F(x, y) + \beta V(y)\}$ – оптимальная политика. В ней представляются все значения y, которые достигают максимума в выражении (1). Если $G(x)$ – единственное значение(уникальный оптимум), то G называется функцией политики.

Уравнение Беллмана было впервые применено к теории управления техническими системами и другим темам прикладной математики, а затем стало важным инструментом экономической теории.

Соответствующим уравнением Беллмана могут быть решены ряд проблем, решающиеся с помощью использования теории оптимального уравнения. Использующийся термин" Уравнение Беллмана" относят к уравнению динамического программирования, связанному с задачей оптимизации в дискретном времени. Уравнением в частных производных является уравнение Беллмана в задачах непрерывной оптимизации, которое имеет название Гамильтона-Якоби-Беллмана.

Чтобы понять уравнение Беллмана, необходимо понять несколько основных понятий. Во-первых, любая задача оптимизации имеет какую-то цель: минимизация времени в пути, минимизация затрат, максимизация

прибыли, максимизация полезности и т. Д. Математическая функция, которая описывает эту задачу, называется целевой функцией.

Динамическое программирование разбивает задачу многопериодного планирования на более простые этапы в разные моменты времени. Следовательно, необходимо отслеживать, как ситуация с решениями меняется со временем. Информация о текущей ситуации, которая необходима для принятия правильного решения, называется «состоянием». Например, чтобы решить, сколько потреблять и тратить в каждый момент времени, люди должны знать (среди прочего) свое первоначальное богатство. Следовательно, богатство будет одной из их переменного состояния, но, вероятно, будут и другие.

Переменные, выбранные в любой данный момент времени, часто называют контрольными переменными. Например, учитывая их текущее состояние, люди могут решить, сколько потреблять сейчас. Выбор управляющих переменных теперь может быть эквивалентен выбору следующего состояния; в более общем случае на следующее состояние влияют другие факторы, помимо текущего контроля. Например, в простейшем случае сегодняшнее богатство (государство) и потребление (контроль) могут точно определять завтрашнее богатство (новое государство), хотя, как правило, другие факторы также влияют на завтрашнее богатство.

Подход динамического программирования описывает оптимальный план, находя правило, которое сообщает, какими должны быть элементы управления, учитывая любое возможное значение состояния. Например, если потребление (c) зависит только от богатства (W), мы бы искали правило это дает потребление как функцию богатства. Такое правило, определяющее элементы управления как функцию государств, называется функцией политики

Наконец, по определению, оптимальным правилом принятия решения является то, которое достигает наилучшего возможного значения цели.

Например, если кто-то выбирает потребление, учитывая богатство, чтобы максимизировать счастье (при условии, что счастье H может быть представлено математической функцией, такой как функция полезности, и определяется чем-то богатством), то каждый уровень богатства будет связан с какой-то наивысший возможный уровень счастья, Наилучшее возможное значение цели, записанное как функция состояния, называется функцией значения.

Ричард Беллман показал, что задача динамической оптимизации в дискретном времени может быть сформулирована в рекурсивной пошаговой форме, известной как обратная индукция, записав взаимосвязь между функцией значения в одном периоде и функцией значения в следующем периоде. Соотношение между этими двумя ценностными функциями и называется «уравнением Беллмана». В этом подходе оптимальная политика в последний период времени заранее указывается как функция значения переменной состояния в то время, и итоговое оптимальное значение целевой функции, таким образом, выражается через это значение переменной состояния.

К особенностям математической модели динамического программирования относятся:

- задача оптимизации формулируется как итоговый многошаговый процесс управления;
- целевая функция, является аддитивной и равна сумме целевых функций каждого шага оптимизации;
- выбор управления на каждом шаге зависит только от состояния самой системы на этом шаге и не влияет на предшествующие шаги (отсутствие обратной связи);
- состояние системы S_k после каждого шага управления зависит исключительно от предшествующего состояния системы и этого

управляющего воздействия x_k (нет последействия), причем оно может быть записано в виде уравнения состояния системы;

- на каждом шаге управление x_k зависит от конечного числа управляющих переменных, а состояния системы S_k- от конечного числа параметров;
- оптимальное управление представляет собой вектор X, который определяется как последовательность оптимальных пошаговых управлений, число которых и определяет количество шагов задачи.

Основные свойства задач, в которых можно применять метод динамического программирования:

- задача должна допускать интерпретацию как многошаговый процесс принятия решения;
- задача должна быть определена для любого числа шагов и иметь структуру, не зависящую от их числа;
- при рассмотрении задачи на каждом шаге должно быть задано множество параметров, описывающих состояние системы;
- выбор решения(управления) на каждом шаге не должен оказывать влияния на предыдущие решения.

Вывод по главе 1

В первой главе выпускной квалификационной работы описана общая постановка задачи динамического программирования.

Приводится уравнение Беллмана, как основное соотношение для решения оптимизационных задач методом динамического программирования.

ГЛАВА 2 ПРОИЗВОДСТВЕННЫЕ ЗАДАЧИ, РЕШАЕМЫЕ МЕТОДОМ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

2.1ВИДЫ ПРОИЗВОДСТВЕННЫХ ЗАДАЧ

Особенность производственных задач, решаемых методом ДП, заключается в том, что процесс, протекающий в системе, зависит либо от времени (от этапов), либо имеет многоступенчатую структуру. Метод решения задач ДП состоит в том, что оптимальное управление строится поэтапно шаг за шагом. На каждом этапе оптимизируется только один шаг, но при этом учитывается изменение всего процесса, так как управление, оптимизирующее целевую функцию только для данного шага, может привести к неоптимальному эффекту всего процесса. Управление на каждом шаге должно быть оптимальным с точки зрения процесса в целом.

В основе вычислительных алгоритмов динамического программирования лежит следующий принцип оптимальности,сформулированныйР. Беллманом: каково бы не было состояние системы S в результате $k-1$ шагов, управление на k-ом шаге должно выбираться так, чтобы оно в совокупности с управлениями на всех последующих шагах с $(k+1)-\text{го}$ до N-го включительно доставляло экстремум целевой функции.

Принцип оптимальности Беллмана можно записать в виде функционального уравнения

$$F_k(x_{k-1},u_k)=\underset{u_k}{\operatorname{extr}}\big(z_k(x_{k-1},u_k)+F_{k+1}(x_k)\big), \qquad (2.1)$$

гдеx_{k-1} –множество состояний, в которых система S может находиться перед k-м шагом;

u_k - множество управлений, которые могут быть выбраны на k-м шаге для перевода системы Sв одно из состояний множестваx_k;

$z_k(x_{k-1}, u_k)$ значение целевой функции на k-м шаге;

$F_k(x_{k-1}, u_k)$ условно-оптимальное значение целевой функции на интервале от k-го до N-го шага включительно.

На основании уравнения (2.1) и с учетом множеств состояний системыx_{k-1}, x_k и множеств управленийu_k, которые могут быть выбраны на$\boldsymbol{k}$-м шаге, вычислительная процедура метода ДП распадается на два этапа: условную и безусловную оптимизацию.

Условная оптимизация осуществляется поэтапно при движении из конечного состоянияx_nсистемы S в первоначальное состояние x_0путем построения на каждом этапе условно-оптимального управления и нахождения условно-оптимального значения функции цели для каждого шага . Безусловная оптимизация осуществляется в обратном направлении: от первого шага к последнему, в результате чего находятся уже оптимальные управления на каждом шаге с точки зрения всего процесса. На втором этапе отрабатываются рекомендации, полученные на первом этапе.

Много интересных многошаговых процессов принятия решений возникает при управлении производственными процессами. Рассмотрим некоторые из них.

Задача о замене оборудования. В настоящее время промышленные предприятия производят замену оборудования в сроки, диктуемые не на основе интуиции, а на основе математических расчетов. Управленцы промышленных предприятий должны владеть методами составления плана замены машинного оборудования, в целях оптимизации его использования. Задачу по замене оборудования можно рассматривать как многоэтапный процесс, к которому применимы методы динамического программирования. Применение методов динамического программирования позволяет максимизировать прибыль или минимизировать затраты.

