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)

01.03.02 Прикладная математика и информатика
(код и наименование направления подготовки / специальности)

Компьютерные технологии и математическое моделирование
(направленность (профиль)/специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Анализ и реализация алгоритма оптимизации стоимости сетевого проекта»

Обучающийся С.С. Маштаков
(Инициалы Фамилия) (личная подпись)

Руководитель к.т.н., доцент, Н.А. Сосина
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант Старший преподаватель, Е.В. Косс
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2022

Аннотация

Тема выпускной квалификационной работы: «Анализ и реализация алгоритма оптимизации стоимости сетевых проектов».

Объектом исследования работы являются методы оптимизации стоимости сетевых задач.

Целью выпускной квалификационной работы является реализация алгоритма и программы для решения оптимизации стоимости сетевых задач.

Актуальность бакалаврской работы заключается в создании программы для решения оптимизации не только времени сетевой задачи, но и её стоимости. Программный код позволит существенно сократить время всех расчётов при решении задачи по оптимизации сетевых проектов.

В первой главе описываются основные понятия и методы решения сетевых задач.

Во второй главе представлено решение сетевой задачи с использованием метода критического пути, а также метода оптимизации по стоимости проекта. Разработан алгоритм оптимизации стоимости конкретного сетевого проекта.

В третьей главе представлена среда для разработки программного кода, демонстрируются основные функции алгоритма и работа программы.

Результатом ВКР является разработанный алгоритм и программа для решения оптимизации стоимости сетевого проекта.

ABSTRACT

The topic of the graduation work is "Analysis and implementation of an algorithm for optimizing the cost of network projects."

The object of the research is the methods of optimizing the cost of network tasks.

The purpose of the graduation work is to implement an algorithm and a program for solving cost optimization of network tasks.

The relevance of the graduation work is that it helps to create a program to optimize not only the time of the network task, but also its cost. The program code will significantly reduce the time of all calculations when solving the problem of optimizing network projects.

The first chapter describes the basic concepts and methods of solving network problems.

The second chapter presents the solution of the network problem using the critical path method, as well as the optimization method for the cost of the project. An algorithm for optimizing the cost of a specific network project has been developed.

The third chapter presents an environment for developing program code, demonstrates the main functions of the algorithm and the operation of the program.

The result of the graduation work is a developed algorithm and a program for solving the optimization of the cost of a network project.

Оглавление

Введение

Сетевые методы применяются в проектной деятельности при планировании целевых научно-исследовательских и проектно-конструкторских разработок сложных объектов, машин и установок, не имеющих в прошлом никаких аналогов и это диктует необходимость владения новейшими методами планирования и управления.

В настоящее время хорошо разработаны алгоритмы минимизации времени выполнения проекта, но не менее актуально решение более сложной задачи оптимизации стоимости проекта.

В работе на примере конкретного проекта рассматриваются возможности сокращения стоимости проекта за счет увеличения интенсивности работ на критическом пути. Общее сокращение стоимости проекта осуществляется при одновременном сокращении параллельных критических путей.

Цель работы: исследование методов оптимизации стоимости сетевых проектов, разработка на их основе алгоритма решения задачи оптимизации стоимости выполнения проекта, создание программного продукта.

В первой главе описываются основные понятия и методы оптимизации сетевых проектов.

Во второй главе представлено решение сетевой задачи с использованием метода критического пути, а также метода оптимизации по стоимости проекта. Так же представлен анализ самого алгоритма по оптимизации стоимости сетевого проекта.

В третьей главе представлен программный код для решения сетевой задачи с описанием основных функций алгоритма оптимизации стоимости проекта.

Глава 1. Назначение и области применения сетевого проекта

1.1 Основные понятия

«Сетевое планирование и управление (СПУ) – это набор различных методов для управления организационных и контрольных мероприятий, анализ, по его оценке, и сокращению затрат и времени по его выполнению с использованием составления сетевого графика, позволяющего получить прогноз продолжительности и стоимости всего проекта в целом.

Для реализации такого сетевого графика необходимо составить сам план работ по выполнению больших и трудоёмких проектов, которые могут состоять из многих сотен различных операций. Поэтому необходимо описать этот план с применением некоторой математической модели. В качестве неё мы и используем сетевую модель.

Сетевая модель – это описание взаимоотношения нескольких комплексов действий (в нашем случае работ), которые заданы в форме сети и, по сути, являются графическим изображением, которым и называют сетевым графиком.

Главными составляющими сетевой модели являются события проекта, а также работа и их данные.

Событие – это момент времени завершения какого-либо процесса, который определяет завершение отдельного этапа проекта. Оно также может являться частным результатом отдельной работы и началом выполнение следующего этапа проекта. При этом момент события не имеет каких-либо критериев по времени поскольку происходит мгновенно с наступлением или завершением любых работ» [9].

Работа – это действия, для выполнения которых необходимо использование ресурсов и времени. В сетевом проекте этот термин имеет несколько значений.

Первое – это сама работа и действия, по её выполнению в ходе которых и требуется затрата ресурсов. Такая работа должна иметь чёткое описание и предоставленное лицо по её выполнению.

Второе – это момент ожидания, протяжённый процесс времени, который не требует использования каких-либо ресурсов, для его выполнения необходимо лишь время. Примером такой работы может служить затвердевание бетона или время просушки окрашенных объектов.

И третье – это фиктивная работа или зависимость переходов взаимосвязанных между собой действительных работ, которая также не имеет затрата ресурсов или времени. Она показывает непосредственную связь одной работы по отношению к результатам другой.

Также немаловажным элементом для построения сетевого графика является путь.

Путь – эта любая последовательность действий по выполнению работ, в которой конец любого события означает начало следующего и так до выполнения всего проекта. Путём от исходного (начального) этапа до конечного называют полным [3][16][18].

На рисунке 1 представлены основные элементы сетевой модели.

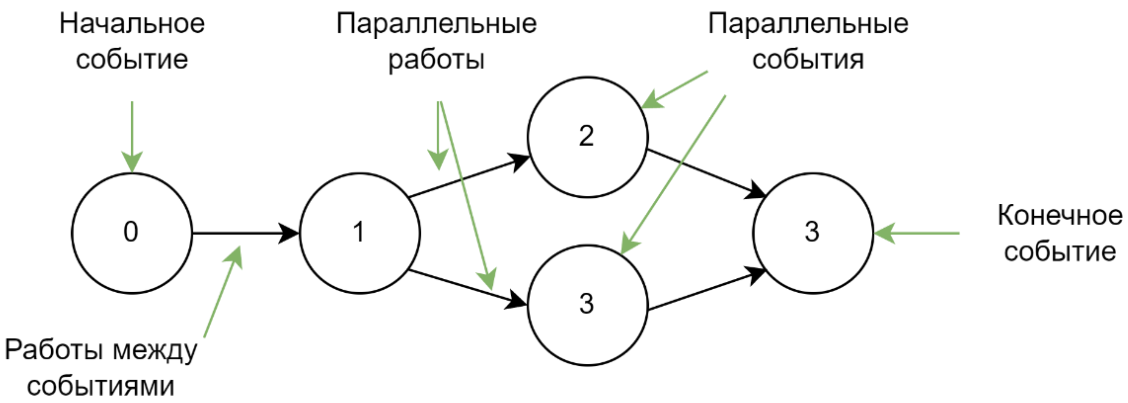

Рисунок 1 – Основные элементы сетевой модели

1.2. Правила построения сетевого графика

«Сетевые графики составляются на начальном этапе планирования сетевого проекта. Вначале планируемый процесс разбивается на отдельные работы, составляется перечень работ и событий, продумываются их логические взаимосвязи и последовательность выполнения. С их помощью оценивается продолжительность и период каждой работы. Затем составляется (сшивается) сетевой график. После упорядочения сетевого графика рассчитываются параметры событий и работ, определяются резервы времени и критический путь. После нахождения критического пути и резервов времени работ и оценки вероятности выполнения проекта в заданный срок должен быть проведён всесторонний анализ сетевого графика и приняты меры по его оптимизации. Этот весьма важный этап в разработке сетевых графиков раскрывает основную идею СПУ. Он заключается в приведении сетевого графика в соответствие с заданными сроками и возможностями организации, разрабатывающей проект» [10].

При построении сетевого графика необходимо использовать следующие правила:

– сетевой график строится слева направо, и каждое событие с большим порядковым номером должно быть расположено правее предыдущего;

– стрелки, изображающие работы, должны также располагаться слева направо;

– в сетевом графике должно существовать только одно начальное и одно конечное событие;

– в сети не должно быть «тупиков», т.е. промежуточных событий, из которых не выходит ни одна работа;

– на сетевом графике не должно быть циклов, состоящих из взаимосвязанных работ, создающих замкнутую цепь [2][6].

1.3 Методы решения сетевых задач

Для того чтобы начать решение сетевого проекта, необходимо составить его сетевой график, основываясь на входных данных проекта и решить задачу этого сетевого графика. Для решения такой задачи в период 1956-1958 гг. были реализованы два метода, которые смогли получить наибольшее признание в сетевом планировании. Первый метод − метод критического пути МКП (Critial Path Method - CPM). Второй − метод оценки и пересмотра проектов, или PERT (Project Evaluation and Review Technique - PERT). Метод PERT был разработан для военно-морских сил США для координации исследовательских и проектных работ. Первое успешное применение этого метода было при разработке программы создания баллистических ракет "Поларис". Немногим позже, на основе метода PERT, был разработан метод критического пути МКП строительным концерном "Дюпон" [7].

1.3.1 Метод критического пути (CPM)

Метод критического пути – это один из способов планирования и управления сроками проекта. В основе данного метода лежит выявление наиболее длительной последовательности процессов, рассчитываемых от начала самого проекта и до его окончания, учитывая при этом их взаимосвязи. Для использования такой методики необходимо создание модели проекта, которая включает в себя:

– список задач, необходимых для реализации проекта;

– взаимосвязи между этими операциями, их последовательность;

– период времени, необходимый для реализации каждой задачи проекта, включающий в себя минимальное и максимальное время на их выполнение.

