

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 оптимизация сетевых графиков в условиях неопределенности»

Обучающийся

С.А. Букушкин
(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 оптимизация сетевых графиков в условиях неопределённости».

Объектом исследования являются методы сетевого планирования.

Предметом исследования является сетевое планирование в условиях неопределенности.

Целью выпускной квалификационной работы является реализация программы построения сетевого графика, а также вычисления вероятностных оценок.

В первой главе выпускной квалификационной работы приводится алгоритм нахождения критического времени выполнения проекта, а также излагается метод построения топологии критического пути. Кроме того, в первой главе приводится обоснование того, что основной параметр сетевой задачи – время выполнения отдельной работы подчиняется β -распределению. Предположение о β -распределении продолжительности работы (i, j) позволяет получить оценки рассматриваемого параметра: математическое ожидание и дисперсию.

Во второй главе выпускной квалификационной работы приводится решение конкретной сетевой задачи в предположении, что время выполнения отдельной работы подчиняется β -распределению, а случайные величины критическое время и критический путь распределены нормально.

В третьей главе выпускной квалификационной работы представлен анализ и обзор сервисов и инструментов необходимых для реализации программного приложения оптимизации сетевых проектов.

Результатом проделанной работы являются приложение, позволяющее определять критическое время и топологию критического пути, а также вычислять вероятностные характеристики сетевых проектов.

ABSTRACT

The title of the graduation work: "Analysis and optimization of network graphs in conditions of uncertainty."

The object of the graduation work is study is the methods of network planning.

The subject of the graduation work is study is network planning under conditions of uncertainty.

The aim of the work is to implement a program for building a network graph, as well as calculating probabilistic estimates.

The first chapter of the graduation work provides an algorithm for finding the critical time of the project, as well as a method for constructing the topology of the critical path. In addition, the first chapter provides a justification for the fact that the main parameter of a network task – the execution time of a separate work is subject to distribution. The assumption about the distribution of the duration of work allows us to obtain estimates of the parameter under consideration: mathematical expectation and variance.

In the second chapter of the graduation work, the solution of a specific network problem is given under the assumption that the execution time of an individual work is subject to a distribution, and the random variables critical time and critical path are distributed normally.

The third chapter of the graduation work presents an analysis and overview of the services and tools necessary for the implementation of the network project optimization software application.

Overall, the result of the graduation work done is an application that allows you to determine the critical time and topology of the critical path, as well as calculate the probabilistic characteristics of network projects.

Оглавление

Введение.....	5
Глава 1. Обзор и анализ методов сетевого планирования	7
1.1 Метод критического пути (СРМ)	7
1.2 Сетевые методы управления проектами.....	10
Глава 2. Задача. Вычисление вероятностных характеристик сетевых проектов	20
Глава 3. Разработка программы анализа и оптимизации сетевых графиков в условиях неопределенности.....	32
3.1 Обзор и анализ программного обеспечения для сетевого планирования	32
3.1.1 Онлайн-сервис Draw.io.....	32
3.1.2 Онлайн-калькулятор «Анализ сетевого графика».....	33
3.2 Реализация программы анализа и оптимизации сетевых графиков	34
3.3 Проверка программы на аналогах решения задачи анализа и оптимизации сетевых графиков.....	41
Заключение	45
Список используемой литературы и используемых источников.....	46
Приложение А Фрагмент программного кода	48

Введение

«В отличие от производственной деятельности, которую можно рассматривать чаще всего, как циклическую, любая проектная деятельность является однократной и неповторимой. Если даже проект повторяется, то прошлый опыт может лишь ограниченно подсказать руководителю проекта, чего можно ожидать при выполнении проекта, так как любая проектная деятельность сопряжена с неопределенностью. Информация, используемая в управлении проектами, обычно не бывает достоверной на все сто процентов» [15].

Таким образом, анализ и оптимизация сетевых графиков в условиях неопределённости играет важную роль.

Актуальность работы заключается в том, что большинство решений связанных с сетевыми проектами приходится принимать в условиях неточности исходной информации о самом объекте и внешней среде, в которой он функционирует и развивается, а значит в условиях неопределенности.

Объектом исследования являются методы сетевого планирования.

Предметом исследования является сетевое планирование в условиях неопределенности.

Целью выпускной квалификационной работы является реализация программы построения сетевого графика, а также вычисления вероятностных оценок.

Задачи выпускной квалификационной работы состоят в следующем:

1. Изучить математические методы решения сетевых задач;
2. Выполнить оптимизацию сети в условиях неопределённости и оценить её основные параметры;
3. Построить алгоритм решения сетевой задачи в условиях неопределённости и реализовать программный модуль.

Работа состоит из трёх глав:

В первой главе выпускной квалификационной работы приводится алгоритм нахождения критического времени выполнения проекта, а также излагается метод построения топологии критического пути. Кроме того, в первой главе приводится обоснование того, что основной параметр сетевой задачи – время выполнения отдельной работы подчиняется β -распределению. Предположение о β -распределении продолжительности работы (i, j) позволяет получить оценки рассматриваемого параметра: математическое ожидание и дисперсию.

Во второй главе выпускной квалификационной работы приводится решение конкретной сетевой задачи в предположении, что время выполнения отдельной работы подчиняется β -распределению, а случайные величины критическое время и критический путь распределены нормально.

В третьей главе выпускной квалификационной работы представлен анализ и обзор сервисов и инструментов необходимых для реализации программного приложения оптимизации сетевых проектов.

В приложении содержится только фрагмент кода.

Глава 1. Обзор и анализ методов сетевого планирования

Сетевое планирование – это метод начала и завершения незаконченных составляющих проекта, который позволяет провести оценку временных форматов системы, успешно реализующаяся в пределах этого проекта, а также получение или прогнозирование общего периода реализации задачи.

Основными элементами, которыми пользуются в моделях, являются событие и работа.

Событие – момент, отражающий определенный период выполнения задачи, завершения какого-либо этапа, а также момент выполнения новой работы. Событие не имеет определённый этап по времени, так как оно выполняется мгновенно.

Работа – процесс, в котором требуется время и ресурсы для успешного его выполнения в поставленные сроки и без лишних затрат. Также существует несколько видов работы:

– ожидание работы - процесс, который происходит в определенный период времени, без затрат и ресурсов (простым примером такого вида может служить просушка после покраски обоев или затверждение бетона);

– фиктивные работы – особенный вид, где происходит связь между промежутками работ. Этот вид не является затратным ни по времени, ни по ресурсам. Лишь после того, как закончится первый этап работы может начаться следующий этап, но никак не одновременно.

1.1 Метод критического пути (СРМ)

Одним из методов сетевого планирования является метод критического пути. Также его ещё называют анализом критического пути. Это некий алгоритм, который планирует действия в определенном проекте. Метод был разработан в конце 1950-х годов. Разработчиками стали Морган Уокер и Джеймс Келли. Первым событием, где использовался критический

путь стало строительство в Нью-Йорке Башен Близнецов. На рисунке 1 представлен критический путь.



Рисунок 1 – Критический путь

Этот метод был изобретен именно для реализации сложных проектов. Но в обычной жизни он применяется достаточно редко, так как даже при его использовании редко у кого получается выполнить задачу в поставленный промежуток времени. Именно критический путь распределяет основные работы проекта. Также в проекте могут присутствовать и вспомогательные работы, но они являются лишь подготовкой для реализации критических работ.

Рассмотрим преимущества и недостатки в данном методе, а также шаги для его реализации.