Задача оптимального распределения ресурсов. В производственной практике очень часто возникают задачи на оптимальное распределение ресурсов между предприятиями или внутри предприятия. Кроме того, к задаче оптимального распределения ресурсов можно отнести еще ряд задач. Например, задачу о размещение по торговым и складским помещениям какого-либо товара, задачу о распределении средств между различными отраслями промышленности и т.п.

Задача о загрузке транспортного средства.В транспортное средство требуется погрузить несколько видов груза, так, чтобы результат загрузки был эффективным. Например, максимизирует стоимость груза, размещенного в транспортном средстве, известна грузоподъемность транспортного средства, вес единицы груза и соответствующая эффективность.

Эту задачу называют еще «задачей о рюкзаке». Предполагается, что турист собирается в поход и оптимизирует вес рюкзака и ценность вложенного груза.

Задача о наикратчайшем пути через сети.Рассматривается сеть, состоящая из N узлов, занумерованных $1,2,\dots,N$, ивзаимосвязанных звеньев. Необходимо определить путь через сеть, который соединяет два заданных узла, полное время прохождения по которому по крайней мере не больше, чем для любого другого пути, соединяющего заданные узлы. При выборе маршрутов транспортных средств по транспортным сетям задача имеет огромное значение.

Задача планирования производственной линии.Ведется последовательная обработка Nразличных деталей на двух машинах. Известно время обработки каждой из Nдеталей. Требуется найти порядок обработки, минимизирующий время простоя каждой из машин и тем самым сокращающий общее время обработки деталей (задача Джонсона).

Выше перечисленные задачи не составляют полный список производственных задач, решаемых динамическим программированием.Аппарат динамического программирования используется и в численных решениях классических функциональных уравнений, обыкновенных дифференциальных уравнений и дифференциальных уравнений с частными производными. Кроме того, компьютерные технологии позволяют значительно расширить спектр задач, решаемых поэтапно, с применением метода динамического программирования в том числе и на производстве.

В разделе рассматриваются решения ряда производственных задач методом динамического программирования аналитически:

2.2 Задача о замене оборудования

Следует найти оптимальный план замены оборудования на 6-летний период, если известны производительность оборудования $r(t)$ и остаточная стоимость оборудования $S(t)$ в зависимости от возраста, стоимость нового оборудования P (заданы в таблицах). Возраст оборудования к началу эксплуатации равен 1 год.

Таблица 2.1 - исходные данные

t	0	1	2	3	4	5	6
$r(t)$	11	11	7	6	6	5	4
$S(t)$	10	8	7	7	5	5	3

P = 11 - цена нового оборудования

Составим алгоритм решения задачи о замене оборудования, состоящий из трех шагов:

1. Задать предполагаемый срок эксплуатации;

2. Определить входные параметры ($r(t)$ - доход от эксплуатации в течение одного года оборудования возраста t лет и $S(t)$ - остаточную стоимость оборудования);

3. Задать цену нового оборудования (*P).*

На первом шаге, где $k = 6$ и возможные состояния системы $t = 1, 2, \dots, 6$, расчет будем вести по формуле:

Полученные значения: $F_6(1) = 11, F_6(2) = 7, F_6(3) = 7, F_6(4) = 6, F_6(5) = 5, F_6(6) = 4.$

Составим матрицу значений функции Беллмана(таблица 2.2).

Таблица 2.2- матрица функции Беллмана

k	t					
	1	2	3	4	5	6
1	46					
2	39	31	30			
3	32	25	24			
4	25	19	18	16	13	
5	18	13	12	11	9	
6	11	7	6	6	5	4

Далее составим матрицу логических значений(Сохранить/Заменить), следуя правилу:

ЕСЛИ >= то значение = «Сохранить», иначе, значение «Заменить».

Результат представим в таблице 2.3:

Таблица 2.3- матрица логических значений

k	t					
	1	2	3	4	5	6
1	C					

2	С	3				
3	С	С	3			
4	С	С	3	3		
5	С	С	С	3	3	
6	С	С	С	С	С	С

Ответ: максимально возможная прибыль при эксплуатации оборудования достигается, если замену оборудования провести в начале третьего года его эксплуатации.

2.3 Задачи распределения инвестиций

Следует определить оптимальный план расширения производства трех компаний, если известна их прибыль в год при отсутствии вложений и при инвестировании 1, 2, 3 или 4 млн. Определить, при каком инвестировании будет максимальный процент прироста прибыли.

Таблица 2.4 – исходные данные

f1	f2	f3	x_i
30	50	45	0
70	100	90	1
365	370	260	2
430	465	615	3
600	750	710	4

Первый этап решения задачи распределения инвестиций называется условная оптимизация. Решение разобьем на шаги. 1-ый шаг. $k = 3$(таблица 2.5).

Таблица 2.5 – первый шаг решения

e^2	u^3	$e^3 = e^2 - u^3$	$f_3(u^3)$	$F*_3(e^3)$	$u_3(e^3)$
1	0	1	45		

	1	0	90	90	1
2	0	2	45		
	1	1	90		
	2	0	260	260	2
3	0	3	45		
	1	2	90		
	2	1	260		

Таблица 2.5 – продолжение таблицы

	3	0	615	615	3
4	0	4	45		
	1	3	90		
	2	2	260		
	3	1	615		
	4	0	710	710	4

Далее второй шаг при k =2.

Таблица 2.6 – второй шаг решения

e^1	u^2	$e^2 = e^1 - u^2$	$f_2(u^2)$	$F*_2(e^1)$	$F_1(u^2, e^1)$	$F*_2(e^2)$	$u_2(e^2)$
1	0	1	50	90	140	140	0
	1	0	100	0	100		
2	0	2	50	260	310		
	1	1	100	90	190		
	2	0	370	0	370	370	2
3	0	3	50	615	665	665	0
	1	2	100	260	360		
	2	1	370	90	460		
	3	0	465	0	465		

4	0	4	50	710	760	760	0
	1	3	100	615	715		
	2	2	370	260	630		
	3	1	465	90	555		
	4	0	750	0	750		

Далее третий шаг при $k = 3$.

Таблица 2.7 – третий шаг решения

e^0	u^1	$e^1 = e^0 - u^1$	$f_1(u^1)$	$F*_1(e^0)$	$F_0(u^1, e^0)$	$F*_1(e^1)$	$u_1(e^1)$
1	0	1	30	140	170	170	0
	1	0	70	0	70		
2	0	2	30	370	400	400	0
	1	1	70	140	210		
	2	0	365	0	365		
3	0	3	30	665	695	695	0
	1	2	70	370	440		
	2	1	365	140	505		
	3	0	430	0	430		
4	0	4	30	760	790	790	0
	1	3	70	665	735		
	2	2	365	370	735		
	3	1	430	140	570		
	4	0	600	0	600		

Примечание: Столбцы 1 (вложенные средства), 2 (проект) и 3 (остаток средств) для всех трех таблиц одинаковы, поэтому их можно было бы сделать общими. Столбец 4 заполняется на основе исходных данных о

функциях дохода, значения в столбце 5 берутся из столбца 7 предыдущей таблицы, столбец 6 заполняется суммой значений столбцов 4 и 5 (в таблице 3-го шага столбцы 5 и 6 отсутствуют).

В столбце 7 записывается максимальное значение предыдущего столбца для фиксированного начального состояния, и в 8 столбце записывается управление из 2 столбца, на котором достигается максимум в 7.

Второй этап решения задачи распределения инвестиций называется безусловная оптимизация.Из таблицы 3-го шага имеем:

$F *_1 (e^0 = 4млн.руб.) = 780\ тыс.руб$, то есть максимальная прибыль от инвестирования$e^0 = 4млн.руб. = 780\ тыс.руб.$

Из этой же таблицы получаем, что первому предприятию следует выделить$u *_1 (e^0 = 4млн.руб.) = 0\ млн.руб.$ При этом остаток средств составит: $e^1 = e^0 - u^1, e^1 = 4 - 0 = 4\ млн.руб.$

Из таблицы 2-го шага имеем $F *_2 (e^1 = 4\ млн.руб.) = 740\ тыс.руб.$,т.е. максимальная прибыль при $e^1 = 4\ млн.руб. = 740\ тыс.руб.$

Из этой же таблицы получаем, что второму предприятию следует выделить $u *_2 (e^1 = 4\ млн.руб) = 1\ млн.руб.$

При этом остаток средств составит $e^2 = e^1 - u^2, e^2 = 4 - 1 = 3\ млн.руб.$

Последнему предприятию достается 3 млн.руб. Итак, инвестиции в размере 4 млн.руб. необходимо распределить следующим образом: первому предприятию ничего не выделять, второму предприятию выделить 1 млн.руб., третьему предприятию выделить 3 млн.руб., что обеспечит максимальную прибыль, равную 780 тыс.руб[5].