Ниже, на рисунке продемонстрирован пример метода критического пути на сетевом графике.

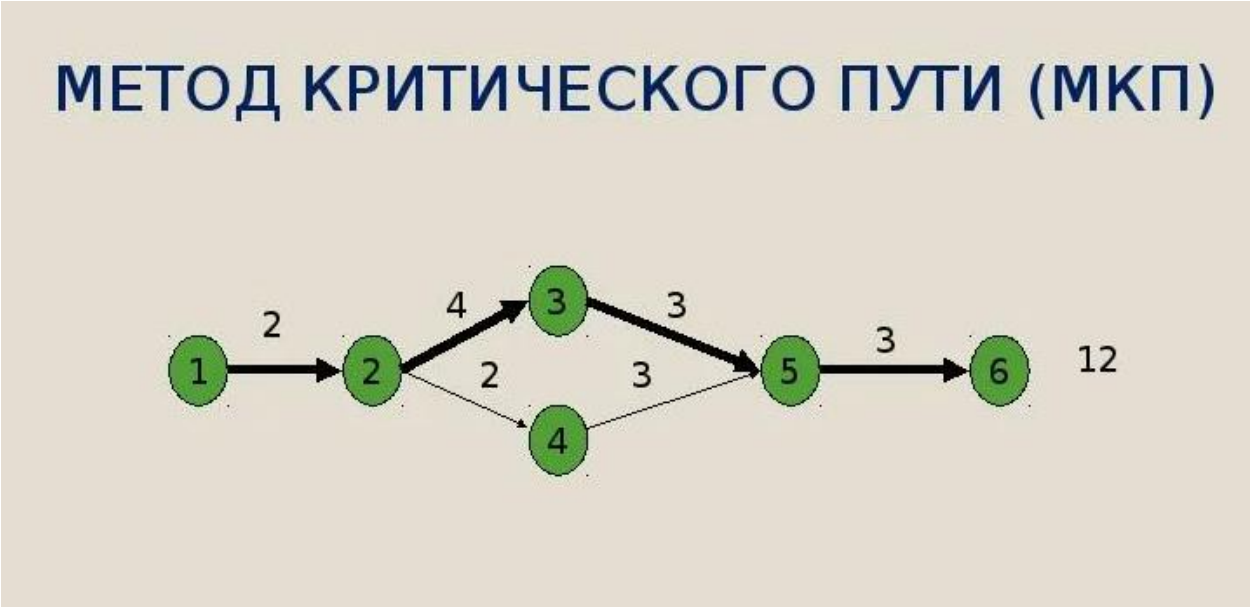

Рисунок 2 – Критический путь

Имея эти данные, применив метод критического пути можно рассчитать максимальную продолжительность времени необходимого для выполнения всех работ проекта. Также этот метод позволяет выявить самые ранние и поздние события начала и окончания каждой выполненной работы, которые не приведут к увеличению времени для завершения сетевой модели проекта.

Все процессы, которые лежат в основе критического пути, называются критическими операциями (критической работой). Суммарное критическое время всех этих работ и будут составлять само критическое время проекта.

Именно этот критический путь позволяет нам оптимизировать сетевую модель проекта, за счет нахождения параллельных критических путей сетевого графика и сокращения работ, лежащих на изначальном критическом пути. Однако при его использовании возникают моменты, где необходимо использовать логические операции для дальнейшего принятия решений, таких как: «стоит ли сокращать критические пути для нового резерва и не увеличит ли это стоимость всего проекта?» И этот момент нам придется решить при составлении алгоритма оптимизации.

1.3.2 Метод PERT

Метод PERT (англ. Program Evaluation Review Technique, рус. Техника Оценки и Анализа Программ и проектов) – метод позволяющий производить анализ взаимосвязанный производственных процессов для нахождения времени, необходимого для завершения какого-то определённого процесса. Этот метод является инструментом, который вычисляет ожидаемое значение продолжительности работ или отдельного процесса. При управлении проектами метод PERT практически всегда используется в сочетании с методом критического пути (*англ. CPM,* Critical Path Method).

«Метод PERT и метод критического пути принципиально различаются по области их применения. Метод критического пути используется для оценки сроков завершения всего проекта или групп взаимосвязанных задач, а метод PERT применяют для оценки длительности отдельных задач проекта» [5].

Для того, чтобы можно было оценить время выполнения всех процессов (задач) проекта, необходимо знать пессимистическую и оптимистическую вероятностные оценки продолжительности этих процессов. В упрощенном виде весь метод можно представить данной формулой:

$$T = \frac{(t_{\text{опт}} + 4M + t_{\text{пес}})}{6}, \quad (1)$$

T – время выполнения задачи или процесса;

$t_{\text{опт}}$ – оптимистическое время, то есть минимальное время для выполнения работы;

$t_{\text{пес}}$ – пессимистическое время, то есть максимальное время для выполнения работы;

M – наиболее вероятное оценка ожидания (длительности работы).

Данное уравнение используется для того, чтобы рассчитать сколько времени необходимо для каждого этапа проекта.

1.4 Выбор метода для разработки алгоритма оптимизации стоимости

«Оптимизация сетевого графика в зависимости от полноты решаемых задач может быть условно разделена на частную и комплексную. Видами частной оптимизации сетевого графика являются: минимизация времени выполнения комплекса работ при заданной его стоимости; минимизация стоимости комплекса работ при заданном времени выполнения проекта.

Комплексная оптимизация представляет собой нахождение оптимального соотношения величин стоимости и сроков выполнения проекта в зависимости от конкретных целей, ставящихся при его реализации» [10].

«Для оптимизации сетей и, в частности, для создания алгоритма оптимизации сетевых проектов могут быть использованы эвристические методы, т.е. методы, учитывающие индивидуальные особенности сетевых графиков» [1].

При использовании таких методов, можно руководствоваться уже готовыми математическими методами и приёмами. Это может помочь при создании собственного алгоритма решения какой-либо задачи. В нашем случаи рассматриваются два схожих метода: CPM и PERT.

И поскольку в реализации алгоритма нам необходим именно расчёт значений сроков и стоимости выполнения всего проекта в целом, а не частями, можно сделать вывод что для этого нам больше подходит метод CPM, ведь он в большей степени соответствует нашим требованиям в разработке алгоритма.

На рисунке 3 изображён наглядный пример таблицы данных для решения сетевого проекта.

Работа	Опирается на работы	$t_{пес}$	$t_{опт}$	Стоимость сокращения работы на один день, s_k
b_1	–	8	3	6
b_2	–	10	4	8
b_3	–	6	1	4
b_4	b_1	9	1	6
b_5	b_1	5	1	3
b_6	b_3	2	1	2
b_7	b_2, b_5, b_6	4	1	3
b_8	b_2, b_5, b_6	13	4	9
b_9	b_4, b_7	8	1	5
b_{10}	b_3	17	6	10
b_{11}	b_2, b_5, b_6, b_{10}	10	2	7

Рисунок 3 – Таблица данных проекта

Для проведения оптимизации сетевого графика необходимо знать значение не только продолжительности времени работ, то есть их максимальные и минимальные значения, но также показатели затрат на проведение ускорения каждой работы. И чтобы работать с данными было проще, нужно иметь заполненную таблицу с всеми необходимыми данными для расчета сетевого проекта (рис. 3).

Самый очевидный вариант оптимизации сетевого графика с учетом снижения стоимости предполагает использование резервов времени работ, рассчитываемых после построения сетевого графика.

И в этом случае можно пойти двумя путями:

– найти критический путь сетевого графика проекта без использования сокращений всех работы проекта, то есть использовать весь лимит времени, срок выполнения проекта в этом случае будет максимальным, и далее сокращать время выполнения работ критического пути, тем самым создавая новые критические пути и уменьшая стоимость и время проекта;

– найти критический путь сетевого графика проекта с использования сокращений всех работ, время выполнения проекта при этом станет минимальным. И в данных условиях, использовать отмену сокращений работ, лежащих на некритическом пути, тем самым уменьшив стоимость проекта, не меняя при этом сроки его выполнения.

Выводы к главе 1

В первой главе представлены теоретические основы методов решения сетевых задач. Рассматривались два метода: метод критического пути МКП (Critial Path Method и метод оценки и пересмотра проектов PERT (Project Evaluation and Review Technique). На основе анализа методов, принято решение на первом этапе исследований использовать метод критического пути (CPM).

Глава 2. Алгоритм оптимизации стоимости сетевого проекта

Оптимизация и нахождения идеального баланса между сроками и ценой проекта всегда являлась и будет являться одной из важных задачей для многих предпринимателей и не только. Люди всегда стримились к улучшению и упрощению их жизни, и поэтому они разрабатывают всевозможные способы для экономии денежных средств и времени. Оптимизация сетевых проектов может является одним из способов приближения человечества к этой цели [12][19].

2.1 Решение сетевой задачи

Предположим, что некоторая компания решает в кратчайшие сроки организовать производство какого-либо товара. Состав работ по этому проекту (см. таблицу 1) состоит из 12 мероприятий. Необходимо найти оптимальный по стоимости вариант выполнения проекта [11].

Таблица 1 – Входные данные для решения сетевой задачи

Номер работы n	Обозначение a_n	Путь работы $put(a_n)$	t_{max}	t_{min}	Сокращение 1 дня работы S_k
1	a_1	–	15	7	9
2	a_2	a_1	22	6	10
3	a_3	a_1	8	3	6
4	a_4	a_2	30	9	10
5	a_5	a_3	16	5	11
6	a_6	a_4	15	3	9
7	a_7	a_5	13	3	8
8	a_8	a_5	8	2	6
9	a_9	a_5	9	2	7
10	a_{10}	a_6, a_7	6	1	4
11	a_{11}	a_8	7	1	3
12	a_{12}	a_9, a_{10}, a_{11}	3	1	2

Стоимость выполнения одного дня проекта равна: S = 11 денежным единицам. Считая $t_{\text{пес}}$ продолжительностью работы с минимальной допустимой интенсивностью ($t_{\text{пес}} = t_{max}$), а $t_{\text{опт}}$ – продолжительностью работы с максимальной возможной интенсивностью ($t_{\text{опт}} = t_{min}$), найдем оптимальный по стоимости вариант выполнения проекта.