Во-первых, критический путь дает возможность не сбиться с поставленного пути в начале проекта. График определяет конкретные работы или задачи, которые должны быть выполнены в определенный период. Также, если задачи начинают откладываться, то критический путь подскажет какие задачи можно опустить, чтобы вернуться в поставленный промежуток времени. Чтобы критический путь работал всегда перед началом нужно четко определить общие план проекта:

- минимальное количество задач, которые могут быть отменены;
- количество времени, которое отводится на реализацию всего проекта;
- определенное время, когда каждая задача должна либо стартовать, либо завершаться;
- определить временный резерв, который можно использовать для анализа и оптимизации определенных задач;
- найти и разработать альтернативы для замены устаревших путей.

Критический путь – это путь от начальной к конечной вершине сетевого графика, проходящий только через критические работы.

Также есть и ряд требований:

- все задачи должны быть распределены;
- определить длительность для каждой задачи;
- начало новой задачи начинается с окончанием предыдущей.

На рисунке 2 представлен пример метода критического пути.

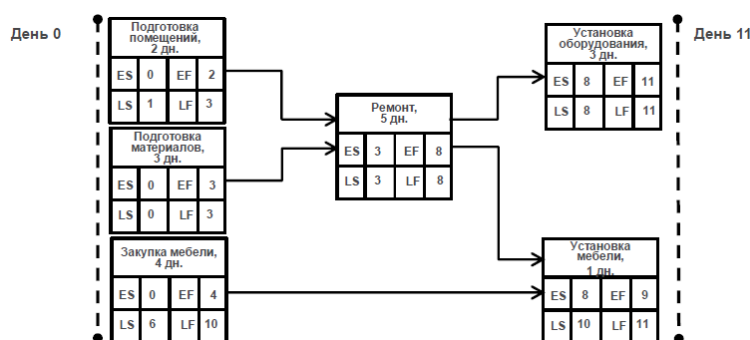


Рисунок 2 – Пример метода критического пути

Цели критического пути:

- уменьшение общего времени события в связи с уменьшением сроков задач;
- определение статуса каждой задачи в данном проекте [1][4][14].

1.2 Сетевые методы управления проектами

«Существует огромное количество типов сетевых моделей. Рассмотрев наличие вероятностных элементов сетевые модели разделяются на три структуры:

- детерминированная;
- стохастическая;
- смешанная.

Суть детерминированной модели заключается в том, что работы, а также их временная продолжительность и конечные требования строго установлены.

В этой модели лучше всего использовать графический или аналитический способы нахождения временных характеристик.

Стохастический метод используется, если работы определены с вероятностью. Для нахождения временных характеристик лучше использовать методы теории вероятности, а также математической статистики.

Смешанные модели включают в себя, как и стохастические, так и детерминированные структуры.

Рассмотрим теперь в целом сетевые модели в условиях неопределённости, а также и характеристики сетевых графиков» [15].

Критическое время - ранний срок наступления конечного события и обозначается $T_{кр}$. Проект не может быть завершён раньше момента времени. Значит, критическое время – это минимальный срок окончания проекта.

Менее, чем за критическое время проект выполнить нельзя, не сокращая время выполнения работ. Любая задержка работ приводит к увеличению общей продолжительности работ. Критическая работа не может быть передвинута из-за увеличения продолжительности работ. Смещение времени без увеличения всего проекта может быть только у момента начала работы. Таким образом, чтобы определить критическое время необходимо определить ранний срок выполнения завершающего проекта.

Сетевые модели определяются тем, что время выполнение всего проекта содержат случайные величины. Можно сказать, что продолжительность работы $(t(i,j))$ – это случайная величина, где имеются свои числовые характеристики. Это математическое ожидание $t(i,j)$ и дисперсия σ , которые находятся по формулам:

$$\text{Математическое ожидание } \bar{t} = \frac{a+4m+b}{6}, \quad (1)$$

Где, a – меньшее время выполнения работы;

b – наибольшее время выполнения работы;

m – вероятное время выполнения работы;

$$\text{Дисперсия } \sigma^2 = \left(\frac{b-a}{6}\right)^2. \quad (2)$$

Значения b и a были рассмотрены в формуле математического ожидания.

Зная эти значения, можно найти критический путь сетевого графика.

«Сетевой график имеет несколько требований, которые необходимо соблюдать:

– вся сеть всегда должна быть изображена и идти слева на право. Любое событие, которые выполнялось раньше всегда должны быть расположены левее;

– график не должен иметь тупиковые события, такое может произойти только если событие завершается;

– модель не должна содержать событий, у которых нет предшествующих работы. Исключение составляют лишь исходные значения;

– замкнутых циклов в сетевой модели не допускается» [3].

«Проанализируем все основные параметры сетевого графика.

– ранний срок $t_p(j)$ реализации события j – самый ранний срок завершения всех работ. На рисунке он располагается сверху вершины. Значения решаются путём сложения раннего срока события $t_p(i)$ и длительность самой работы $t(i,j)$:

$$t_p(j) = t(i,j) + t_p(i). \quad (3)$$

Чтобы рассчитать все ранние сроки необходимо при подсчете двигаться слева на право (от исходного до конечного).

– поздний срок $t_n(i)$ реализации события i – самый поздний срок завершения всех работ. На рисунке он располагается снизу вершины. Находится как разность позднего срока $t_n(j)$ и длительность работы $t(i,j)$:

$$t_n(i) = t_n(j) - t(i,j). \quad (4)$$

Поздний срок рассчитывается обратно предыдущему. Двигаться справа на лево (от конечного до исходного).

С помощью сетевого графика и находится критическое время $T_{кр}$. Любой проект не может быть завершён ранее критического времени. Критический путь – это путь, по которому будет проходить самый ранний срок выполнения работы. На сетевом графике обозначается двойной линией. Любая задержка по критическому пути увеличивает срок выполнения всего проекта» [10].

Еще одним важным элементом являются резервы времени. Они бывают двух типов:

- полный резерв времени;
- независимый резерв времени;

Полный резерв времени даёт нам возможность узнать на сколько можно увеличить выполнение конкретной работы в проекте, учитывая, что срок выполнения всего проекта не должен измениться.

Рассчитывается как разность позднего срока и раннего срока событий и продолжительность работы:

$$r_n(i, j) = t_n(j) - t_p(i) - t(i, j). \quad (5)$$

Используется этот резерв в том случае, если предыдущая работы была завершена раньше срока.

Независимый резерв времени используется если все предшествующие работы закончились позднее срока, а все следующие начинались в ранний срок [16][17].

Использование этого резерва никак не отображается на других работах и находится по формуле:

$$r_n(i, j) = t_p(j) - t_n(i) - t(i, j). \quad (6)$$

Найдя критическое время, критический путь, а также резервы времени, мы можем посчитать время выполнения проекта [2][15].

Необходимо определить с заданной надежностью γ максимальный срок T выполнения всех работ. Рассчитывается путём сложения нормированного отклонения случайной величины ($z_\gamma \sigma_{k_p}$) и минимального времени, необходимо для выполнения проекта (\bar{t}_{k_p}):

$$T = \bar{t}_{k_p} + z_\gamma \sigma_{k_p}. \quad (7)$$

Отклонение случайно величины рассчитывается с помощью формулы Лапласа.

Таким образом, учет вероятностного выполнения работ имеет огромное значения для выполнения проекта. Позволяет определить вероятность выполнения проекта в указанный срок [7][9][10].