2.4 Задача о загрузке транспортного средства

Предположим что самолет загружается предметами Nразличных типов (таблица 2.8) с весом w_jи стоимостью $c_j, \mathrm{j} = 1, \mathrm{n}.$ Максимальная

грузоподъемность равна $W = 5$.Определить максимальную стоимость груза, вес которого не более W.

Таблица 2.8 - Исходные данные

Тип j	Вес w_j	Стоимость c_j
1	3	45
2	1	60
3	2	30

Сделаем математическую постановку. Пустьx_j– количество предметовj-го типа, загружаемых в самолет. Тогда математическая модель имеет вид:

$$Z(X) = \sum_{j=1}^{n} c_j x_j \to max. \quad (2.2)$$

$$\sum_{j=1}^{n} w_j x_j \leq W. \quad (2.3)$$

$$x_j \geq 0. \quad (2.4)$$

$$X_j - \text{целые}. \quad (2.5)$$

Отличительными особенностями задачи динамического программирования будут:

1. Этапjсвязан с загрузкой предметовj-го типа количествеx_jединиц (x_j– управляемая переменная);

2. Состояние загружаемого самолетаs_j на этапеjопределяется через ограничение (2.3) математической модели (2.2) –(2.5). В алгоритме прямой прогонки состояния$S_j = \sum_{i=1}^{j} w_i x_i, j = 1 \dots n; 0 \leq S_j \leq W; S_n = W$.

В алгоритме обратной прогонки:

$S_j = \sum_{i=j}^{n} w_i x_i, j = 1 \ldots n; 0 \leq S_j \leq W;\ S_1 = W$

3. Цель управления на этапе $z_j = c_j x_j$.

4. Варианты решенияx_jэтапа jописываются количеством предметов типаj: $x_j = 0,1 \ldots, \left[\frac{w}{w_j}\right]$ – целые

Пусть $F_j(S_j)$ – значение целевой функции (максимальная стоимость предметов, включенных на этапах$j, j = 1,..,n$при заданном состоянии системыS_j.

Рекуррентное соотношение для процедур обратной прогонки:

$$F_{n+1}(S_{n+1}) = 0. \quad (2.6)$$

$$F_j(S_j) = max\{c_j x_j + F_{jd}(S_j - w_j x_j)\}, j = 1 \ldots n. \quad (2.7)$$

$$S_j = 0,1 \ldots W. \quad (2.8)$$

$$S_j \geq w_j x_j. \quad (2.9)$$

Таблица 2.9 - Первый этап

$S3$	$c_3 x_3$						opt	
	$X3 = 0$	$X_3 = 1$	$X_3 = 2$	$X_3 = 3$	$X_3 = 4$	$X_3 = 5$	$F_3 * (S_3)$	$X_3 *$
0	0	-	-	-	-	-	0	0
1	0	30	-	-	-	-	30	1
2	0	30	60	-	-	-	60	2
3	0	30	60	90	-	-	90	3
4	0	30	60	90	120	-	120	4
5	0	30	60	90	120	150	150	5

Таблица 2.10 - Второй этап

S_2	$C_2X_2 + F_3(S_2 - w_2x_2)$		opt	
	$X_2 = 0$	$X_2 = 1$	$F_2 * (S_2)$	$X_2 *$
0	$0 + 0 = 0$	—	0	0
1	$0 + 30 = 30$	—	30	0
2	$0 + 60 = 60$	—	60	0
3	$0 + 90 = 90$	$80 + 0 = 80$	90	0
4	$0 + 120 = 120$	$80 + 30 = 110$	120	0
5	$0 + 150 = 150$	$80 + 60 = 140$	150	0

Значение условного оптимума $F_3(S_3)$ берется из предыдущей таблицы.

Таблица 2.11 - Третий этап

$S1$	$C_1x_1 + F_2(S_1 - w_1x_1)$				
	$X_1 = 0$	$X_1 = 1$	$X_1 = 2$	$F_1 * (S_1)$	X_1 ^
5	$0 + 150 = 150$	$65 + 90 = 155$	$130 + 30 = 160$	160	2

Определение управляемых переменных начинается с последней таблицы (обратный ход):

$x_1 * = 2 \rightarrow S_1 = 2 * 2 = 4$

$S_2 = 5 - 4 = 1 \rightarrow x2 = 0 \rightarrow S_3 = 1 - 0 = 1 \rightarrow x_3 = 1$

Оптимальное решение $X * = (2,0,1)$

2.5 Задача о наикратчайшем пути через сети

Сеть дорог с двухсторонним движением задана матрицей расстояний в км (матрицей весовых коэффициентов). Необходимо найти для данной сети дорог самый короткий маршрут из пункта 1 в пункт 10.

Схематически сетевая модель задачи изображена на рис. 6.3 в виде неориентированной сети.

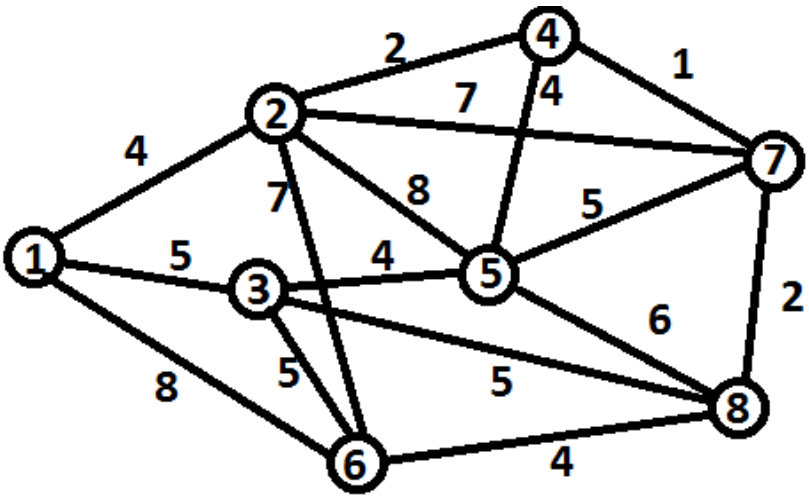

Рисунок 2.1 – Сетевая модель задачи

Продемонстрируем работу алгоритма Дейкстры, отмечая временные и постоянные пометки непосредственно в узлах сети. Для этого каждый узел изобразим в виде овала и разделим его на три части следующим образом

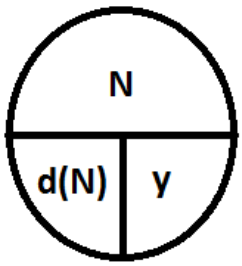

Рисунок 2.2 – Алгоритм Дейкстры

ГдеN -номер узла,$d(N)$- временная пометка узла N, если она становится постоянной, то ее подчеркиваем. Т.е. получим $d(N)$,y - номер узла, придя из которого получена пометка $d(N)$

На первом шаге решениямык исходному узлу приписываем постоянную пометку, равную нулю, т.е.$d(1) = 0$и$y: = 1$. Всем остальным узлам, приписываем временные пометки, равные∞, т.е.$d(j) = \infty , j = 2, \dots ,8$. Отметим это на сети.