Для этого строим сетевой график проекта и рассчитываем его критический путь.

Но перед построением рассмотрим основные обозначения элементов сетевого графика, рисунок 4.

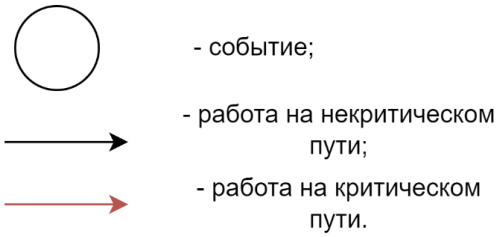

Рисунок – 4 Обозначения элементов сетевого графика

Теперь строим сетевой график проекта с минимальной интенсивностью проведения работ, рисунок 5.

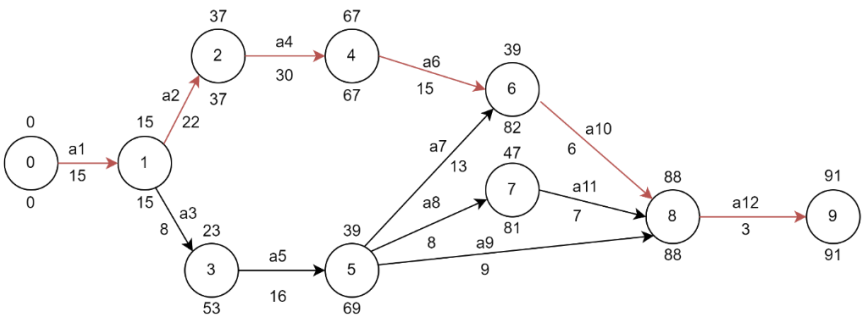

Рисунок 5 – Сетевой график проекта с минимальной интенсивностью проведения работ

Построив для него сетевой график и проанализировав, выяснили, что на первом этапе построенного графика проект будет выполнен за $T_{кр}$ = 91 день и потребует расходов в S = 91 × 11 = 1001 ден. ед. (рис. 3). Теперь попробуем оптимизировать данный проект.

Существует способ, позволяющий сделать затраты на выполнение проекта еще меньше, чем на графике представленным на рисунке 5, уменьшив при этом время его выполнения. Метод предполагает поэтапное снижение интенсивности работ на критических пути [1][10]. На данном этапе обозначим значение времени резервной дуги, её путь, как:

$$T_{р} = \sum(a_n \in T_{р}). \quad (2)$$

Значение времени критического пути так же выразим формулой:

$$T_{кр} = \sum(a_n \in T_{кр}). \quad (3)$$

Тогда формула нахождения резерва будет иметь вид:

$$R = T_{кп} - T_{р}. \quad (4)$$

$T_{кп}$ – это суммарное время работ дуги параллельной любой некритической дуге, $T_{кп} \in T_{кр}$. То есть это время части или же всего критического пути. Пример $T_{кп}$ резерва R (1, 3, 5, 6):

Начальное и конечное события в этом резерве – это события 1 и 6, значит и работы при расчетах резерва мы будем брать только между этими событиями. $T_{кп}$ и $T_{р}$ этого резерва будут рассчитываться так:

$T_{р}$ = a3+a5+a7,

$T_{кп}$ = a2+a4+a6.

Теперь найдем резервы некритических дуг:

R (1, 3, 5, 6) = (22 + 30 + 15) – (8 + 16 + 13) = 67 – 37 = 30,

R (1, 3, 5, 7, 8) = (22 + 30 + 15 + 6) – (8 + 16 + 8 + 7) = 73 - 39 = 34,

R (1, 3, 5, 8) = (22 + 30 + 15 + 6) – (8 + 16 + 9) = 73 - 33 = 40.

Наименьший резерв имеет дуга R (1, 3, 5, 6) равный 30 дням.

Попробуем сократить работу на нашем критическом пути. Пусть Δ_k – величина стоимости сокращения проекта на один день. Тогда $\Delta_k = S - S_k$, где S – это стоимость одного дня проекта, а S_k – стоимость сокращения длительности работы на 1 день. По условию задачи S = 11 ден. ед. Так же введём величину t_k^c – максимальное количество дней, на которое может быть сокращена работа:

$$t_k^c = t_{max} - t_{min}. \quad (5)$$

Таблица 2 – Расчёт резервов сетевой задачи на первом этапе

Работа	t_{max}	t_{min}	S_k	$\Delta_k = S - S_k$	t_k^c	$t_k^c \times \Delta_k$
а2	22	6	10	11 - 10 = 1	16	16
а4	30	9	10	11 - 10 = 1	21	21
а6	15	3	9	11 - 9 = 2	12	24

Выгоднее всего сократить работу а6, сократив её на 12 дней мы сократим стоимость проекта на 24 ден. ед., но у нас останется еще 30 - 12 = 18 дней на сокращение других работ. Далее, сократим работу а4 на 18 дней получим сокращение стоимости проекта на 18 ден. ед.

Ниже представлен второй этап сетевого графика после перерасчета его данных, рисунок 6.

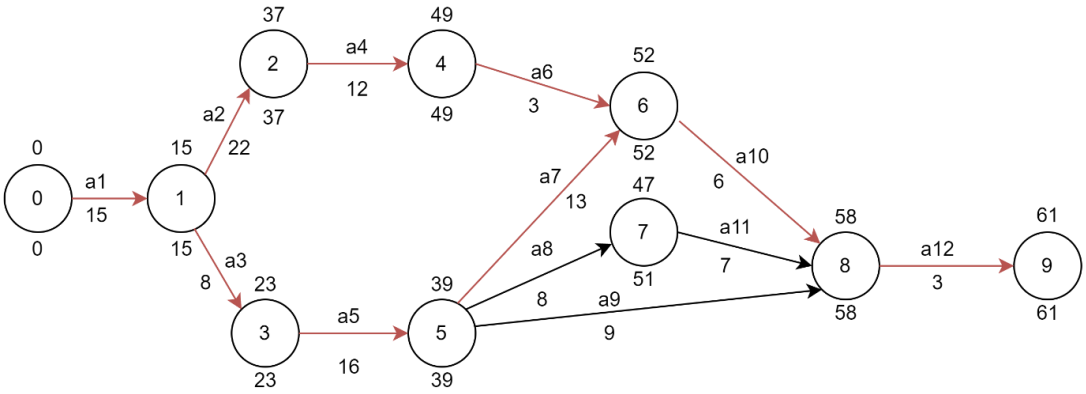

Рисунок 6 – Второй этап

Итого общее сокращение стоимости проекта составит 24 + 18 = 42 ден. ед. Критическое время на этом этапе составит $T_{кр} = 61$ день, а его стоимость будет равной 61 × 11 + (12×9 + 18×10) = 671 + (108+180) = 671 + 288 = 959 ден. ед.

Следующий наименьший резерв имеет дуга R (5, 7, 8) равный (a6+a10) -(a8+a11) = (13+6) -(8+7) =19-15=4 дням. Также продолжаем сокращать работы критического пути:

Таблица 3 - Расчёт резервов сетевой задачи на втором этапе

Работа	t_{max}	t_{min}	S_k	$\Delta_k = S - S_k$	t_k^c	$t_k^c \times \Delta_k$
a7	13	3	8	11 - 8 = 3	10	30
a10	6	1	4	11 - 4 = 7	5	35

Выгоднее всего сократить работу a10, сократив её на 4 дней мы сократим стоимость проекта на 4×7=28 ден. ед. Критическое на этом этапе составит $T_{кр} = 57$ дней, а его стоимость будет равной 57 × 11 + (12×9 + 18×10 + 4×4) = 627 + (108+180+16) = 627 + 304 = 931 ден. ед. Для проверки можно посчитать и по-другому: 1001 - (42 +28) = 931 ден. ед.

Конечная модель оптимизированного сетевого графика представлена на рисунке 7.

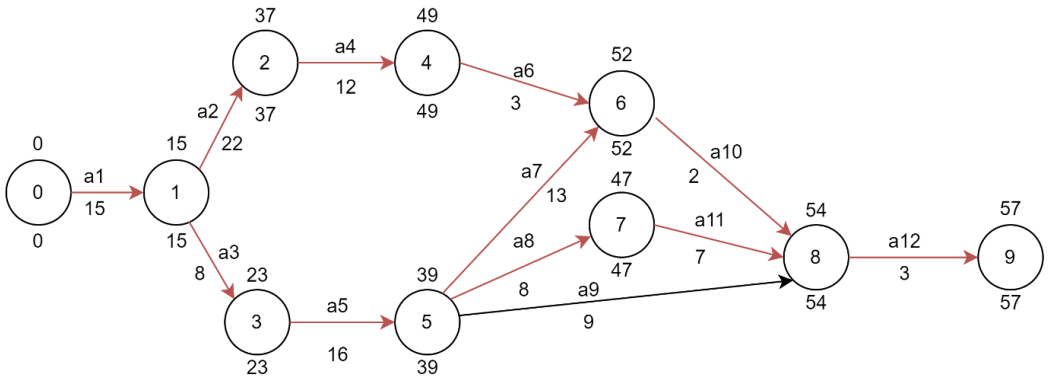

Рисунок 7 – Оптимизированный сетевой график

Дальнейшее сокращение работ с возможным уменьшением стоимости проекта невозможно, так как последний резерв опирается лишь на часть критической дуги (не получится сократить работу а10 на 9 дней), и для уменьшения критического пути придется сокращать работы всех параллельных критических путей, а их сокращение вместе потребует дополнительных вложений денежных средств [17]. Итак, получен оптимальный по стоимости вариант выполнения проекта (рис. 7).

Стоимость проекта этого графика составляет 931 ден. ед. А сроки выполнения составляют 57 рабочих дней.

Для сравнения найдем вариант стоимости проекта при выполнении всех работ с максимальной интенсивностью, рисунок 8.