1.3 Вероятностные характеристики сетевых планов

В отличие от производственной деятельности, которую можно рассматривать чаще всего, как циклическую, любая проектная деятельность является однократной и неповторимой. Если даже проект повторяется, то прошлый опыт может лишь ограниченно подсказать руководителю проекта, чего можно ожидать при выполнении проекта, так как любая проектная деятельность сопряжена с неопределенностью. Информация, используемая в управлении проектами, обычно не бывает достоверной на все сто процентов. Вероятность наступления отдельных событий, влияющих на конечный результат, может быть установлена с некоторой степенью точности. Из-за отсутствия необходимой информации вероятность отдельных событий установить невозможно. Грамотное заключение контракта и дальнейшее планирование выполнения проекта невозможно без учета неопределенности исходной информации.

«Продолжительность работы по сетевому графику чаще всего является случайной величиной, о которой известно, что она может принимать лишь одно из ряда возможных значений, характеризующейся своим законом распределения, своими числовыми характеристиками: математическим ожиданием - $\bar{t}(i, j)$; дисперсией - $\sigma^2(i, j)$.

Установлено, что распределение продолжительности работ непрерывно и обладает свойствами унимодальности (наличием единичного максимума у кривой распределения) и положительной асимметрией (максимум кривой распределения смещен влево относительно медианы). Известное в математической статистике β - распределение обладает перечисленными

свойствами, поэтому в дальнейшем вычисления проводятся в предположении, что продолжительность работ имеет β -распределение.

Для определения числовых характеристик (математического ожидания и дисперсии) случайной величины продолжительности работы (i, j) на основании опроса ответственных исполнителей проекта и экспертов определяют три временные характеристики:

$t_{нв}(i, j)$ - наиболее вероятная оценка продолжительности работ (оценка продолжительности работ при нормальных условиях);

$t_0(i, j)$ - оптимистическая оценка, продолжительности работ (оценка при наиболее благоприятных условиях);

$t_n(i, j)$ - пессимистическая оценка (оценка при наиболее неблагоприятных условиях)» [3].

Очевидно, что $t_0(i, j) < t_{нв}(i, j) < t_n(i, j)$.

При определении временных оценок работы (i, j) учитывается ряд факторов. Оценки не должны зависеть от возможных ситуаций при выполнении других работ; не должны влиять на оценки события, которые случаются крайне редко (стихийные бедствия: пожары, наводнения, ураганы и т.д.). Оценки должны учитывать возможности появления событий, классифицируемых как случайные величины, например, вызванных погодными условиями [11][12][13].

Плотность вероятности $f(t)$ β -распределения продолжительности работы (i, j) имеет четыре параметра распределения: t_n, t_0, α, γ

$$f(t) = \frac{(t-t_0)^\alpha (t_n-t)^\gamma}{\int_{t_0}^{t_n} (t-t_0)^\alpha (t_n-t)^\gamma dt} ; \quad (8)$$

Параметры α, γ заменяются математическим ожиданием - $\bar{t}(i, j)$; дисперсией - $\sigma^2(i, j)$.

На рисунке 3 представлено нормальное распределение.

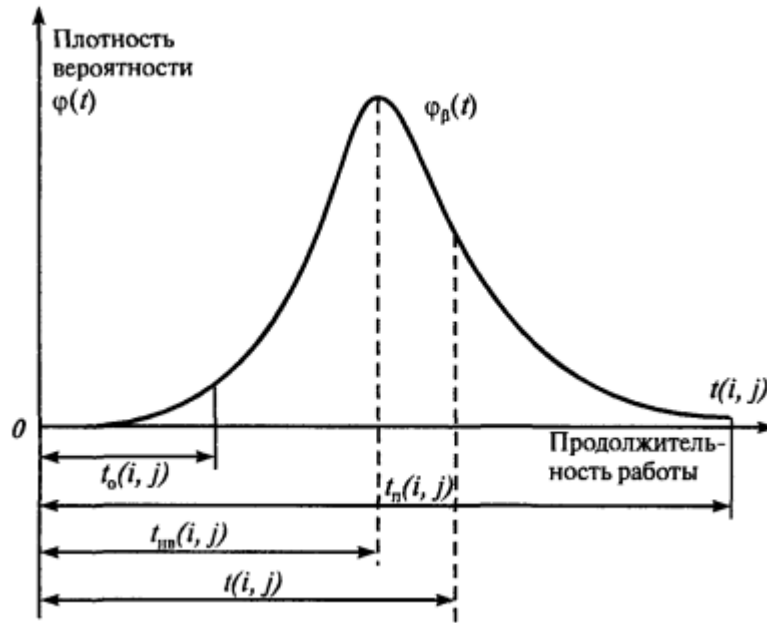


Рисунок 3 – Нормальное распределение

«Предположение о β -распределении продолжительности работы (i, j) позволяет получить следующие оценки математического ожидания и дисперсии

$$\bar{t}(i, j) = \frac{t_0(i, j) + 4t_{нн}(i, j) + t_n(i, j)}{6} ; \quad (9)$$

$$\sigma^2(ij) = \left[\frac{t_n(i, j) - t_0(i, j)}{6} \right]^2 ; \quad (10)$$

«Наиболее вероятное время выполнения работы $t_{нн}$ довольно сложно оценить даже специалистам. Поэтому при проектировании чаще используется упрощенная, менее точная оценка средней продолжительности работ

$$\bar{t}(i, j) = \frac{2t_0(i, j) + 3t_n(i, j)}{5}; \quad (11)$$

$\bar{t}(i, j)$ и $\sigma^2(i, j)$ позволяют определить временные параметры сетевого графика и оценить их надежность.

При достаточно большом количестве работ, принадлежащих пути T , можно применить центральную предельную теорему Ляпунова, на основании которой можно утверждать, что общая продолжительность пути T подчиняется нормальному закону распределения со средним значением $\bar{t}(T)$ равным сумме средних значений $\bar{t}(i, j)$ продолжительности составляющих его работ и дисперсией σ^2 равной сумме соответствующих дисперсий $\sigma^2(i, j)$ » [15].

$$\bar{t}(T) = \sum_{i,j} \bar{t}(i, j); \quad (12)$$

$$\sigma^2(T) = \sum_{i,j} \sigma^2(i, j). \quad (13)$$

Все временные параметры в условиях неопределенности являются средними значениями соответствующих случайных величин. Любая сеть с случайными продолжительностями работ не останавливается только нахождению временных параметров. Важным является и то, что оценка вероятности выполнения проекта не превзойдет директивный срок. Предположим, что срок выполнения проекта – это случайная величина, то получится:

$$P(t_{kp} \leq T) = \frac{1}{2} + \frac{1}{2} \Phi\left(\frac{T - \bar{t}_{kp}}{\sigma_{kp}}\right). \quad (14)$$

На рисунке 4 изображена продолжительность работы.

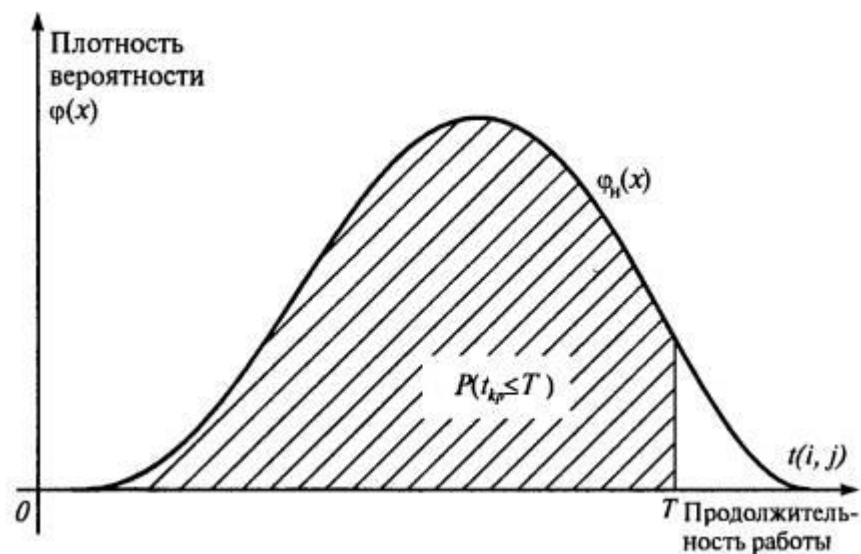


Рисунок 4 – Продолжительность работы

$\Phi(z)$ – значение интеграла вероятностей Лапласа;

σ_{kp} – среднее квадратическое отклонение длины критического пути:

$$\sigma_{kp} = \sqrt{\sigma_{kp}^2}, \quad (15)$$

а \bar{t}_{kp} и σ_{kp}^2 рассчитываются по формулам 12 и 13 соответственно.