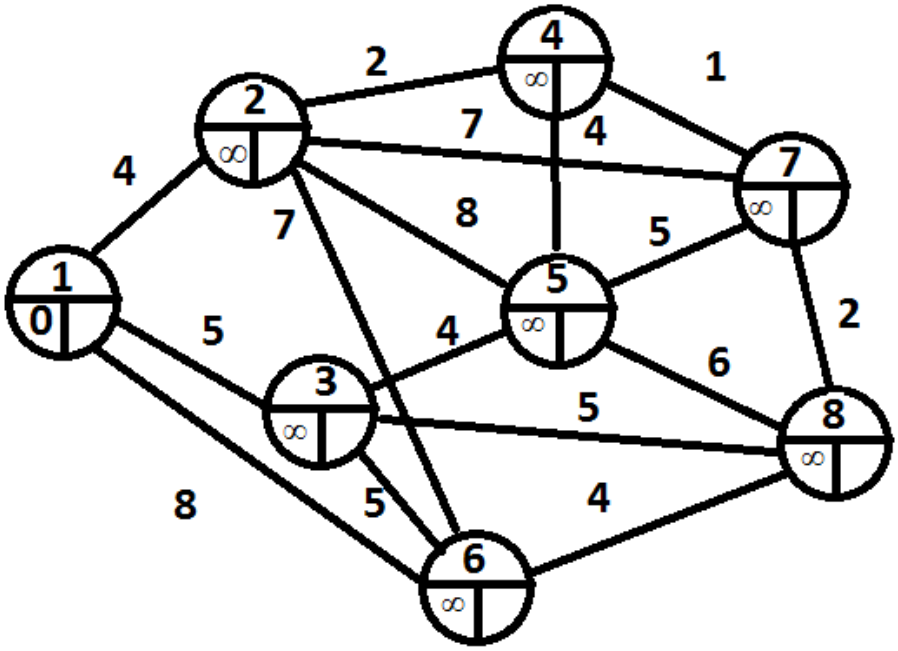

Рисунок 2.3 – Первый шаг решения задачи

На первом шаге узлы *2*,*3*и*6*смежные с последним, получившим постоянную пометку узлом*1*. Новые значения временных пометок этих узлов будут:

$$d(2) := \min\{d(2), d(1) + c_{12}\} = \min\{\infty, 0 + 3\} = 3.$$

$$d(3) := \min\{d(3), d(1) + c_{13}\} = \min\{\infty, 0 + 4\} = 4.$$

$$d(6) := \min\{d(6), d(1) + c_{16}\} = \min\{\infty, 0 + 9\} = 9.$$

Минимальной временной пометкой является пометка узла *2*, т.к. $\min\{d(2), d(3), d(4), d(5), d(6), d(7), d(8)\} = \min\{3,4,9, \infty. \infty. \infty. \infty\} = 3 = d(2)$.

Поэтому она становится постоянной. Подчеркиваем ее и$y := 2$.Помечаем дугу (1,2). Отметим это на сети:

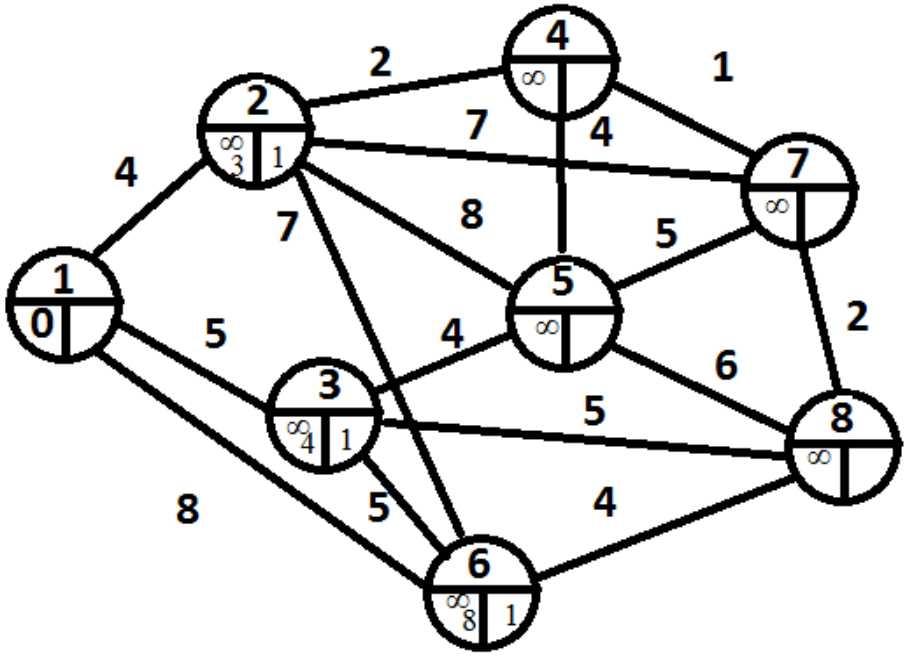

Рисунок 2.4 – Второй шаг решения задачи

На втором шаге поскольку узел*8*не помечен постоянной пометкой, переходим к шагу 2.Узлы*4,5,6*и*7*смежные с последним, получившим постоянную пометку узлом*2*. Новые значения временных пометок этих узлов будут:

$$d(4) := \min\{d(4), d(2) + c_{24}\} = \min\{\infty, 3 + 2\} = 5.$$

$$d(5) := \min\{d(5), d(2) + c_{25}\} = \min\{\infty, 3 + 9\} = 12.$$

$$d(6) := \min\{d(6), d(2) + c_{26}\} = \min\{9,3 + 8\} = 9.$$

$$d(7) := \min\{d(7), d(2) + c_{27}\} = \min\{\infty, 3 + 8\} = 11.$$

Минимальной временной пометкой является пометка узла *3*, т.к. $\min\{d(3), d(4), d(5), d(6), d(7), d(8)\} = \min\{4,5,12,9.11.\infty\} = 4 = d(3)$.

Поэтому она становится постоянной. Подчеркиваем ее и$y := 3$Помечаем дугу (1,3). Отметим это на сети.

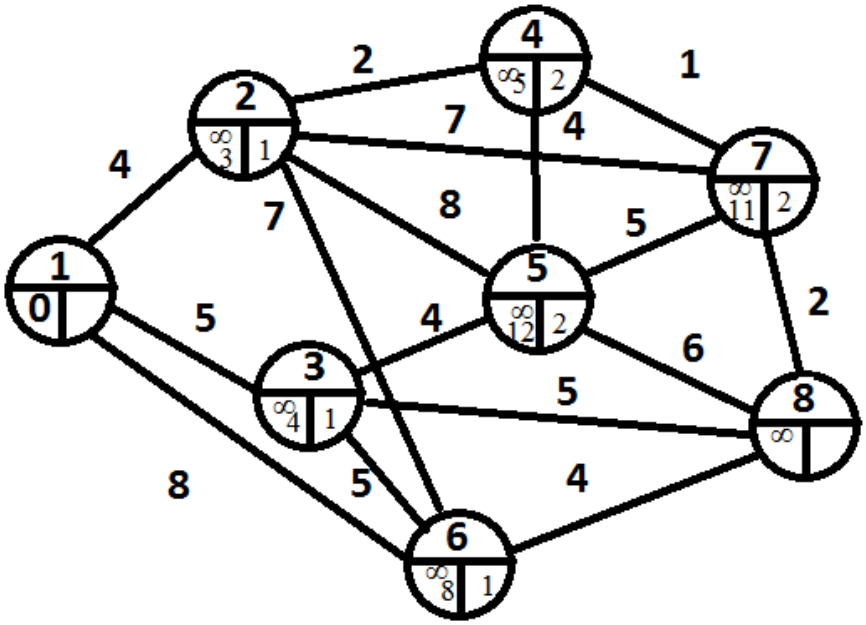

Рисунок 2.5 –Третий шаг решения задачи

На третьем шаге поскольку узел*8*не помечен постоянной пометкой, переходим к шагу2. Узлы*5*,*6*и*8*смежные с последним, получившим постоянную пометку узлом*3*. Новые значения временных пометок этих узлов будут:

$$d(5) := \min\{d(5), d(3) + c_{35}\} = \min\{12{,}4 + 6\} = 10.$$

$$d(6) := \min\{d(6), d(3) + c_{36}\} = \min\{9{,}4 + 7\} = 9.$$

$$d(8) := \min\{d(8), d(3) + c_{38}\} = \min\{\infty, 4 + 7\} = 11.$$

Минимальной временной пометкой является пометка узла*4*, т.к.$\min\{d(4), d(5), d(6), d(7), d(8)\} = \min\{5{,}10{,}9{,}11.11\} = 5 = d(4)$

Поэтому она становится постоянной. Подчеркиваем ее и$y := 4$.Помечаем дугу (2,4). Отметим это на сети.