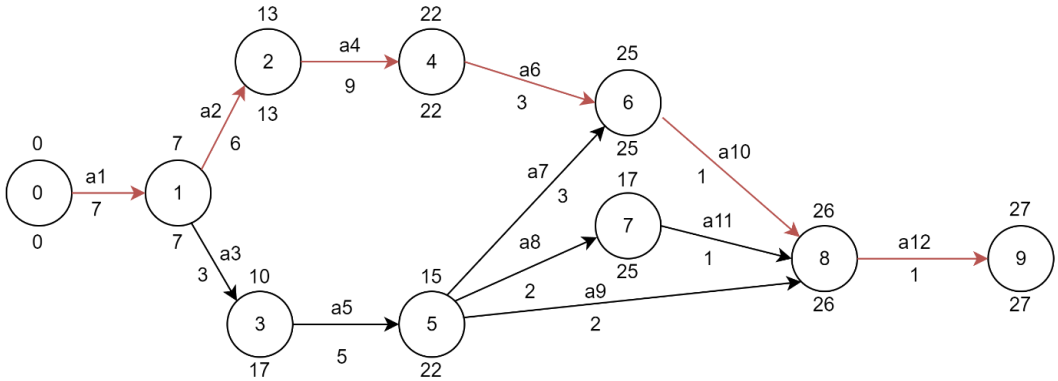

Рисунок 8 – Сетевой график с максимальной интенсивностью проведения работ

Проект будет выполнен за 27 дней при стоимости 27 × 11 + 908 = 297+908 =1205 ден. ед. Так выглядит сетевой график этого проекта, (рис. 8). Число «908», назовём его $S_{\text{доп}}$ – это стоимость всех сокращений, и оно вычисляется по формуле:

$$S_{\text{доп}} = \sum((t_{max}(a_n) - t_{min}(a_n)) \times S_k(a_n)). \quad (6)$$

Таблица 4 – Расчет данных для подсчета дополнительной стоимости проекта.

Обозначение a_n	t_{max}	t_{min}	$t_{\text{сокр}}$	Сокращение 1 дня работы S_k	$t_{\text{сокр}} \times S_k$
a_1	15	7	8	9	72
a_2	22	6	16	10	160
a_3	8	3	5	6	30
a_4	30	9	21	10	210
a_5	16	5	11	11	121
a_6	15	3	12	9	108
a_7	13	3		8	80
a_8	8	2	6	6	36
a_9	9	2	7	7	49
a_{10}	6	1	5	4	20
a_{11}	7	1	6	3	18
a_{12}	3	1	2	2	4

Данные для его подсчёта следует брать из таблицы расчетных данных (см. таблица 4), где $t_{\text{сокр}}$ – это разница максимальной и минимальной интенсивности работ:

$$t_{\text{сокр}} = t_{max} - t_{min}, \quad (7)$$

$S_{\text{доп}}$ =72+160+30+210+121+108+80+36+49+20+18+4 = 908.

Сравнив его стоимость со стоимостью оптимизированного сетевого графика (рис 7), видно, что стоимость варианта с максимальной интенсивностью работ самая большая и соответственно не выгодная для проекта. Поэтому попробуем оптимизировать этот вариант, убрав сокращение работ на некритических путях.

В первую очередь при этом уменьшать сокращение надо для тех работ, у которых стоимость сокращения больше, поскольку отмена их сокращения по времени даст может дать максимальное уменьшение стоимости проекта. Будем действовать так называемым «обратным ходом». Рассмотрим возникшие некритические дуги после максимального сокращения всех работ. Найдем их резервы.

R (1, 3, 5, 6) = (6+9+3) - (3+5+3) = 18 – 11 = 7,

R (1, 3, 5, 7, 8) = (6+9+3+1) - (3+5+2+1) = 19 – 11 = 8,

R (1, 3, 5, 8) = (6+9+3+1) - (3+5+2) = 19 – 10 = 9.

При решении задачи таким способом, критическое время должно оставаться неизменным, т.е. равным 27 дням. Это означает, что все работы, лежащие на критическом пути, должны остаются неизменными. Их продолжительности сокращены до минимально возможного предела и не подлежат дальнейшему изменению. В данной ситуации следует увеличивать продолжительности работ, которые находятся на некритических дугах, (отменить сокращение), составляя как можно больше параллельных дуг к критическому пути. Чтобы это сделать максимально выгодно для проекта, нужно выбрать максимально выгодные комбинации работ, сроки выполнения которых мы будем одновременно увеличивать. У нас имеется три параллельных некритических дуги: (1, 3, 5, 6), (1, 3, 5, 7, 8), (1, 3, 5, 8). Резервы их времени составляют 7, 8 и 9 соответственно. Самый наименьший резерв имеет дуга (1, 3, 5, 6) равный 7 дням. Для одновременного увеличения этих дуг на 7 дней необходимо выбрать работы на дугах так, чтобы суммарная стоимость сокращения этих работ была максимальной.

Необходимо выбирать их из работ некритических дуг: а3, а5, а7, а8, а9, а11. Наибольшую стоимость сокращения имеет работа а5 равную 11 ден. ед. Она находится на всех некритических дугах и имеет запас сокращения 11 дней. Отменив её сокращение на 7 дней, мы сократим стоимость проекта на 11×7=77 ден. ед. Но при таком сокращении придется сокращать и другие работы одновременно с а5 так как лишь её сокращение не даст нам появления нового критического пути.

Найдем наиболее выгодную комбинацию с помощью данных, представленных в таблице 5 и 6. При подсчёте данных всегда первые работы в уравнении увеличиваем до максимума, при этом не превышая количество дней на критическом пути. Для этого будем рассматривать комбинации работ, увеличивая их до предела, начиная с первой, лежащий на некритическом пути, и переходя к следующей проделывая тоже самое, до тех пор, пока не достигнем их лимита, который не должен превышать критическое время в 27 дней.

Стоимость сокращения комбинаций работ выразим формулой:

$$S_{\text{сокр}} = \left(S_k(a_n) \times \max\left(t_{\text{сокр}}(a_n)\right) + S_k(a_{n+1}) \times \max\left(t_{\text{сокр}}(a_{n+1})\right) \dots\right). \quad (8)$$

Эта формула позволит нам найти все возможные комбинации сокращения работ, при условии, что отмена их сокращений не приведёт к увеличению $T_{\text{кр}} = 27$ дней. Так же стоит взять во внимание то, что a_n берётся исключительно из таблицы работ некритических дуг и их стоимости.

Таблица 5 - Работы некритических дуг и их стоимость.

Дуга	(1, 3, 5, 6)			(1, 3, 5, 7, 8)				(1, 3, 5, 8)		
Работа a_n	а3	а5	а7	а3	а5	а8	а11	а3	а5	а9
Стоимость S_k	6	11	8	6	11	6	3	6	11	3

Таблица 6 – Комбинации работ для сокращения стоимости проекта.

Работы	Стоимость сокращения $S_{сокр}$
a3+a5+a8+a9	5×6+2×11+1×6+2×7=72
a3+a5+a11+a9	5×6+2×11+1×3+2×7=69
a3+a7+a8+a9	5×6+2×8+3×6+4×7=92
a3+a7+a11+a9	5×6+2×8+3×3+4×7=83
a5+a8+a9	7×11+1×6+2×7=97
a5+a11+a9	7×11+1×3+2×7=94
a7+a8+a11+a9	7×8+6×6+2×3+7×7=147
a7+a11+a8+a9	7×8+6×3+2×6+7×7=135
a8+a3+a7+a9	6×6+2×6+5×8+7×7=137
a8+a5+a7+a9	6×6+2×11+5×8+7×7=147
a8+a11+a7+a9	6×6+2×3+5×8+7×7=131
a9+a3+a7+a8	7×7+2×6+5×8+6×6=137
a9+a3+a7+a11	7×7+2×6+5×8+6×3=115
a9+a5+a7+a8	7×7+2×6+11×8+6×6=147
a9+a5+a7+a11	7×7+2×11+5×8+6×3=125
a11+a3+a7+a9	4×3+2×6+5×8+7×7=113
a11+a5+a7+a9	4×3+2×11+5×8+7×7=123
a11+a8+a7+a9	4×3+2×6+5×8+7×7=113

Из таблицы мы получили три одинаково максиминах сокращения стоимости, два из которых имеют критический путь на всех дугах. Третий способ сокращения, а именно a7, a8, a11 и a9 дает нам критический путь везде, кроме дуги (5, 8) (рис. 9).

Однако при попытках сделать из него оптимальный вариант сокращения, мы получим тот же результат, как если бы мы сначала сокращали работы a8 и a9, при этом сумма сокращения стоимости не изменится.

Изменённый сетевой график второго варианта изображён на рисунке 9.

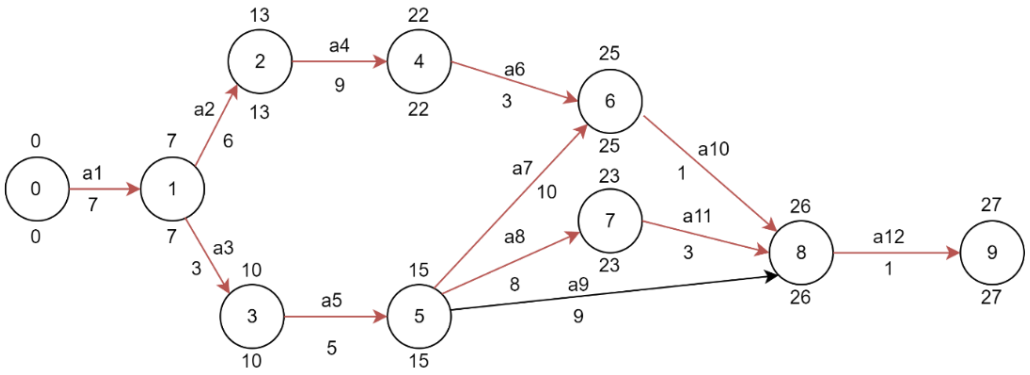

Рисунок 9 – Изменённый сетевой график второго варианта

Поэтому просто выберем последний вариант с максимально большим сокращением стоимости. Это будет сокращения работ a9+a5+a7+a8 (рис. 9). При использовании этого варианта графика мы имеем продолжительность работ 27 дней, но при этом общая её стоимость будет составлять 1205-147=1058 ден. ед.