Если оценка вероятности выполнения проекта мала, то опасность срыва проекта велика, поэтому надо принять меры по пересмотру задач, а также по перераспределению ресурсов. Если же оценка вероятности проекта высока, то можно полагать, что проект будет выполнен в указанный срок.

Также необходимо произвести расчёт и максимального срока выполнения проекта с заданной надежностью. Тогда:

$$T = \bar{t}_{кр} + z_{\beta} \sigma_{кр}^2, \quad (16)$$

где z_{β} – нормированное отклонение случайной величины, которая рассчитывается с помощью функции Лапласа $\Phi(z_{\beta}) = \beta$.

Выводы к главе 1

Первая глава ВКР посвящена обзору алгоритма нахождения критического времени выполнения проекта, а также методу построения топологии критического пути. Приводится обоснование того, что основной параметр сетевой задачи – время выполнения отдельной работы подчиняется β -распределению, что позволяет получить оценки рассматриваемого параметра: математическое ожидание и дисперсию. Кроме того, приводится обоснование того, что случайная величина критический путь подчиняется нормальному закону распределения. Предположение, что критический путь подчиняется нормальному распределению позволяет получить интервальные оценки неизвестных параметров.

Глава 2. Задача. Вычисление вероятностных характеристик сетевых проектов

«Для трехпараметрической модели найти ожидаемое время выполнения проекта, определить вероятность выполнения проекта не позднее заданного срока, найти интервал гарантированного (с вероятностью $P = 0,9973$) времени выполнения проекта, оценить максимально возможный срок выполнения проекта с заданной надежностью.

Таблица 1 – Данные сетевой задачи

Работа	Опирается на работы	$t_{\text{пес}}$	$t_{\text{вер}}$	$t_{\text{опт}}$
b ₁	-	9	4	3
b ₂	-	7	5	4
b ₃	-	13	6	2
b ₄	b ₁	8	6	3
b ₅	b ₁	6	5	2
b ₆	b ₃	10	8	3
b ₇	b ₂ , b ₅ , b ₆	9	4	3
b ₈	b ₂ , b ₅ , b ₆	13	7	5
b ₉	b ₄ , b ₇	9	6	2
b ₁₀	b ₃	11	5	3
b ₁₁	b ₂ , b ₅ , b ₆ , b ₁₀	9	5	2

Директивный (заданный) срок выполнения проекта $T_{\text{дир}} = 27$ дней. Заданная надежность $\gamma = 0,90$. Выполнить те же расчеты для двухпараметрической модели. Сравнить результаты» [6].

Решение сетевой задачи.

Найдем ожидаемую продолжительность работ ($t_{ож}$) для трехпараметрической модели по формуле:

$$t_{ож} = \frac{t_{пес} + 4t_{вер} + t_{опт}}{6}. \quad (17)$$

Например, $t_{ож}(b_1) = \frac{9+4*4+3}{6} = \frac{28}{6} \approx 4,66 \approx 5;$

$$t_{ож}(b_2) = \frac{7+4*5+4}{6} = \frac{31}{6} \approx 5,17 \approx 5;$$

$$t_{ож}(b_3) = \frac{13+4*7+5}{6} = \frac{39}{6} \approx 6,50 \approx 7;$$

$$t_{ож}(b_4) = \frac{8+4*6+3}{6} = \frac{35}{6} \approx 5,83 \approx 6;$$

$$t_{ож}(b_5) = \frac{6+4*5+2}{6} = \frac{28}{6} \approx 4,66 \approx 5;$$

$$t_{ож}(b_6) = \frac{10+4*8+3}{6} = \frac{45}{6} \approx 7,50 \approx 8;$$

$$t_{ож}(b_7) = \frac{9+4*4+3}{6} = \frac{28}{6} \approx 4,66 \approx 5;$$

$$t_{ож}(b_8) = \frac{6+4*5+2}{6} = \frac{46}{6} \approx 7,66 \approx 8;$$

$$t_{ож}(b_9) = \frac{9+4*6+2}{6} = \frac{35}{6} \approx 5,83 \approx 6;$$

$$t_{ож}(b_{10}) = \frac{11+4*5+3}{6} = \frac{34}{6} \approx 5,66 \approx 6;$$

$$t_{ож}(b_{11}) = \frac{9+4*5+2}{6} = \frac{31}{6} \approx 5,17 \approx 5.$$

Для упрощения дальнейших вычислений округляем полученные величины до целых чисел (по правилам округления с избытком и недостатком). Для сравнения найдем также ожидаемую продолжительность работ ($t^*_{ож}$) для двухпараметрической модели по формуле:

$$t^*_{ож} = \frac{3t_{пес} + 2t_{опт}}{5} \quad (18)$$

Например, $t^*_{ож}(b_1) = \frac{3*9+2*3}{5} = \frac{33}{5} \approx 6,60 \approx 7;$

$$\begin{aligned}
t^*_{\text{ож}}(b_2) &= \frac{3 \cdot 7 + 2 \cdot 4}{5} = \frac{27}{5} \approx 5,40 \approx 5; \\
t^*_{\text{ож}}(b_3) &= \frac{3 \cdot 13 + 2 \cdot 2}{5} = \frac{43}{5} \approx 8,60 \approx 9; \\
t^*_{\text{ож}}(b_4) &= \frac{3 \cdot 8 + 2 \cdot 3}{5} = \frac{30}{5} \approx 6 \approx 6; \\
t^*_{\text{ож}}(b_5) &= \frac{3 \cdot 6 + 2 \cdot 2}{5} = \frac{22}{5} \approx 4,40 \approx 4; \\
t^*_{\text{ож}}(b_6) &= \frac{3 \cdot 10 + 2 \cdot 3}{5} = \frac{36}{5} \approx 7,20 \approx 7; \\
t^*_{\text{ож}}(b_7) &= \frac{3 \cdot 9 + 2 \cdot 3}{5} = \frac{33}{5} \approx 6,60 \approx 7; \\
t^*_{\text{ож}}(b_8) &= \frac{3 \cdot 13 + 2 \cdot 5}{5} = \frac{49}{5} \approx 9,80 \approx 10; \\
t^*_{\text{ож}}(b_9) &= \frac{3 \cdot 9 + 2 \cdot 2}{5} = \frac{31}{5} \approx 6,20 \approx 6; \\
t^*_{\text{ож}}(b_{10}) &= \frac{3 \cdot 11 + 2 \cdot 3}{5} = \frac{39}{5} \approx 7,80 \approx 8; \\
t^*_{\text{ож}}(b_{11}) &= \frac{3 \cdot 9 + 2 \cdot 2}{5} = \frac{31}{5} \approx 6,20 \approx 6.
\end{aligned}$$