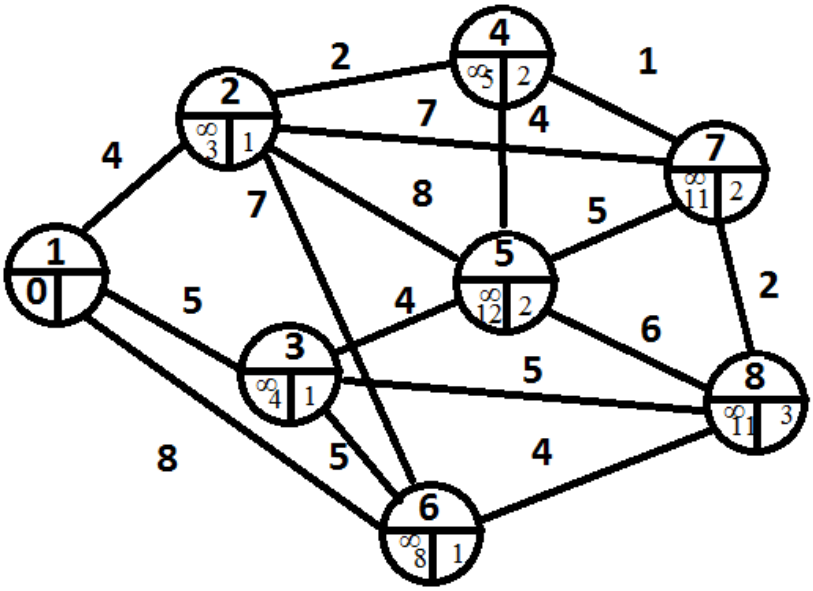

Рисунок 2.6– Четвертый шаг решения задачи

На четвертом шаге поскольку узел*8*не помечен постоянной пометкой , переходим к шагу 2.Узлы*5*и*7*смежные с последним, получившим постоянную пометку узлом*4*. Новые значения временных пометок этих узлов будут:

$$d(5) := \min\{d(5), d(4) + c_{45}\} = \min\{10{,}5 + 5\} = 10.$$

$$d(7) := \min\{d(7), d(4) + c_{47}\} = \min\{11{,}5 + 1\} = 6.$$

Минимальной временной пометкой является пометка узла*7*, т.к. $\min\{d(5), d(6), d(7), d(8)\} = \min\{10{,}9{,}6.11\} = 6 = d(7)$она становится постоянной. Подчеркиваем ее и$y := 7$.Помечаем дугу (4,7). Отметим это на сети:

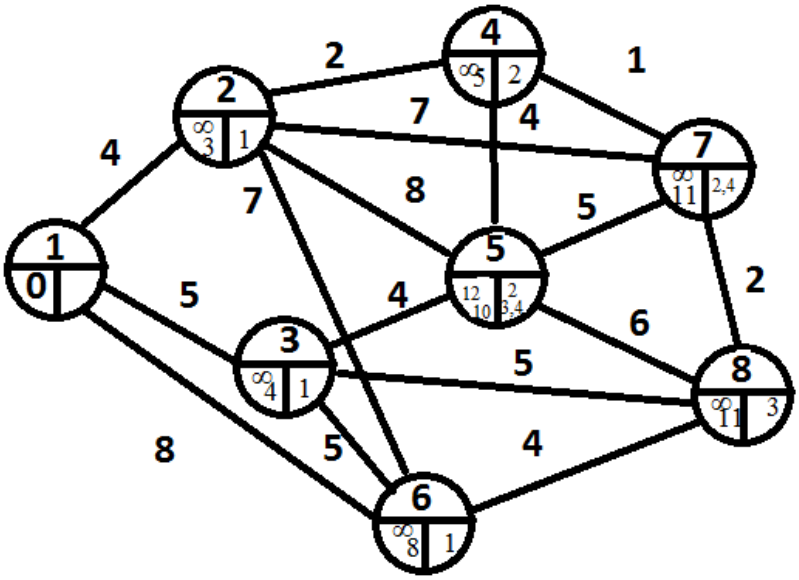

Рисунок 2.7– Пятый шаг решения задачи

На пятом поскольку узел*8*не помечен постоянной пометкой, переходим к шагу 2. Узлы *5* и *8* смежные с последним, получившим постоянную пометку узлом*7*. Новые значения временных пометок этих узлов будут:

$$d(5) := \min\{d(5), d(7) + c_{75}\} = \min\{10,6+6\} = 10.$$

$$d(8) := \min\{d(8), d(7) + c_{78}\} = \min\{11,6+2\} = 8.$$

Минимальной временной пометкой является пометка узла*8*, т.к.$\min\{d(5), d(6), d(8)\} = \min\{10,9,8\} = 8 = d(8)$она становится постоянной. Подчеркиваем ее и$y := 8$.Помечаем дугу (7,8). Отметим это на сети.

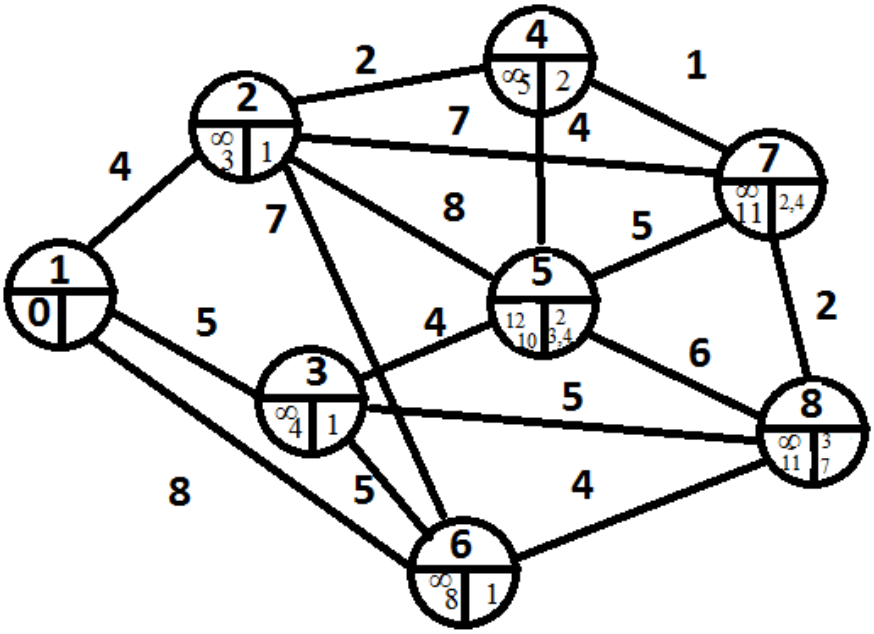

Рисунок 2.8– Шестой шаг решения задачи

На шестом шаге поскольку узел*8*помечен постоянной пометкой, то алгоритм завершает работу: кратчайший путь из узла1в узел назначения8 найден.

Построенное дерево кратчайших путей состоит из дуг (1,2), (1,3), (2,4), (4,7), (7,8)(они помечены на последней сети). Кратчайший путь от узла*1*до узла*8*составляет 8 км, т.к. постоянная пометка этого узла $d(8) = 8$и состоит из дуг (1,2), (2,4), (4,7), (7,8).

2.6 Задача расписания работы технологической линии (задача Джонсона)

Рассмотрим проблему последовательной обработки на двух станках N разных деталей, если известно время A_i и B_i обработки i-й детали на соответствующих станках. Очевидно, что первая машина будет полностью загружена, но вторая может периодически простаивать. Мы попытаемся найти порядок обработки, который минимизирует время простоя второй

машины и тем самым уменьшает общее время обработки деталей. Если мы обозначим через X_i - простое ожидание i-й части, то:

$$X_1 + X_2 = max(A_1 + A_2 - B_1, A_1). \quad (2.10)$$

$$X_1 + X_2 + X_3 = max(A_1 + A_2 + A_3 - B_1 - B_2, A_1 + A_2 - B_1, A_1). \quad (2.11)$$

$$\sum x_i = max(\sum A_i - \sum B_i). \quad (2.12)$$

Если мы обозначим через $F\,(t, A_k, B_k \,/\, k = 1..N)$ - общее время обработки N частей, при условии, что вторая машина включается с задержкой t и используется оптимальный порядок обработки, то с учетом С учетом принципа оптимальности (независимо от выбора исходных деталей процедура выбора следующих должна быть оптимальной) имеем:

$F(t, A_k, B_k / k = 1 \ldots N) = min(A_i + F(B_i + max(t - A_i, 0), A_k, B_k = 1 \ldots N, k \neq i))$ **Ошибка**

.