Ниже представлена модель оптимизированного сетевого график второго варианта, рисунок 10.

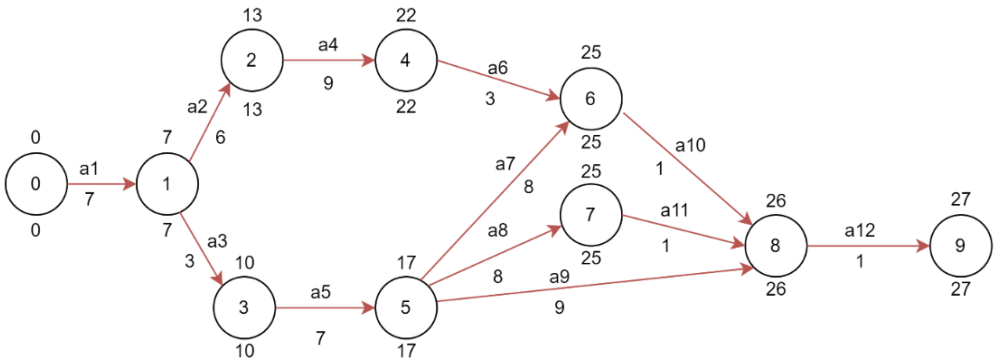

Рисунок 10 – Оптимизированный сетевой график второго варианта

Таким образом мы получаем два варианта сокращения времени и стоимости данного проекта:

– первый вариант сокращает затраченное время проекта с 91 до 57 дня при этом уменьшая его стоимость с 1001 ден. ед. до 931 ден. ед;

– второй же высчитывает минимальный срок проведения проекта: 27 дней (это на 30 дней меньше, чем первый), и уменьшает его стоимость с 1205 до 1058 ден. ед. за счет отмены сокращения наилучшей комбинации работ.

Таблица 7 – Данные двух вариантов решения сетевого проекта.

Способы оптимизации	Начальная стоимость графика	Оптимизированная стоимость графика	Время выполнения проекта
Первый	1001 ден. ед.	931 ден. ед.	57 дней
Второй	1205 ден. ед.	1058 ден. ед.	27 дней

Оба эти варианта сокращают денежные затраты проекта, однако время уменьшается только в первом варианте решения. Но несмотря на это оба метода решения имеют права на существование поскольку оба оптимизируют стоимость проекта, но на разных значениях по времени. И только сам заказчик проекта должен решить, какой способ подходит ему больше для решения своего проекта.

2.2 Анализ алгоритма оптимизации стоимости сетевого проекта

Для того чтобы проанализировать практичность методов решения сетевой задачи, построим график, демонстрирующий результаты методов для их сравнения, рисунок 11.

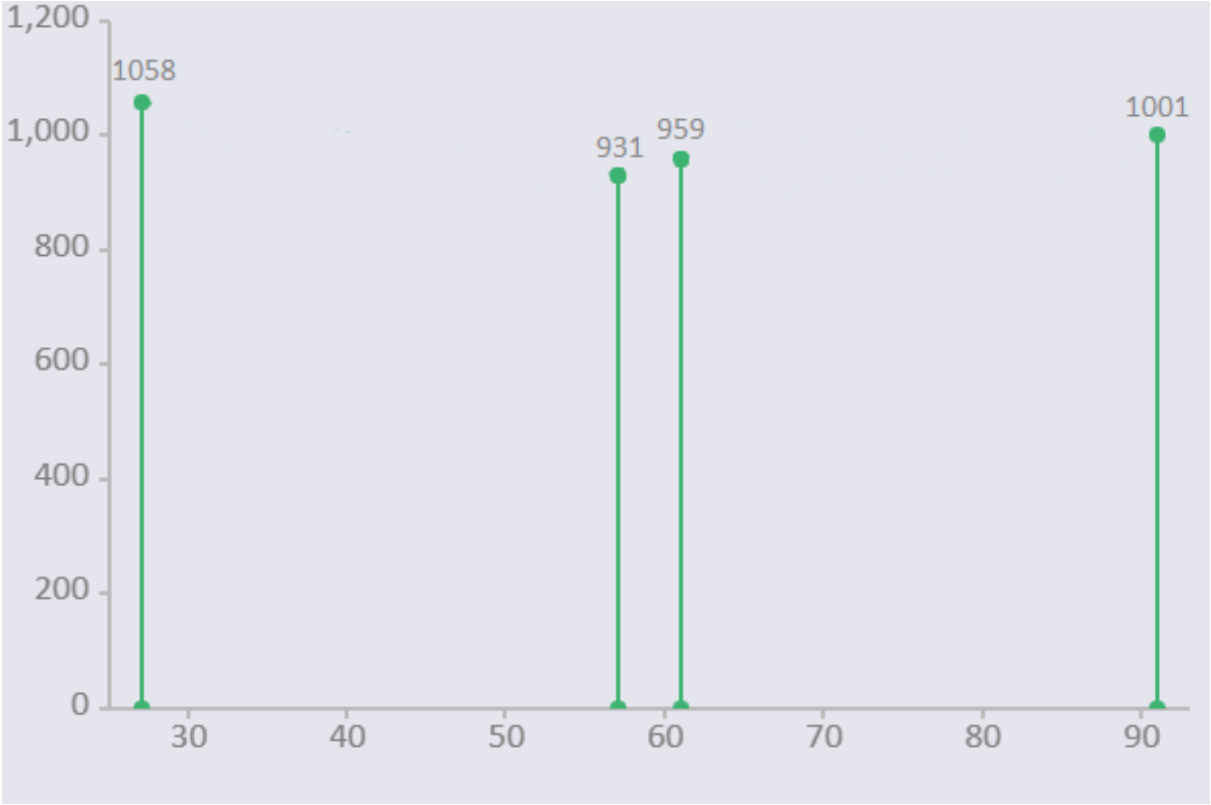

Рисунок 11 – график сравнения результатов

На графике представлены соотношения времени и стоимости всех вариантов, полученных в ходе решения сетевой задачи. Координата X – время выполнения проекта в сутках, Y – стоимость выполнения проекта в ден. ед. Как мы видим, самый оптимальный по стоимости это вариант с соотношением стоимость/врем = 931/57. Стоимость одного дня этого варианта ровна $S_{\text{дня}} = \frac{931}{57} = 16{,}3$ ден.ед. Второй метод демонстрирует кратчайший срок выполнения проекта и имеет соотношение стоимость/врем = 1058/27. Стоимость его дня такова: $S_{\text{дня}} = \frac{1058}{27} = 39{,}2$ ден. ед.

Как можно увидеть, каждый из вариантов решения задачи имеет преимущество хотя бы в одном критерии. Однако, задачей выпускной квалификационной работы является оптимизация стоимости проекта, поэтому наиболее подходящий вариант дает первый способ решения, при котором стоимость выполнения проекта минимальна (S = 931 ден. ед).

Вывод к главе 2

Вторая глава посвящена разработке алгоритма оптимизации стоимости сетевого проекта.

Представлено два варианта реализации алгоритма оптимизации. Каждый из вариантов оптимизации осуществляется в два этапа.

Для первого варианта строится сетевой график проекта с минимальной интенсивностью проведения всех работ, определяется его критический путь и топология пути. После чего, выполняется оптимизация сетевого графика по стоимости за счет сокращения длительности работ, лежащих на критическом пути.

Для второго варианта строится сетевой график проекта с максимальной интенсивностью проведения всех работ, определяется его критический путь и топология пути. После чего, выполняется оптимизация сетевого графика за счет отмены сокращения работ, лежащих на некритическом пути.

Полученные решения анализируются и выбирается то, которое больше устраивает заказчика из соотношения стоимости и времени выполнения проекта. В рассмотренной в работе задаче наименьшую стоимость дает первый вариант решений.

Глава 3. Разработка и анализ программы для оптимизации стоимости сетевых проектов

3.1 Дополнительное программное обеспечение для разработки алгоритма программы

Реализуем в виде программного кода алгоритмы, представленные во второй главе. Предварительно рассмотрим вспомогательные программы и сервисы, с помощью которых было решена сетевая задача.

Для построения сетевого графика проекта воспользуемся уже готовым ПО, которое позволит нам справится с этим гораздо быстрее, чем рисовать его вручную. Одной из таких программ является браузерное приложение Draw.io.

3.1.1 Онлайн приложение для построения графиков Draw.io

«Draw.io – это бесплатное онлайн-приложение для создания диаграмм для рабочих процессов, BPM, организационных, сетевых диаграмм.»

Главной особенностью этого приложения является его легкодоступность. Его можно использовать имея любое устройство, обладающее браузером. Приложения обладает большим количеством разных инструментов и функций, позволяющих строить различные диаграммы, блок-схемы и, что для нас и важно, сетевые графики. После построения любой наглядной графической модели, её можно спокойно сохранить в любом удобном для вас формате для дальнейшего использования [25].

Наглядный пример построенной графической модели в онлайн-сервисе draw.io представлен на рисунке 12.