Двухпараметрическая модель проще, но дает менее точные оценки. Для вычисления дисперсий продолжительностей работ воспользуемся формулой:

$$\sigma^2(t_{\text{ож}}) = \left(\frac{t_{\text{пес}} - t_{\text{опт}}}{6} \right)^2 \quad (19)$$

$$\begin{aligned}
\sigma^2(t_{\text{ож}}(b_1)) &= \left(\frac{9 - 3}{6} \right)^2 = \left(\frac{6}{6} \right)^2 = 1; \\
\sigma^2(t_{\text{ож}}(b_2)) &= \left(\frac{7 - 4}{6} \right)^2 = \left(\frac{3}{6} \right)^2 \approx 0,25; \\
\sigma^2(t_{\text{ож}}(b_3)) &= \left(\frac{13 - 2}{6} \right)^2 = \left(\frac{11}{6} \right)^2 \approx 3,36; \\
\sigma^2(t_{\text{ож}}(b_4)) &= \left(\frac{8 - 3}{6} \right)^2 = \left(\frac{5}{6} \right)^2 \approx 0,69; \\
\sigma^2(t_{\text{ож}}(b_5)) &= \left(\frac{6 - 2}{6} \right)^2 = \left(\frac{4}{6} \right)^2 \approx 0,44; \\
\sigma^2(t_{\text{ож}}(b_6)) &= \left(\frac{10 - 3}{6} \right)^2 = \left(\frac{7}{6} \right)^2 \approx 1,36;
\end{aligned}$$

$$\sigma^2(t_{\text{ож}}(b_7)) = \left(\frac{9-3}{6}\right)^2 = \left(\frac{6}{6}\right)^2 \approx 1;$$

$$\sigma^2(t_{\text{ож}}(b_8)) = \left(\frac{13-5}{6}\right)^2 = \left(\frac{8}{6}\right)^2 \approx 1,77;$$

$$\sigma^2(t_{\text{ож}}(b_9)) = \left(\frac{9-2}{6}\right)^2 = \left(\frac{7}{6}\right)^2 \approx 1,36;$$

$$\sigma^2(t_{\text{ож}}(b_{10})) = \left(\frac{11-3}{6}\right)^2 = \left(\frac{8}{6}\right)^2 \approx 1,77;$$

$$\sigma^2(t_{\text{ож}}(b_{11})) = \left(\frac{9-2}{6}\right)^2 = \left(\frac{7}{6}\right)^2 \approx 1,36.$$

Таблица 2 – Полученные дополнительные данные

Работа	Опирается на работы	$t_{\text{пес}}$	$t_{\text{вер}}$	$t_{\text{опт}}$	$t_{\text{ож}}$	$t^*_{\text{ож}}$	σ^2
b ₁	-	9	4	3	5	7	1
b ₂	-	7	5	4	5	5	0,25
b ₃	-	13	6	2	7	9	3,36
b ₄	b ₁	8	6	3	6	6	0,69
b ₅	b ₁	6	5	2	5	4	0,44
b ₆	b ₃	10	8	3	8	7	1,36
b ₇	b ₂ , b ₅ , b ₆	9	4	3	5	7	1
b ₈	b ₂ , b ₅ , b ₆	13	7	5	8	10	1,77
b ₉	b ₄ , b ₇	9	6	2	6	6	1,36
b ₁₀	b ₃	11	5	3	6	8	1,77
b ₁₁	b ₂ , b ₅ , b ₆ , b ₁₀	9	5	2	5	6	1,36

«Таким образом трехпараметрическая модель сведена к однопараметрической. Теперь можно построить сетевой график и рассчитать его временные характеристики» [6]. На рисунке 5 представлен сетевой график.

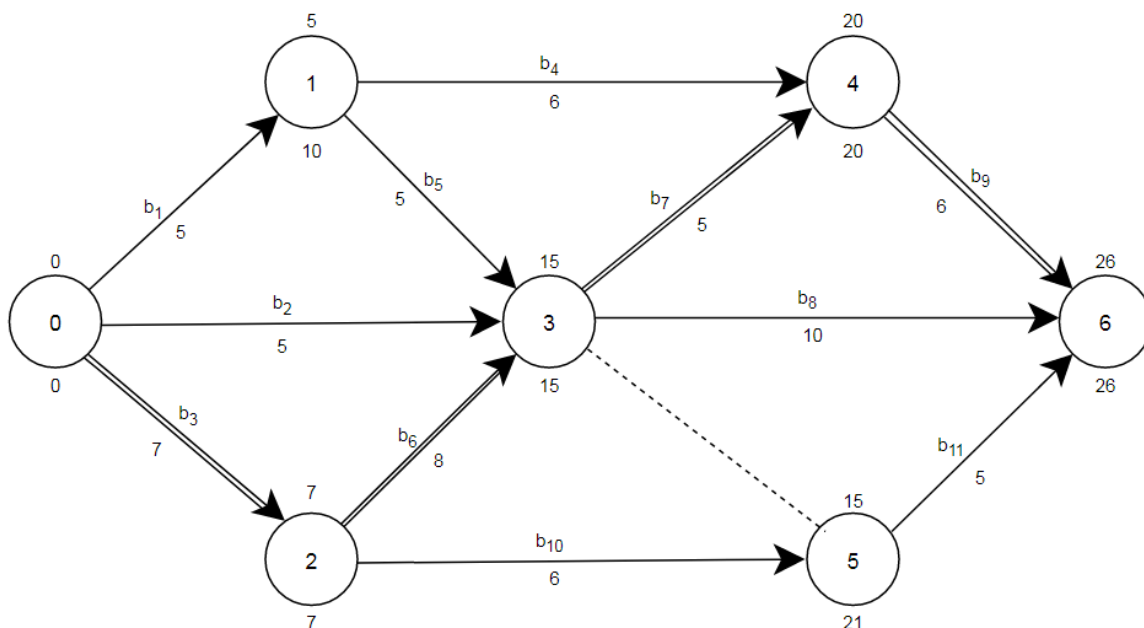


Рисунок 5 – Сетевой график

Таблица 3 – время ожидания

Работа	Опирается на работы	$t_{ож}$	σ^2
b_1	–	5	1,00
b_2	–	5	0,25
* b_3	–	7	3,36
b_4	b_1	6	0,69
b_5	b_1	5	0,44
* b_6	b_3	8	1,36
* b_7	b_2, b_5, b_6	5	1,00
b_8	b_2, b_5, b_6	8	1,77
* b_9	b_4, b_7	6	1,36
b_{10}	b_3	6	1,77
b_{11}	b_2, b_5, b_6, b_{10}	5	1,36

Ожидаемое критическое время $T_{кр} = 26$.

На критическом пути лежат работы b_3, b_6, b_7, b_9 .

«Найдем дисперсию критического пути.

$$\sigma_{kp}^2 = \sigma^2(b_3) + \sigma^2(b_6) + \sigma^2(b_7) + \sigma^2(b_9) = 3,36 + 1,36 + 1,00 + 1,36 = 7,08.$$

Среднеквадратическое отклонение критического пути

$$\sigma_{kp} = \sqrt{7,08} \approx 2,66$$

Найдем вероятность того, что проект будет выполнен не позднее заданного срока ($T_{дир} = 27$ день):

$$P(t_{kp} \leq 27) = 0,5 + \Phi\left(\frac{27-26}{2,66}\right) = 0,5 + \Phi(0,38) \approx 0,5 + 0,1480 \approx 0,6.$$

Таким образом, имеются неплохие шансы (60 %) выполнить проект в заданный срок.