Если после i-й детали при оптимальном порядке обрабатывается j-я, то: $F(t, A_k, B_k / k = 1 \ldots N) = A_i + A_j + F(t_{ij}, A_k, B_k / k = 1 \ldots N;\ k \neq i, j)$,

где

$t_{ij} = B_i + max[B_j + max(t - A_j, 0) - A_j, 0] = B_i + B_j - A_i - A_j + max[t, max(t, max(A_j - A_i - B_j, A_j))]$

Если $max(A_i + A_j - B_j, A_i) < max(A_j + A_i - B_j, A_j)$,

то сначала разумнее обрабатывать j-ю деталь.

Можно показать, что указанное условие необходимости перестановки эквивалентно условию: $min(A_j, B_i) < min(A_j, B_j)$

Соответственно ищем наименьшее из всех значений A_i и B_i. Если найденное значение совпадает с некоторым значением A_i, то мы помещаем i-ю часть, которая должна быть обработана первой; если он совпадает с

некоторым B_i, то последний. Мы повторяем эту процедуру для всех других деталей.

Пусть информация о времени обработки задана таблицей:

Таблица 2.12 - исходные данные

3	4
7	3
4	5
8	6
5	9
8	7

Решим задание по шагам. Первый шаг: минимальное из значений равно 3 и соответствует A1: первая деталь обрабатывается первой

Таблица 2.13 - первый шаг

3	4
-	-
-	-
-	-
-	-
-	-

Во втором шаге минимальное из значений равно 3 и соответствует B2: вторая деталь обрабатывается последней.

Таблица 2.14 - второй шаг

3	4
-	-
-	-

-	-
-	-
7	3

В третьем шаге минимальное из значений равно 4 и соответствует $A3$: третья деталь обрабатывается первой.

Таблица 2.15 - третий шаг

3	4
4	5
-	-
-	-
-	-
7	3

На пятом шаге минимальное значение равно 6 и соответствует $B4$. четвертая деталь обрабатывается последней.

Таблица 2.16 - четвертый шаг

3	4
4	5
-	-
-	-
8	6
7	3

На пятом шаге минимальное значение равно 5 и соответствует $A5$. Пятая деталь обрабатывается первой

Таблица 2.17 - пятый шаг

3	4
4	5
5	9
-	-

8	6
7	3

На шестом шаге минимальное значение равно 6 и соответствует $B6$. Шестая деталь обрабатывается последней.

Таблица 2.18 - шестой шаг

3	4
4	5
5	9
8	7
8	6
7	3

В итоге упорядоченная информация принимает вид:

Таблица 2.19 - итоговый вариант

3	4
4	5
5	9
8	7
8	6
7	3

Время простоя второй машины при первичном порядке равно :

$$max(2\ ,2+8-3\ ,2+8+4-3-3\ ,2+8+4+9-3-3-6\ ,2+8+4+9+6-3-3-6-5\ ,2+8+4+9+6+9-3-3-6-5-8)=$$
$$max(2,7,8,11,12,13=13$$

.

Время простоя при оптимальной перестановке равно :

$$max(2\ ,2+4-3\ ,2+4+6-3-6\ ,2+4+6+9-3-6-8\ ,2+4+6+9+9-3-6-8-7\ ,2+4+6+9+9+8-3-6-8-7-5\ )=$$
$$max(2,3,3,4,6,9)=\ 9$$

Вывод по главе 2

В данной главе сформулированы некоторые из производственных задач, решаемых методом динамического программирования.Представлены математические модели этих задач и аналитическое решение на основе уравнения Беллмана.

ГЛАВА 3 РЕАЛИЗАЦИЯ АЛГОРИТМА РЕШЕНИЯ ПРОИЗВОДСТВЕННЫХ ЗАДАЧ

3.1 Решения производственных задач методом динамического программирования в программной среде MSEXCEL

Перед разработкой программной реализации решение производственных задач методом динамического программирования на программном языке Python, реализуем решение задач в сторонней программной среде.

Воспользуемся программой, работающая с электронными таблица, от разработчиков Windows, программная среда MSEXCEL.

MicrosoftExcel(также иногда называется MicrosoftOfficeExcel[4]) — программа для работы с электронными таблицами, созданная корпорацией Microsoft для MicrosoftWindows, Windows NT и Mac OS, а также Android, iOS и WindowsPhone. Она предоставляет возможности экономико-статистических расчетов, графические инструменты и, за исключением Excel 2008 под Mac OS X, язык макропрограммирования VBA (VisualBasicforApplication). MicrosoftExcel входит в состав MicrosoftOffice и на сегодняшний день Excel является одним из наиболее популярных приложений в мире.

Нами была решена в качестве примера задача о распределение инвестиций.

Предположим, что оптимальные размеры и потоки инвестирования, если прибыль от вложений (X_i) в проекты (A_i) распределилась следующим образом:

Таблица 3.1 - исходные данные

X_i	A_1	A_2	A_3	A_4
0	0	0	0	0
10	8	12	10	9
20	16	22	19	17
30	24	30	26	25
40	28	35	32	32
50	32	37	36	38

Теперь для решения этой задачи воспользуемся Excel.Для этого выделим шаги тренда t_i, вложения x_iи прибылиA_i. Затем для каждого из четырех проектов построим средствами MSExcel графическую зависимость прибыли А от шага тренда ($t = 1, 2, 3, 4, 5, 6$). Активизируем точки графика, щелкнув по ним левой клавишей мыши, затем нажмем правую клавишу и выберем режим «Добавить линию тренда». Для всех четырех проектов наилучшим типом является полиномиальный5- ой степени. С помощью полученных уравнений трендов находим теоретические значения прибыли при различных значениях шага тренда t_i. Уравнения моделей тренда, коэффициенты аппроксимации и теоретические значения прибыли, представлены на рисунке 3.1.

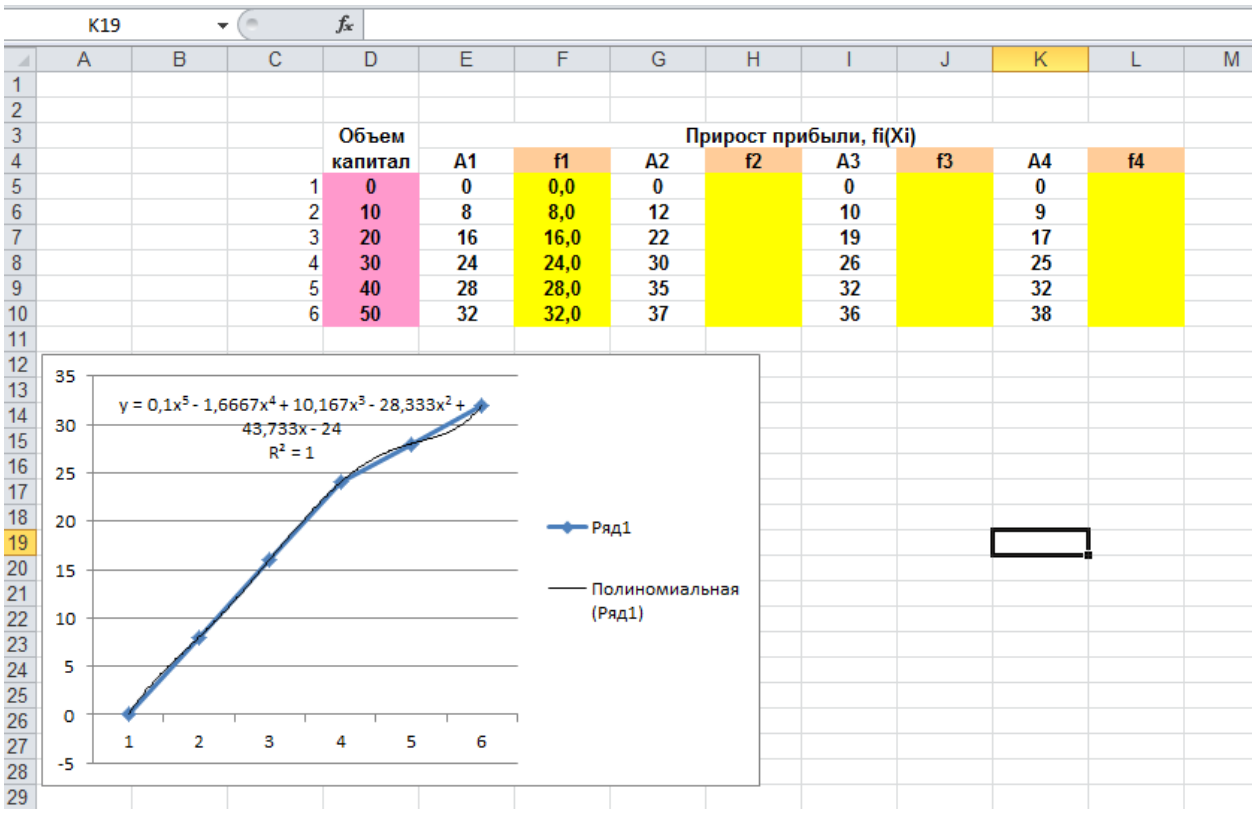

Рисунок 3.1– Графические зависимости прибыли от вложений и полиномиальные тренды этих зависимостей