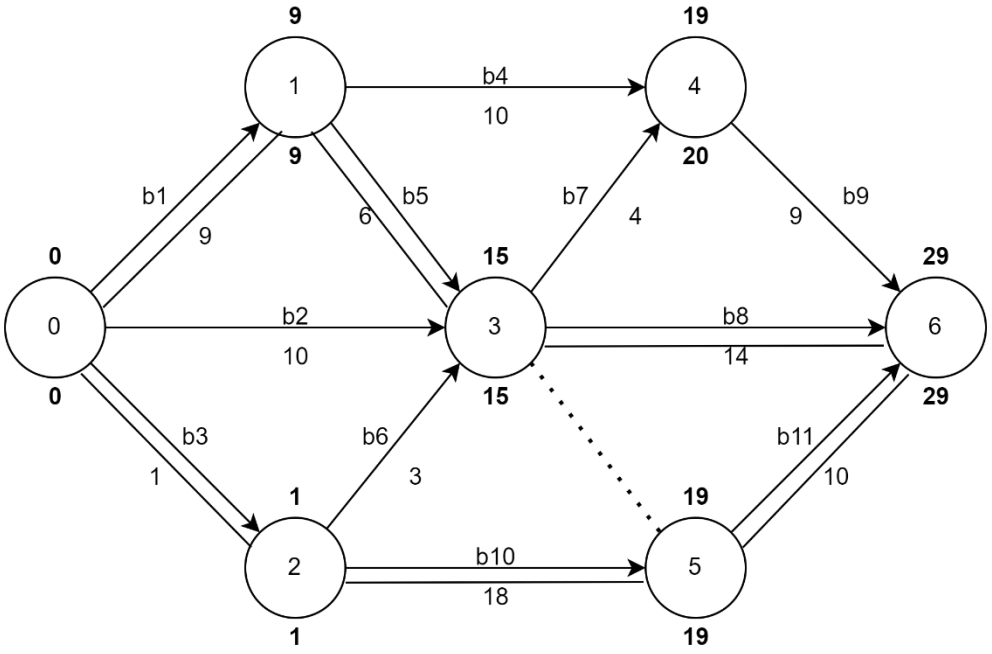

Рисунок 12 – построенный сетевой график

Так же все графики, используемые во второй главе для решения сетевой задачи, были реализованы в этом приложении. Конечно, существуют более удобные аналоги данной среды с удобным интерфейсом и дизайном и естественно браузерное приложение не сравниться с ними в этом, но они все являются платными. Эта и является ещё одним из главных преимуществ draw.io.

3.1.2 Браузерный-калькулятор «Сетевой график онлайн»

Браузерный-калькулятор «Сетевой график онлайн» таже позволяет строить сетевые графики, но основной его задачей является решения различных математических задач.

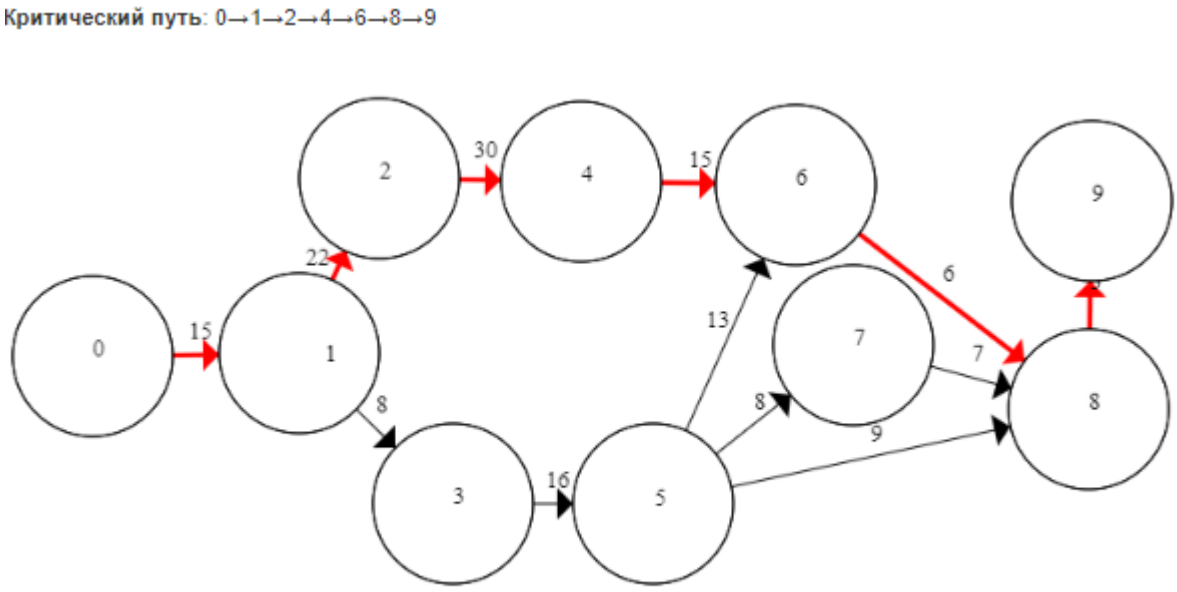

Рисунок 13 – график, построенный калькулятором

К примеру, он может с лёгкостью рассчитать критический путь сетевого графика по его входным данным (рис. 13). И эта лишь малая часть его возможностей. Сетевой график онлайн, как уже было сказано, может решать разные информационные и математические задачи, с поэтапным решением которых можно ознакомиться сразу после вывода ответа с описанием всех действий и используемых формул. Так же здесь присутствует много подсказок в использовании приложения для начинающих и имеются видеоуроки по составлению задач [14].

Однако даже его возможностей не хватит для решения оптимизации сетевой задачи, поскольку наш алгоритм не ограничивается нахождением критического пути.

Номер события	Сроки свершения события: ранний $t^{р}(i)$	Сроки свершения события: поздний $t^{п}(i)$
0	0	0
1	15	15
2	37	37
3	23	53
4	67	67
5	39	69
6	82	82
7	47	81
8	88	88
9	91	91

Рисунок 14 – расчёт срока событий

При вводе данных задачи, рассмотренной во второй главе, калькулятор смог справится лишь с нахождением критического времени и расчетом срока совершения событий (рис. 13,14).

3.2 Разработка программы по оптимизации стоимости сетевого проекта

Для реализации программного кода была использована среда Microsoft Visual Studio. Это среда разработки представляет собой многофункциональную программу для написания различный консольных приложений, графических программ и веб-сайтов. Она включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и как отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор

форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и схем базы данных [8][13][15].

В комплект Visual Studio входят следующие основные компоненты:

– Visual Basic.NET - для разработки приложений на VisualBasic;

– Visual C++ - на традиционном языке C++;

– Visual C# - на языке C# (Microsoft);

– Visual F# - на F# (Microsoft Developer Division).

Главные преимущества среды Microsoft Visual Studio:

– Поддержка большого количества языков программирования;

– Интуитивный стиль кодирования;

– Контроль выполнения и отладка программы;

– Безопасность среды;

– Быстрая скорость разработки кода.

К недостатком же данной среды можно отнести следующее:

– Программа имеет большой вес для хранения данный;

– Для комфортной работы с ней необходимо иметь хорошие по мощности устройства;

– Труднодоступность в бесплатном её использовании;

– Долгий процесс обучения для использования среды.

Ниже, на рисунке 15, представлена среда разработки Visual Studio.

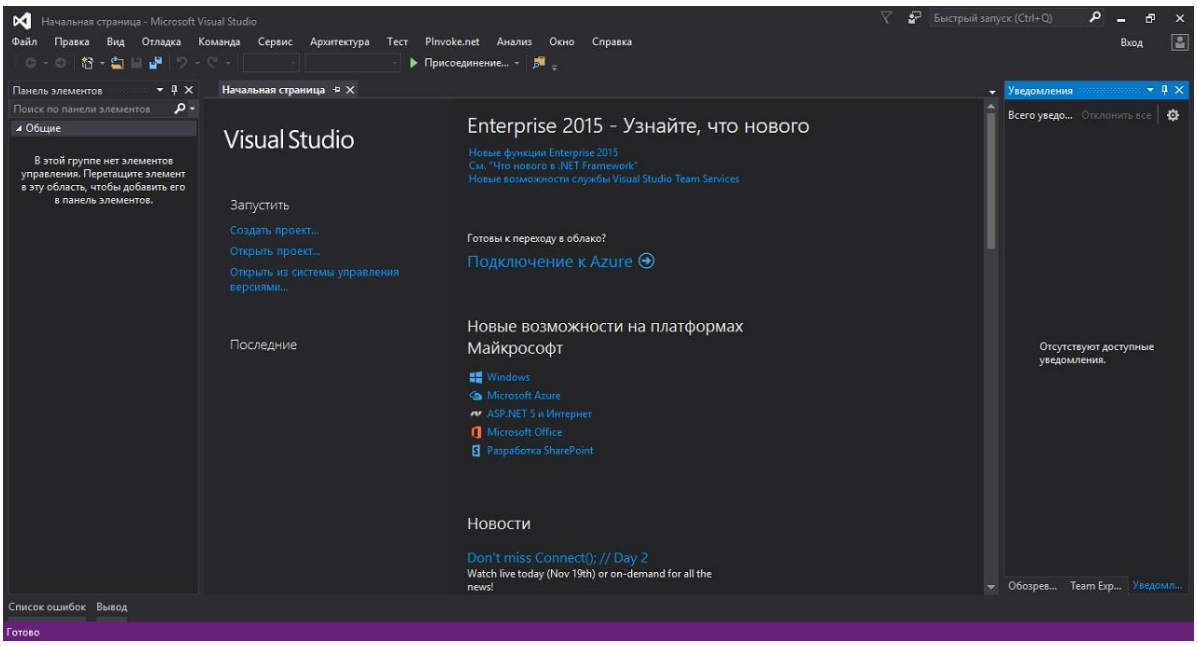

Рисунок 15 – среда разработки Visual Studio

Сама программа была написана на языке C++. Этот язык является одним из самых популярных языков в программировании. Его можно использовать как для написания простых программ, так и для сложных операционных систем. Данным языком можно реализовывать как в бесплатных целях, так и для различных коммерческих платформ [20][21][22][26].

«C++ завоевал большую популярность, чем C, потому что что стал мощным инструментом для разработки сложного программного обеспечения. Это послужило причиной для того, чтобы не рассматривать изучение C как обязательное перед изучением C++ [4].»