Найдем интервал гарантированного времени выполнения проекта. Воспользуемся правилом «трех сигм»: $3\sigma_{kp} = 3 * 2,66 = 7,98 \approx 8$, то есть с вероятностью почти в 0,9973 проект будет выполнен за 26 ± 8 дней.

$$P(18 \leq t_{kp} \leq 34) = P(|t_{kp} - 26| \leq 8) \approx P(|t_{kp} - 26| \leq 3\sigma_{kp}) = 0,9973$$

Более точное значение:

$$P(18 \leq t_{kp} \leq 34) = P(|t_{kp} - 26| \leq 8) = 2\Phi\left(\frac{8}{2,66}\right) = 2\Phi(3,00) = 2 \times 0,49865 = 0,9973$$

$$P(t_{kp} \leq 34) = 0,5 + \Phi\left(\frac{34-26}{2,66}\right) = 0,5 + \Phi(3,00) = 0,5 + 0,49865 = 0,9986$$

Следовательно, можно с большой долей уверенности гарантировать, что максимальный срок выполнения проекта не превысит 34 дня. Оценим максимально возможный срок T выполнения проекта с заданной надежностью $\gamma = 0,90$.

По таблице значений функции Лапласа найдем доверительный коэффициент z_γ для заданной надежности γ .

$$\text{Так как } P(|t_{kp} - T_{кр}| \leq z_{0,90}\sigma_{kp}) = 2\Phi\left(\frac{z_{0,90}\sigma_{kp}}{\sigma_{kp}}\right) = 2\Phi(z_{0,90}) = 0,90,$$

$$\text{То из } 2\Phi(z_{0,90}) = 0,90 \Rightarrow \Phi(z_{0,90}) = 0,450 \quad z_{0,90} = 1,65 \quad \text{и}$$

$$P(|t_{kp} - 26| \leq 1,65 \times 2,66) = P(|t_{kp} - 26| \leq 4,39) = (21,61 \leq t_{kp} \leq 30,39) = 0,90.$$

Это значит, что с надежностью 0,90 проект будет завершен в период от 22 до 30 дней» [2].

$$P(t_{кр} \leq T) = 0,5 + \Phi\left(\frac{T - T_{кр}}{\sigma_{кр}}\right) = 0,5 + \Phi(z\gamma) = \gamma, \text{ где } T - T_{кр} = z\gamma\sigma_{кр}.$$

$$\text{Случай в проекте } 0,5 + \Phi(z_{0,90}) = 0,90 \quad \Phi(z_{0,90}) = 0,40,$$

$$z_{0,90} = 1,28 \text{ и}$$

$T = T_{кр} + z\gamma\sigma_{кр} = 26 + 1,28 \times 2,66 \approx 29,40 \approx 30$, то есть с надежностью 0,90 проект будет завершен не позже 30 дней.

$$\text{Здесь использована функция Лапласа вида } \Phi(x) = \frac{1}{\sqrt{2\pi}} \int_0^x e^{-\frac{t^2}{2}} dt. \quad (20)$$

Следует также помнить, что функция Лапласа данного вида нечетная, т. е. $\Phi(-x) = -\Phi(x)$, а при $x \geq 5$ значение $\Phi(x) \approx 0,5$.

«Рассмотрим теперь двухпараметрическую модель. Такая модель используется в тех случаях, когда сложно определить наиболее вероятные продолжительности выполнения отдельных работ. Ожидаемая продолжительность работ ($t^*_{ож}$) для двухпараметрической модели нами уже вычислена. Среднеквадратическое отклонение этих продолжительностей то же, что и у трехпараметрической модели. Итак, для двухпараметрической модели имеются следующие данные, представленные в таблице» [6]:

Таблица 4 – Данные двухпараметрической модели

Работа	Опирается на работы	$t_{пес}$	$t_{опт}$	$t^*_{ож}$	σ^2
b_1	–	9	3	7	1,00
b_2	–	7	4	5	0,25
b_3	–	13	2	9	3,36
b_4	b_1	8	3	6	0,69
b_5	b_1	6	2	4	0,44
b_6	b_3	10	3	7	1,36
b_7	b_2, b_5, b_6	9	3	7	1,00
b_8	b_2, b_5, b_6	13	5	10	1,77
b_9	b_4, b_7	9	2	6	1,36

b_{10}	b_3	11	3	8	1,77
b_{11}	b_2, b_5, b_6, b_{10}	9	2	6	1,36

Проведем анализ полученной модели. Найдем временные характеристики событий и работ. Построим структурный сетевой график. На рисунке 6 представлен сетевой график.

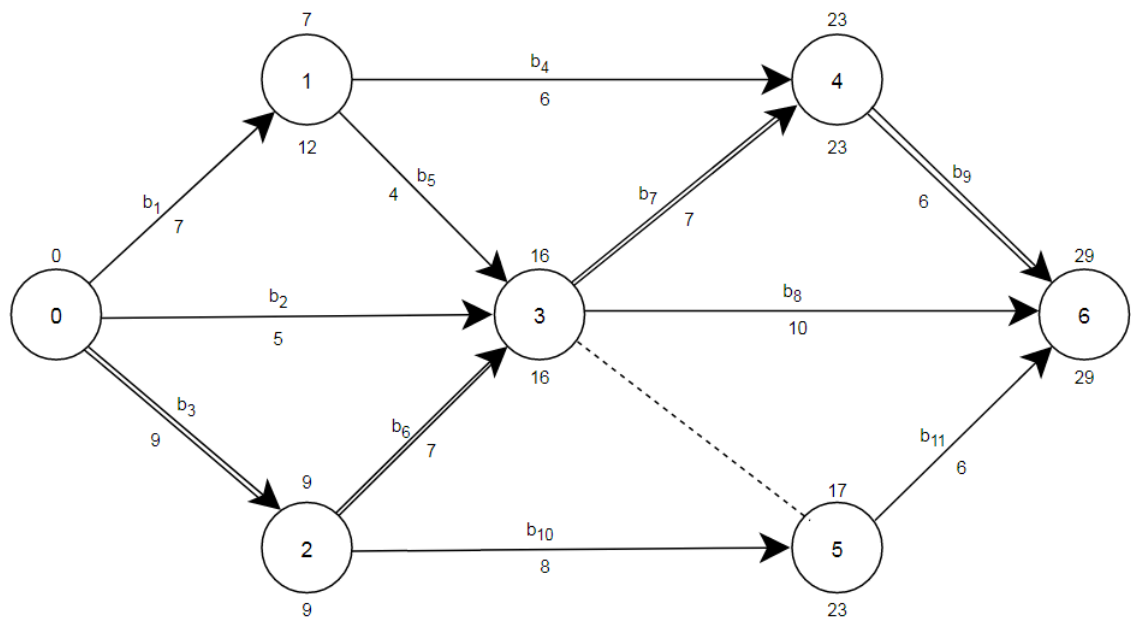


Рисунок 6 – Сетевой график

Таблица 5 – Сроки наступления и резервы времени событий

Событие	Ранний срок $T_p(i)$	Поздний срок $T_n(i)$	Резерв времени $R(i)$
*0	0	0	0
1	7	12	5
*2	9	9	0
*3	16	16	0
*4	23	23	0
5	17	23	6
*6	29	29	0

Критический путь проходит через события с нулевым резервом времени, т. е. через события 0, 2, 3, 4, 6. Критическое время $T_{кр} = 29$ дней.