В ячейку M32 вводим выражение для общей (суммарной) прибыли, которую надо максимизировать, - это сумма всех четырех полиномиальных функций. Зависимыми переменными в этой функции являются искомые значения шагов тренда, которые будут располагаться в ячейках E32-H32. Суммарные вложения не должны превышать 50 тыс. ед., следовательно, вводим ограничение $10 * (E32 + F32 + G32 + H32 - 4)$ в ячейку D37.

Выбираем из главного меню MSExcel режим «Поиск решения» и заполним открывшееся диалоговое окно в соответствии с требованиями. Нажмем клавишу« выполнить» и получим результат оптимизации.

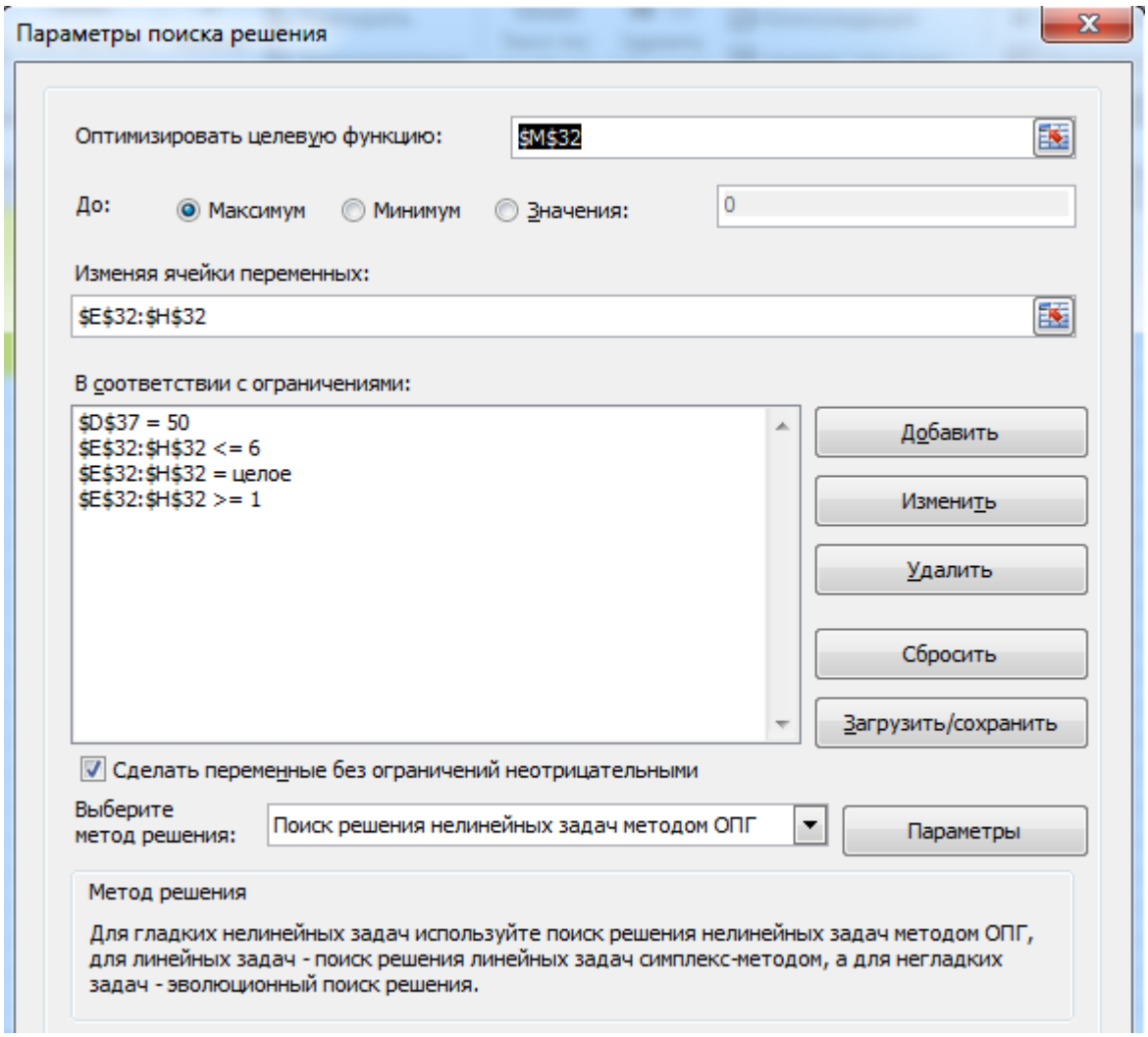

Рисунок3.2 – Модель максимизации прибыли

M32 | =I32+J32+K32+L32

		x1	x2	x3	x4	f1	f2	f3	f4	max
номер шага тренда	1	3	3	2	0,0003	22,0048	19,0094	8,9992	50,0137
Объем капитал	0	20	20	10	A1	A2	A3	A4	

Ограничения
50

Рис. 3.3 – Оптимальное распределение капиталовложений между проектами

Нами была продемонстрировано решение задачи о распределения ресурсов в программной среде MSExcel. Перейдем к обоснованию выбора языка программирования для дальнейшей программной реализации производственных задач

3.2 Выбор языка программирования

При выборе языка программирования мы остановились на языке Python , так как на данном языке программирования нами не было найдено программной реализации решения производственных задач методом динамического программирования, в отличии от других объектно-ориентир ованных языков программирования. Перед выбором среды разработки на данном языке программирования, обоснуем выборнами язык программирования.

Python (в русском языке распространено название питон)– высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Pythonминималистичен. В то же время стандартная библиотека включает большой объём полезных функций.

Python поддерживает структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное программирование. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений, высокоуровневые структуры данных. Поддерживается разбиение программ на модули, которые, в свою очередь, могут объединяться в пакеты.

Эталонной реализацией Python является интерпретатор CPython, поддерживающий большинство активно используемых платформ. Он распространяется под свободной лицензией PythonSoftwareFoundationLicense, позволяющей использовать его без ограничений в любых приложениях, включая проприетарные. Есть реализация интерпретатора для JVM с возможностью компиляции, CLR, LLVM, другие независимые реализации. Проект PyPy использует JIT-компиляцию, которая значительно увеличивает скорость выполнения Python-программ.

Python — активно развивающийся язык программирования, новые версии с добавлением/изменением языковых свойств выходят примерно раз в два с половиной года. Язык не подвергался официальной стандартизации, роль стандарта де-факто выполняет CPython, разрабатываемый под контролем автора языка.

3.3 Выбор графического фреймворка для реализации интерфейса программы

Для реализации программного продукта мной был выбран графический интерфейс от компании JetBrainsPyCharm. Перед тем как приступить кразработке расскажу немного о данной среде разработки.

PyCharm–интегрированная среда разработки для языка программирования Python. Предоставляет средства для анализа кода, графический отладчик, инструмент для запуска юнит-тестов и поддерживает веб-разработку на Django. PyCharm разработана компанией JetBrainsна основе IntelliJ IDEA.

PyCharm работает под операционными системами Windows, Mac OS X и Linux.