Итак, приступим к непосредственной реализации программы. Для начала создадим новый проект (рисунок 16):

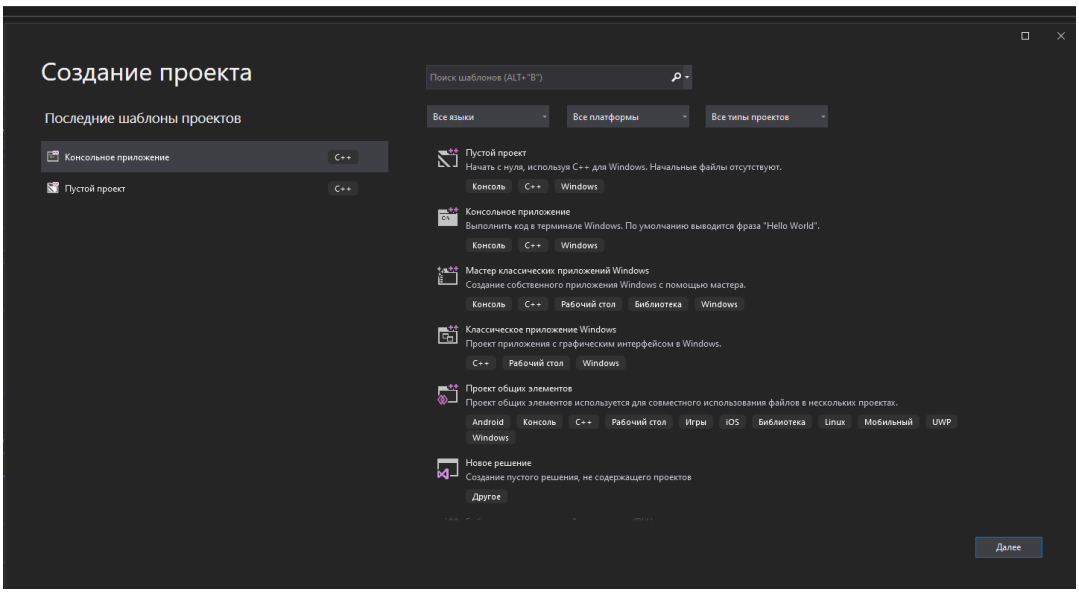

Рисунок 16 – Создание проекта

Подключив все необходимые библиотеки среды, приступаем к написанию основных функций программы (рисунок 17):

```
void fillMass(int** info, int n, string* paths, int days, int price, string mainWay)
{
    //-----

    int** graph = new int* [n];
    for (int i = 0; i < n; i++)
        graph[i] = new int[n];

    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            graph[i][j] = 0;

    for (int i = n - 1; i >= 0; i--)
    {
        int point = 0, mark = 1;

        if (paths[i].length() < 2)
        {
            point = atoi(paths[i].c_str());

            if (point > 0)
                graph[point - 1][i] = mark++;
        }
        else
        {
            for (int j = 0; j < paths[i].length(); j++)
                if (paths[i][j] != ',')
```

Рисунок 17 – Функция fillMass

Функция fillMass является реализацией первого варианта оптимизации задачи. Она содержит в себе матрицу смежности, отражающую пути между работами. Пример кода представлен в Приложении А.

По этой матрице находятся резервные пути, разделяются и рассматриваются по отдельности. Проверяется критерий оптимизации, и в случае выполнения критерия, работы на критическом пути оптимизируются, пересчитываются время и цена работ. То есть, она проделывает все те же операции, которые были рассмотрены во второй главе при решении сетевой задачи [23][24].

Начало программного кода функции первого варианта оптимизации задачи (рисунок 18):

```
void SecondMethod(int** info, int n, string* paths, int price, string mainWay)
//void SecondMethod(int n, string* paths, int price, string mainWay)
{
    cout << "\n\nВторой метод\n";

    int days = 0, reduction = 0, defeat = 0, max = 0, a = 0, b = 0, c = 0, d = 0, tA = 0, tB = 0, tC = 0, tD = 0, tP = 24;

    cout << "\nКоличество дней: ";
    cin >> days;

    for (int i = 0; i < n; i++)
        reduction += (info[i][0] - info[i][1]) * info[i][2];

    cout << "\nСтоимость " << price * days + reduction << endl;

    int r[] = { 3, 5, 7, 8, 9, 11 };
    int p[] = { 17, 22, 25, 25, 26, 26 };

    for (int i = 0; i < 6; i++)
        for (int j = 0; j < 6; j++)
            for (int k = 5; k >= 0; k--)
                for (int l = 5; l >= 0; l--)
```

Рисунок 18 – Функция SecondMetod

Функция SecondMetod реализует второй способ оптимизации задачи, так называемый «обратный ход». Он основан на фиксации значения дней для критического пути, но с уменьшением стоимости работ, которые лежат на резервных дугах. Иными словами, она так же выбирает оптимальную комбинацию тех работ, которые при уменьшении дают нам максимальное сокращения стоимости проекта.

Теперь приступим к запуску программы (рисунок 19):

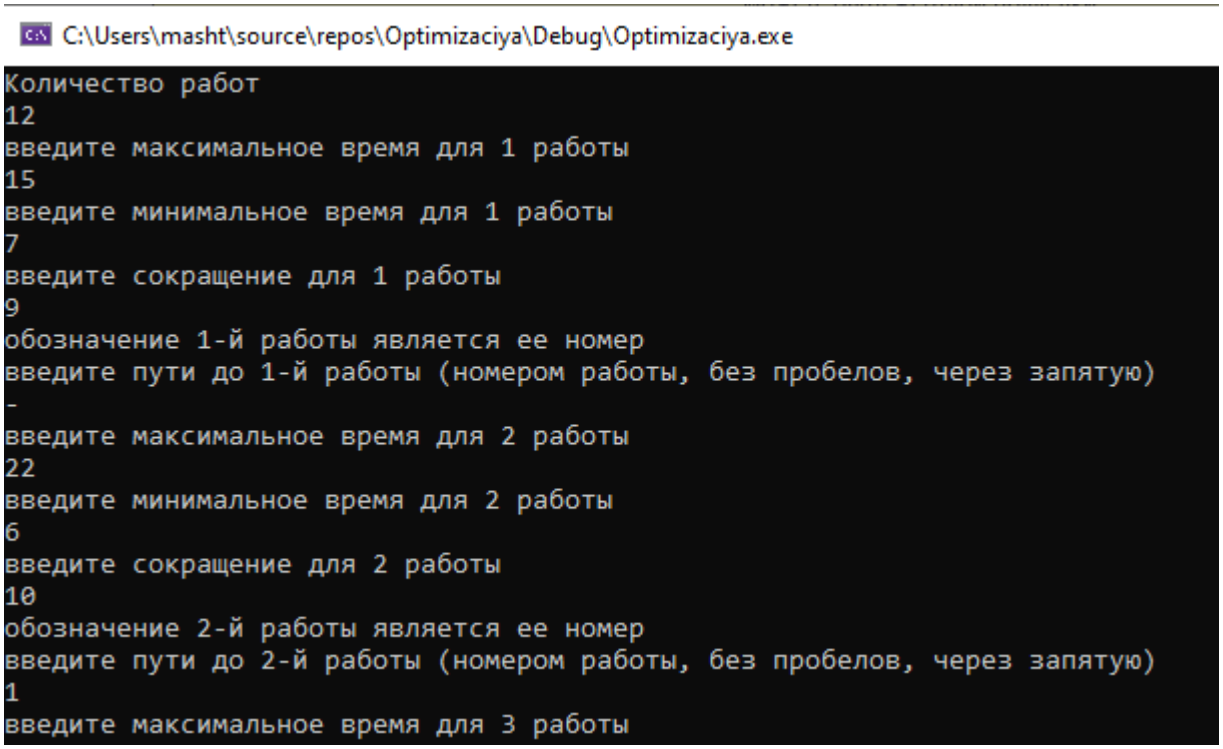

Рисунок 19 – Запуск программы

При запуске, программа просит нас ввести входные данные для расчета самой задачи, а именно: количество дней, период их длительности и их стоимость, а также предшествующие им работы. После ввода данных работ, нужно ввести стоимость одного рабочего дня проекта, и программа приступит к расчёту оптимизации.

Данные для ввода в текстовом формате представлены на рисунке 20.

данные.txt – Блокнот

Файл Правка Формат Вид Справка

Номер работы n	Обозначение a(n)	Путь работы	t_max	t_min	Сокращение одного дня работы
1	1	0	15	7	9
2	2	1	22	6	10
3	3	1	8	3	6
4	4	2	30	9	10
5	5	3	16	5	11
6	6	4	15	3	9
7	7	5	13	3	8
8	8	5	8	2	6
9	9	5	9	2	7
10	10	6,7	6	1	4
11	11	8	7	1	3
12	12	9,10,11	3	1	2

Рисунок 20 – Исходные данные в текстовом файле

Данные также можно ввести в текстовом формате по образцу, показанном на рисунке 20. Достаточно просто указать в консоли название txt файла с данными, а затем также ввести стоимость одного рабочего дня проекта.

Данные на выходе программы представлены на рисунке 21.

```
Критический путь 1,2,4,6,10,12

количество дней 91

Расходы при обычном варианте: 1001

Наименьший резерв 30 по пути 1,3,5,7,10,12

Сократили работ 6 на 12 и 4 на 18 дней

Стоимость проекта 959 дней 61

Наименьший резерв 4 по пути 1,3,5,8,11,12

Сократили работ 10 на 4 дней

Стоимость проекта 931 дней 57


Второй метод

Стоимость 1205

147

Второй вариант стоимости проекта 1058 дней 27
```

Рисунок 21 – Вывод программы в консоли

Как мы видим, на выходе программа также продемонстрировала нам два варианта стоимости проекта за разные промежутки времени, и они соответствуют нашим данным, которые были рассчитаны во второй главе. Это означает что программа успешно справилась с решением данной задачи.

Выводы к главе 3

Третья глава ВКР посвящена разработке программы алгоритма по оптимизации стоимости сетевого проекта. Результаты проделанной работы позволили сделать следующие выводы:

– в качестве среды для написания программы была выбрана Microsoft Visual Studio;

– для реализации алгоритма был использован язык программирования C++;

– разработанная программа позволила успешно реализовать решение задачи оптимизации стоимости сетевого проекта.

Заключение

Выпускная квалификационная работа была посвящена проблеме анализа и реализации алгоритма оптимизации стоимости сетевых проектов.

В процессе работ над ВКР решены следующие задачи:

– были изучены и проанализированы методы решения сетевых задач;

– разработан алгоритм по оптимизации стоимости сетевых задач;

– был написан программный код, основанный на алгоритме оптимизации стоимости сетевых проектов.

Все задачи в выпускной квалификационной работе были выполнены.

В первой главе представлено математическое обоснование методов решения по оптимизации стоимости сетевых проектов и их анализ.