Таблица 6 – Характеристика работы

Работа	t^*_0 ж	σ^2	$S_p(b_k)$	$S_{\pi}(b_k)$	$E_p(b_k)$	$E_{\pi}(b_k)$	$r_{\pi}(b_k)$	$r_{\pi}(b_k)$
$b_1 = (0,1)$	7	1,00	0	5	7	12	5	0
$b_2 = (0,3)$	5	0,25	0	11	5	16	11	11
$b_3 = (0,2)$	9	3,36	0	0	9	9	0	0
$b_4 = (1,4)$	6	0,69	7	17	13	23	10	5
$b_5 = (1,3)$	4	0,44	7	12	11	16	5	0
$b_6 = (2,3)$	7	1,36	9	9	16	16	0	0
$b_7 = (3,4)$	7	1,00	16	16	23	23	0	0
$b_8 = (3,6)$	10	1,77	16	19	26	29	3	3
$b_9 = (4,6)$	6	1,36	23	23	29	29	0	0
$b_{10} = (2,5)$	8	1,77	9	15	17	23	6	0
$b_{11} = (5,6)$	6	1,36	17	23	23	29	6	0
$\phi = (3,5)$	0	0	16	23	16	23	7	1

$\Phi = (3,5)$ – фиктивная работа.

Критические работы – b_3, b_6, b_7, b_9 . Резервы времени этих работ равны нулю.

«Найдены наиболее ранние и наиболее поздние сроки начала и окончания каждой работы.

Наиболее ранний возможный срок начала работы $b_k = (i, j)$:

$$S_p(b_k) = S_p(i, j) = T_p(i) \quad (21)$$

Наиболее поздний допустимый срок начала работы $b_k = (i, j)$:

$$S_{\pi}(b_k) = S_{\pi}(i, j) = T_{\pi}(j) - t_{ij} \quad (22)$$

Наиболее ранний возможный срок окончания работы $b_k = (i, j)$:

$$E_p(b_k) = E_p(i, j) = T_p(i) + t_{ij} \quad (23)$$

Наиболее поздний допустимый срок окончания работы $b_k = (i, j)$:

$$E_n(b_k) = E_n(i, j) = T_n(j) \quad (24)$$

Полный резерв работы $b_k = (i, j)$ можно найти и проверить по формуле:

$$r_n(b_k) = r_n(i, j) = T_n(j) - T_p(i) - t_{ij} = S_n(b_k) - S_p(b_k) = E_n(b_k) - E_p(b_k) \quad (25)$$

Независимый резерв работы $b_k = (i, j)$ может быть найден по формулам:

$$r_n(b_k) = r_n(i, j) = T_p(j) - T_p(i) - t_{ij} = r_n(i, j) - R(i) - R(j) \quad (26)$$

Резервы времени и коэффициенты напряженности некритических дуг запишем в таблицу 7» [6]

Таблица 7 – Резервы времени и коэффициенты напряженности некритических дуг

Некритические дуги	Резерв времени дуги, $R(b) = a-b$	Коэффициент напряженности дуги, $N(b)$
(0,1,4,6)	10	$19/29 \approx 0,65$
(0,1,3,6)	8	$21/29 \approx 0,72$
(0,1,3,5,6)	12	$17/29 \approx 0,59$
(0,3,4,6)	11	$18/29 \approx 0,62$
(0,3,5,6)	18	$11/29 \approx 0,40$
(0,2,3,5,6)	7	$22/29 \approx 0,76$
(0,2,5,6)	6	$23/29 \approx 0,79$
(0,1,4)	5	$11/16 \approx 0,69$

Подведём итог нашей работы:

Критическое время $T_{кр} = 29$

На критическом пути лежат работы b_3, b_6, b_7, b_9

«Дисперсия критического пути:

$$\sigma_{кр}^2 = \sigma^2(b_3) + \sigma^2(b_6) + \sigma^2(b_7) + \sigma^2(b_9) = 3,36 + 1,36 + 1,00 + 1,36 \\ = 7,08.$$

Среднеквадратическое отклонение критического пути

$$\sigma_{кр} = \sqrt{7,08} \approx 2,66$$

Найдем вероятность того, что проект будет выполнен не позднее заданного срока ($T_{дир} = 27$ день):

$$P(t_{кр} \leq 27) = 0,5 + \Phi\left(\frac{27-29}{2,66}\right) = 0,5 - \Phi(0,75) \approx 0,5 - 0,2734 \approx 0,23.$$

Это означает, что вероятность выполнения проекта в срок является 23 %

Найдем интервал гарантированного времени выполнения проекта. Воспользуемся правилом «трех сигм»: $3\sigma_{кр} = 3 * 2,66 = 7,98 \approx 8$, то есть с вероятностью почти в 0,9973 проект будет выполнен за 29 ± 8 дней.

$$P(21 \leq t_{кр} \leq 37) = P(|t_{кр} - 29| \leq 8) \approx P(|t_{кр} - 29| \leq 3\sigma_{кр}) = 0,9973$$

Следовательно, можно с большой долей уверенности гарантировать, что максимальный срок выполнения проекта не превысит 37 дней. Оценим максимально возможный срок T выполнения проекта с заданной надежностью $\gamma = 0,90$.

По таблице значений функции Лапласа найдем доверительный коэффициент z_γ для заданной надежности γ .

$$\text{Так как } P(|t_{кр} - T_{кр}| \leq z_{0,90}\sigma_{кр}) = 2\Phi\left(\frac{z_{0,90}\sigma_{кр}}{\sigma_{кр}}\right) = 2\Phi(z_{0,90}) = 0,90,$$

$$\text{То из } 2\Phi(z_{0,90}) = 0,90 \Rightarrow \Phi(z_{0,90}) = 0,450 \quad z_{0,90} = 1,65 \quad \text{и}$$

$$P(|t_{кр} - 29| \leq 1,65 \times 2,66) = P(|t_{кр} - 29| \leq 4,39) = (24,61 \leq t_{кр} \leq 33,39) = 0,90.$$

Это значит, что с надежностью 0,90 проект будет завершен в период от 25 до 33 дней» [6].

$$P(t_{кр} \leq T) = 0,5 + \Phi\left(\frac{T - T_{кр}}{\sigma_{кр}}\right) = 0,5 + \Phi(z\gamma) = \gamma, \text{ где } T - T_{кр} = z\gamma\sigma_{кр}.$$

$$\text{Случай в проекте } 0,5 + \Phi(z_{0,90}) = 0,90 \quad \Phi(z_{0,90}) = 0,40,$$

$$z_{0,90} = 1,28 \text{ и}$$

$T = T_{кр} + z\gamma\sigma_{кр} = 29 + 1,28 \times 2,66 \approx 32,40 \approx 32$, то есть с надёжностью 0,90 проект будет завершён не позже 32 дней.

Найдём самую напряжённую не критическую дугу. Это дуга (0,2,5,6). Она составляет 0,79. Найдём её среднеквадратическое отклонение

$$\sigma^2(0,2,5,6) = \sigma^2(0,2) + \sigma^2(2,5) + \sigma^2(5,6) = \sigma^2(b_3) + \sigma^2(b_{10}) + \sigma^2(b_{11}) \\ = 3,36 + 1,77 + 1,36 = 6,49$$

$$\sigma(0,2,5,6) = \sqrt{6,49} \approx 2,55$$

«Среднеквадратическое отклонение длины данной дуги (2,55) меньше среднеквадратического отклонения критического пути (2,66). Значит, ожидаемое значение этой дуги $23 \pm 2,55$;» [6].

Выводы к главе 2

Вторая глава ВКР посвящена решению конкретной сетевой задачи в предположении, что время выполнения отдельной задачи подчиняется β -распределению, а случайные величины критическое время и критический путь распределены нормально. Вычислено ожидаемое время выполнения проекта, вычислена вероятность выполнения проекта не позднее заданного срока, определён интервал гарантированного времени выполнения проекта, определён максимально возможный срок выполнения проекта с заданной надёжностью.