PyCharm был выпущен на рынок интегрированных сред разработки для Python для создания конкуренции с PyDev и более распространённой среды разработки KomodoIDE. Бета-версия была выпущена в июле 2010 года, версия 1.0 была выпущена тремя месяцами позже.

Версия 2.0 вышла 13 декабря 2011 года. Версия 3.0 была выпущена 24 сентября 2013 года.

PyCharmCommunityEdition, бесплатная версия с открытым исходным кодом была опубликована 22 октября 2013 года.

В марте 2016 года JetBrains перешла на подписную модель лицензирования, а вместе с этим изменилась и нумерация версий. Теперь номер версии выглядит как YYYY.R, где YYYY — год выпуска, а R — выпуск в течение этого года[8].

3.4 Программная реализация для решения производственных задач на основе метода динамического программирования

По описанной ранее математической модели нами было реализовано программное обеспечения для решения производственных задач на основе метода динамического программирования. Для хранения исходных данных был использован сторонний продукт MSExcel. Используем готовые библиотеки "xird" и "xiwt" для считывания данных с файла MSExceli записывания вывода в другой файл.

```
import xlrd, xlwt
```

Рисунок 3.4 – Импортированные библиотеки

С помощью данных библиотек мы считываем исходные данные из MSExcelв нашу программу, далее опишем метод решения одной из производственной задачи, такую как задача распределения инвестиций.

```
x = variable(9, 'x')
z=(7*x[0] + 3*x[1] +6* x[2] +4*x[3] + 8*x[4] +2* x[5]+x[6] + 5*x[7] +9* x[8])
mass1 = (x[0] + x[1] +x[2] <= 74)
mass2 = (x[3] + x[4] +x[5] <= 40)
mass3 = (x[6] + x[7] + x[8] <= 36)
mass4 = (x[0] + x[3] + x[6] == 20)
mass5 = (x[1] +x[4] + x[7] == 45)
mass6 = (x[2] + x[5] + x[8] == 30)
x_non_negative = (x >= 0)
problem =op(z,[mass1,mass2,mass3,mass4 ,mass5,mass6, x_non_negative])
problem.solve(solver='glpk')
problem.status
```

Рисунок 3.5 – Метод решения задачи

После выполнения метода, данные заносятся в новый файл MSExcel. Результатом данной программы является таблица значений прибыли от реализации продукции на каждом отрезке времени, оптимальная политика управления, и итоговая прибыль.

x1	x2	x3	x4	f1	f2	f3	f4	max
1	3	3	2	0	22,005	19,009	9	50

Рисунок 3.2 – Вывод программы задачи распределения инвестиций

Далее нами описан метод решения производственной задачи такой как: задача замены оборудования. Исходные данные занесены в файл MSExcel.

```
start = time.time()
x1 = pulp.LpVariable("x1", lowBound=0)
x2 = pulp.LpVariable("x2", lowBound=0)
x3 = pulp.LpVariable("x3", lowBound=0)
x4 = pulp.LpVariable("x4", lowBound=0)
x5 = pulp.LpVariable("x5", lowBound=0)
x6 = pulp.LpVariable("x6", lowBound=0)
x7 = pulp.LpVariable("x7", lowBound=0)
x8 = pulp.LpVariable("x8", lowBound=0)
x9 = pulp.LpVariable("x9", lowBound=0)
problem = pulp.LpProblem('0',pulp.LpMaximize)
problem += -7*x1 - 3*x2 - 6* x3 - 4*x4 - 8*x5 -2* x6-1*x7- 5*x8-9* x9, "Функция цели"
problem +=x1 + x2 +x3<= 74,"1"
problem +=x4 + x5 +x6 <= 40, "2"
problem +=x7 + x8+ x9 <= 36, "3"
problem +=x1+ x4+ x7 == 20, "4"
problem +=x2+x5+ x8 == 45, "5"
problem +=x3 + x6+x9 == 30, "6"
problem.solve()
```

Рисунок 3.6 – Метод решения задачи замены оборудования

Результатом данной программы является таблица задачи, которая заносится в новый файл MSExcel, представленная на рисунке ниже.

0	42	-	-	-
1	38	30	***21***	11
2	***34***	27	19	***10***
3	33	***24***	17	9
4	***33***	24	14	8
5	33	24	14	6
6	33	24	14	3

Рисунок 3.3 – Результат программ

ЗАКЛЮЧЕНИЕ

Тема бакалаврской работы посвящена актуальной проблеме реализации алгоритма решения производственных задач на основе метода динамического программирования.

В ходе выполнения бакалаврской работы достигнуты следующие результаты:

1. Проанализирован общий подход динамического программирования к решению некоторых типов производственных задач

2. Приводится аналитическое решение представленных производственных задач с помощью метода динамического программирования.

3. Выполнена реализация алгоритма представленных производственных задач.

Основным результатом выполненной ВКР является программная реализация алгоритма решения производственных задач методом динамического программирования.

Результаты работы могут быть рекомендованы для решения задач управления ресурсами экономических и производственных систем.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Гладких Б.А. Методы оптимизации и исследования операций. Часть 1. Введение в исследование операций. Линейное программирование. — Томск: НТЛ, 2009. — 200 с.
2. Окулов С.М. Динамическое программирование. – М.:Бином. Лаборатория знаний, 2017. – 296 с.
3. Лежнев, А.В. Динамическое программирование в экономических задачах [Текст]: учеб. пособие / Лежнев А.В. - Москва: Бином, 2010. - 176 с.
4. Ширяв В.И. Исследование операций и численные методы оптимизации: учебное пособие / В.И. Ширяв. Москва: Ленанд, 2017. − 224 с.
5. Окулов С.М. Динамическое программирование / С.М. Окулов, О.А. Пестов. Москва: БИНОМ. Лаборатория знаний, 2012. − 260 с.
6. Некрасова М.Г. Методы оптимизации и теория управления: учебное пособие / М.Г. Некрасова. Комсомольск-на-Амуре: КНАГТУ, 2007. − 132 с.
7. Беллман Р. Динамическое программирование. — М.: Репринт оригинального издания иностранной литературы, 1960 год, 2012.

8. Маккинли У. Python и анализ данных. — Перевод с английского. — М.: ДМК Пресс, 2015. — 482 с. —SanjoyDasgupta ,ChristosH. Papadimitriou, UmeshVazirani. Algorithms = Algorithms. — 1-е изд. — McGraw-Hill Science/Engineering/Math, 2006. — С. 336.

9. Кормен, Т., Лейзерсон, Ч., Ривест, Р., Штайн, К. Глава 15. Динамическое программирование // Алгоритмы: построение и анализ = IntroductiontoAlgorithms / Под ред. И. В. Красикова. — 2-е изд. — М.: Вильямс, 2005. — 1296 с.

10. Марк Саммерфилд. Python на практике. — Перевод с английского. — М.: ДМК Пресс, 2014. — 338 с.

11. Доусон М. Программируем на Python. — СПб.: Питер, 2012. — 432 с.

12. Фёдоров Д. Ю. Основы программирования на примере языка Python / Учебное пособие. — СПб.:Юрайт, 2018. — 167 с.

13. Mathsemestr[Электронный ресурс] Задачи ДП - Режим доступа: https://math.semestr.ru/dinam/dinam_manual.php.

14. Марк Лутц. Программирование на Python / Пер. с англ. — 4-е изд. — СПб.: Символ-Плюс, 2011. — Т. I. — 992 с.

15. А. Н. Чаплыгин. Учимся программировать вместе с Питоном. Учебник. — ревизия 226. — 135 с.

16. Lutz, Mark (2013). LearningPython (5thed.). O'ReillyMedia.

17. Hamilton, Naomi (5 August 2008). "TheA-ZofProgrammingLanguages: Python". Computerworld. Archivedfromtheoriginalon 29 December 2008. Retrieved 31 March 2010.

18. Meyn, Sean (2007), Control Techniques for Complex Networks, Cambridge University Press, archived from the original on 2010-06-19

19. Giegerich, R.; Meyer, C.; Steffen, P. (2004), "A Discipline of Dynamic Programming over Sequence Data" (PDF), Science of Computer Programming, 51 (3): 215–263, doi:10.1016/j.scico.2003.12.005

20. Bertsekas, D. P. (2017), Dynamic Programming and Optimal Control (4th ed.), Athena Scientific.