Во второй главе была решена сетевая задача на оптимизацию стоимости сетевого проекта. Разработанный на основе метода критического пути алгоритм решает задачу в два этапа и представляет два варианта её решения.

При первом варианте решения строится сетевой график проекта с минимальной интенсивностью проведения работ, определяется его критический путь и топология пути. Затем выполняется оптимизация сетевого графика по стоимости за счет сокращения длительности работ, лежащих на критическом пути.

При втором варианте решения строится сетевой график проекта с максимальной интенсивностью проведения всех работ, определяется его критический путь и топология пути. Оптимизация сетевого графика выполняется за счет отмены сокращения работ, лежащих на некритическом пути.

В третьей главе была разработана программа для решения оптимизации стоимости сетевой задачи по алгоритму, представленному во второй главе. Проведён обзор сервисных инструментов для реализации программного

приложения. Для разработки программы была использована среда Microsoft Visual Studio и язык программирования C++.

Список используемой литературы и используемых источников

1. Алгоритм оптимизации сетевого графика по стоимости [Электронный ресурс]. URL: https://studwood.net/1992083/matematika_himiya_fizika/algoritm_optimizatsii_setevogo_grafika_stoimosti
2. Анализ и оптимизация сетевого графика [Электронный ресурс]. URL: https://lektsii.org/8-29566.html
3. Илларионов В.А. Сетевое планирование: Казань 2013. 45 с.
4. Лафоре.Р. Объектно-ориентированное программирование в C++ 4 издание. 2004. – 924 с.
5. Метод PERT [Электронный ресурс]. URL: https://forpm.ru/метод-pert/
6. Москинова Г. И. Дискретная математика. Математика для менеджера в примерах и упражнениях: учебное пособие / Г. И. Москинова. М.: Логос, 2000.
7. Обзор и анализ существующих методов решения задачи. Обоснование выбора метода решения или разработки нового. [Электронный ресурс]. URL: https://studwood.net/1841083/informatika/obzor_analiz_suschestvuyuschih_metodov_resheniya_zadachi_obosnovanie_vybora_metoda_resheniya_razrabotki_novogo
8. Описание среды разработки Microsoft Visual Studio [Электронный ресурс]. URL: https://studbooks.net/2258619/informatika/opisanie_sredy_razrabotki_microsoft_visual_studio
9. Оптимизация стоимости сетевого проекта [Электронный ресурс]. URL:

http://www.inf.tsu.ru/library/DiplomaWorks/CompScience/2007/marenkov/diplom.pdf

10. Оптимизация сетевого графика методом «время – стоимость» [Электронный ресурс]. URL: https://studme.org/80821/ekonomika/optimizatsiya_setevogo_grafika_metodom_vremya_stoimost

11. Плескунов М. А. Задачи сетевого планирования: учебное пособие. Екатеринбург: Изд-во Урал. ун-та, 2014. 92 с.

12. Редькин Н. П. Дискретная математика / Н. П. Редькин. СПб: Лань, 2003.

13. Руководство по Visual Studio [Электронный ресурс]. URL: https://docs.microsoft.com/ru-ru/cpp/get-started/?view=msvc-170

14. Сетевой график онлайн [Электронный ресурс]. URL: https://www.semestr.online/graph/network.php

15. Сидорина Т. Visual Studio C++ - учебное пособие. Санкт-Петербург, 2009. 843 с.

16. Сосина Н.А. Исследование операций/Электронное учебное пособие. В 2-х частях. Часть II/Тольятти: ФГБОУ ВО «Тольяттинский государственный университет», 2022г. Электронный ресурс.

17. Спиридонов Э.С., Телятникова Н. А. Сетевое планирование в строительстве: Учебное пособие, М.: РУТ (МИИТ), 2018. 75 с.

18. Таирова Е.В. Методы сетевого планирования в организации комплексов работ: учебное пособие / Е.В. Таирова. – Иркутск: ИрГУПС, 2007. – 95 с.

19. Управление проектом на основе сетевых моделей: Метод. указания / Самар. гос. аэрокосм. ун-т, Сост. И.Г.Абрамова. Самара, 2007. 58 с.

20. Creating Standard C++ program [Электронный ресурс]. URL: https://docs.microsoft.com/en-us/cpp/windows/walkthrough-creating-a-standard-cpp-program-cpp?view=msvc-170

21. C++ in Visual Studio [Электронный ресурс]. URL: https://docs.microsoft.com/en-us/cpp/overview/visual-cpp-in-visual-studio?view=msvc-170

22. C++ solved programming programs [Электронный ресурс]. URL: https://www.includehelp.com/cpp-programming-examples-solved-cpp-programs.aspx

23. Graph Emplementation in C++ Using Adjacency List [Электронный ресурс]. URL: https://www.softwaretestinghelp.com/graph-implementation-cpp/

24. Graph Emplementation in C++ [Электронный ресурс]. URL: https://stackoverflow.com/questions/5493474/graph-implementation-c

25. Draw.io: обзор, отзывы, аналоги, интеграция, сайт | BizzApps [Электронный ресурс]. URL: https://bizzapps.ru/p/draw-io/

26. Visual C++ на примерах. Анатолий Хомоненко, Галина Довбуш [Электронный ресурс]. URL: https://all-journals.com/books/programming/10174-visual-c-na-primerah-2007-galina-dovbush-anatloii.html

Приложение А

```
string fillMass(int n, string* paths, int price, int** info)
{
   //-----
   string mainWay = "";
   int** graph = new int* [n];
   for (int i = 0; i < n; i++)
      graph[i] = new int[n];

   for (int i = 0; i < n; i++)
      for (int j = 0; j < n; j++)
         graph[i][j] = 0;

   for (int i = n - 1; i >= 0; i--)
   {
      int point = 0, mark = 1;

      if (paths[i].length() < 2)
      {
         point = atoi(paths[i].c_str());

         if (point > 0)
            graph[point - 1][i] = mark++;
      }
      else
      {
         for (int j = 0; j < paths[i].length(); j++)
            if (paths[i][j] != ',')
            {
               if (j < paths[i].length() - 1 && paths[i][j + 1] != ',')
               {
                  point = (paths[i][j] - '0') * 10 + paths[i][j + 1] - '0';
                  j++;
               }
               else
                  point = paths[i][j] - '0';

               if (point > 0)
```

```
                graph[point - 1][i] = mark++;
            }
        }
    }
```

Продолжение Приложение А

```
//---

int max = 0;
for (int i = 0; i < n; i++)
{
    for (int j = 0; j < n; j++)
    {
        if (graph[i][j] > max)
            max = graph[i][j];
    }
}

int flag = max;
string ways = readAllWays(graph, n, max, n);
int tmpL = ways.length();
ways += readAllWays(graph, 10, max, 10);
if (ways.length() > tmpL)
    max++;

if (max == 1)
{
    bool oneWay = true;
    int left = 0, right = 0;

    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            if (i != j && graph[i][j] == 1)
            {
                left++;
            }
            if (i != j && graph[j][i] == 1)
            {
                right++;
            }
        }
        if (right > 1 || left > 1)
```

```
            oneWay = false;
        right = 0;
        left = 0;
    }
```

Продолжение Приложение А

```
    if (oneWay)
    {
        cout << "Путь один, оптимизация не возможна" << endl;
        cin.get();
        return "";
    }
}
else if (max < 1)
{
    cout << "ошибка ввода данных" << endl;
    return "";
}


string* probableWays = new string[max];
int length = 0, tmpDays, days = 0;

for (int i = 0; i < max; i++)
{
    probableWays[i] = splitWays(ways, length);
    length += probableWays[i].length() + 1;
    if (probableWays[i][probableWays[i].length() - 2] != '1' ||
probableWays[i][probableWays[i].length() - 1] != '2')
        probableWays[i] += ",12";


    tmpDays = countOfDay(probableWays[i], info);

    if (tmpDays > days)
    {
        days = tmpDays;
        mainWay = probableWays[i];
    }
}

days = countOfDay(mainWay, info);
secondMethodDays = countOfDay(mainWay, info, false);
```

```
    cout << "Критический путь " << mainWay << endl;
    cout << endl << "количество дней " << days << endl;
```



```
    cout << "\nРасходы при обычном варианте: " << days * price << endl;

if (choise == 1)
{
    ofstream write;
    write.open("result.txt", ios::app);

    if (write.is_open())
    {
        write << "Критический путь " << mainWay << endl;
        write << endl << "количество дней " << days << endl;
        write << "Расходы при обычном варианте: " << days * price << endl;
    }
    write.close();
}

int fullPrice = days * price, optimization = 0, curentPrice = 0;

do
{
    optimization = fullPrice;

    int length = 0, fasterWay = days, tmp = 0, fasterI = 0;

    for (int i = 0; i < max; i++)
    {
        tmp = bestWay(probableWays[i], mainWay, days, info);

        if (tmp < fasterWay && tmp > 0)
        {
            fasterWay = tmp;
            fasterI = i;
        }
    }

    flag--;

    if (fasterWay > 0)
```

```
    {
```

Продолжение Приложение А

```
            cout << "\nНаименьший резерв " << fasterWay << " по пути " <<
probableWays[fasterI] << endl;
        if (choise == 1)
        {
            ofstream write;
            write.open("result.txt", ios::app);

            if (write.is_open())
                write << "Наименьший резерв " << fasterWay << " по пути " <<
probableWays[fasterI] << endl;

            write.close();
        }

        string reductionWays = str(probableWays[fasterI], mainWay);

        curentPrice += findBestWey(reductionWays, price, fasterWay, days, info);

        days -= fasterWay;

        fullPrice = price * days + curentPrice;

        mainWay = probableWays[fasterI];

            cout << "\nСтоимость проекта " << fullPrice <<" дней " <<  days <<
endl;
        if (choise == 0)
        {
            ofstream write;
            write.open("result.txt", ios::app);

            if (write.is_open())
                write << "\nСтоимость проекта " << fullPrice << " дней " <<  days
<<  endl;

            write.close();
        }
    }
```

```
    } while (fullPrice < optimization && flag >= 2);

    return mainWay;
}
```