Глава 3. Разработка программы анализа и оптимизации сетевых графиков в условиях неопределенности

3.1 Обзор и анализ программного обеспечения для сетевого планирования

Подробно рассмотрев задачу и её решение, мы можем приступить к программной реализации продукта, который позволит решать подобные задачи.

Рассмотрим все программы и технологии, которые уже решают такие важные задачи, как анализ и оптимизация сетевых графиков, а также позволяют вычислять вероятностные оценки неизвестных параметров.

3.1.1 Онлайн-сервис Draw.io

Один из самых удобных сервисов по построению любых сетевых графиков. Draw.io – бесплатный онлайн-сервис для создания всех продуктов и инструментов, которые используются в IT-технологиях.

С помощью этого сервиса и были созданы сетевые графики, которые были нужны нам для решения задачи, которая была представлена во второй главе. На рисунке 7 изображен пример сетевого графика.

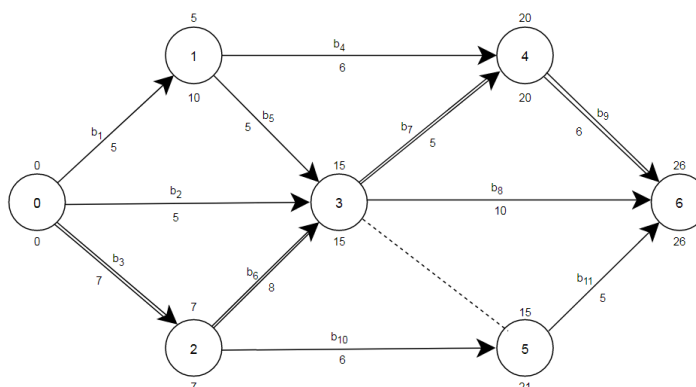


Рисунок 7 – Сетевой график

С помощью этого сервиса создаются прототипы сайтов, приложений и много другого. Главное его преимущество – это конечно же его универсальность. Он даёт возможность создавать организациям не только обычные диаграммы, но также и блок-схемы, UML, электросхемы и многое другое. Также важным преимуществом данного инструмента является то, что он полностью бесплатный. Также работает на любых устройствах и браузерах, а также имеет русский интерфейс. Конечно, у онлайн-сервиса draw.io есть аналоги. Например, одним из них является продукт от компании Microsoft – Microsoft Visio.

Сразу главным отличием от онлайн-сервиса можно сказать то, что продукт является платным. Также можно рассказать про пользовательский интерфейс. Он не самый лучший и чувствует себя немного неуклюже для разработки быстрых диаграмм и графиков.

3.1.2 Онлайн-калькулятор «Анализ сетевого графика»

Простой и удобный калькулятор для решения любых задач по математике, информатике, теории вероятности и многим другим предметам. На рисунке 8 представлен экран для ввода данных.

Анализ сетевого графика

С помощью данной программы можно онлайн определить параметры сетевого графика (рассчитать сроки свершения событий, резервы времени и критический путь), найти коэффициенты напряженности. Оптимизация сетевого графика проводится по следующим критериям: число исполнителей, резервы-затраты, сокращение сроков (см. вкладку [Параметры сетевого графика](#)).

Сетевой график можно нарисовать, а также задать в виде матрицы или таблицы (меню [Действия](#)).

[Ввод данных](#) [Параметры сетевого графика](#) [Решение](#) [Видеоинструкция](#) [Оформление Word](#)

Для сформированного графа можно выполнить следующие действия:

- Расчет коэффициентов напряженности
- Строить диаграмму Ганта Привязать к дате: 28.04.2018
- Решение секторальным методом
- Решение методом потенциалов
- Оптимизировать сетевой график по критерию: число исполнителей
- Формировать техническую документацию
- Оценить вероятность выполнения всего комплекса работ за дней
- Оценить максимально возможный срок выполнения всего комплекса работ с вероятностью %

Продолжительность, ед.изм.:

Рисунок 8 – Экран онлайн-калькулятора

Все вычисления могут производиться как на сайте, так и оформляться в Word. Также немаловажным преимуществом данного сервиса является то, что он не просто решает задачи, но и даёт всю информацию для пользователя как была решена та или иная задача, все формулы и подробное описание. Также преимуществом является его многоязычность. Сервис даёт возможностью использовать разные языки.

Главным недостатком является неудобное заполнения сетевого графика, а также небольшая избыточность информации, что создаёт трудности для нахождения этой информации в других источниках.

Проделав большой анализ и обзор всех сервисов и инструментов, можно сделать вывод, что лучшими характеристиками обладает онлайн-калькулятор. На его основе мы и приступим реализовывать своё собственное приложение анализа и оптимизация сетевых графиков в условиях неопределённости.

3.2 Реализация программы анализа и оптимизации сетевых графиков

В качестве средства средств разработки будем использовать Microsoft Visual Studio.

Microsoft Visual Studio – это программная среда по разработке приложений как с графическим интерфейсом, так и с консольным на базе операционной системы Windows [18].

На рисунке 9 изображена среда разработки Microsoft Visual Studio.

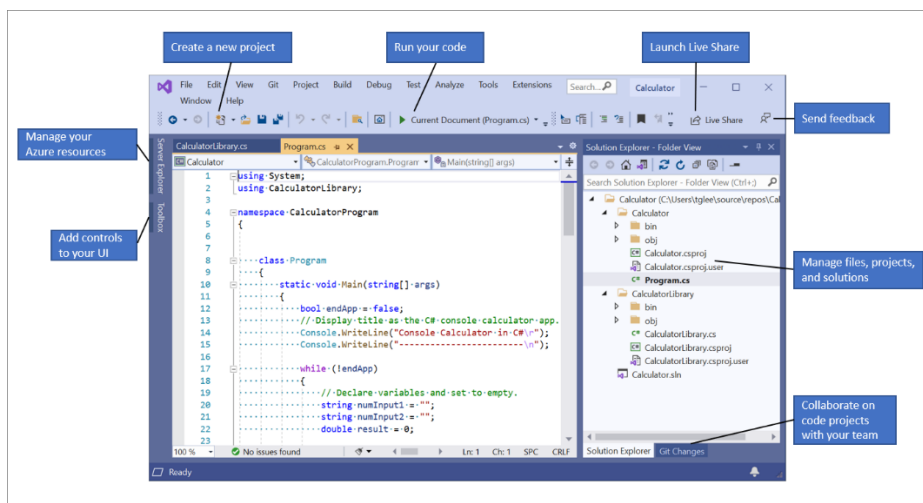


Рисунок 9 – Среда разработки Microsoft Visual Studio

В комплекте идут следующие компоненты:

– Visual Basic.Net

Представляет собой язык программирования, реализованный компанией Microsoft. Работает на платформе Microsoft и похож на синтаксис Бейсика. Преимуществом является поддержка различных языков, кроссплатформенность (переносимая платформа, поддерживается на всех операционных системах – Microsoft, MacOS, Linux), большая коллекция библиотека классов, разнообразие технологий.

– Visual C++

Самый универсальный инструмент, которая соединяет мощную библиотеку классов с визуальными инструментами. Без основ программирования будет невозможно написать программный продукт. Кроме общих инструментов, Visual Studio имеет и средства, которые необходимы для разработки машинного кода. Это MSBuild – система проектов, которая используется только в Microsoft Visual Studio. Поддерживается на всех операционных системах такие, как Microsoft, Linux, Android и iOS. Использует многие окна и средства аналогично другим языкам.

– Visual C#

Идёт активное развитие. Имеет много полезных функций:

- а) инкапсуляция;
- б) типизация;
- в) перегрузка операторов;
- г) наследование.

Поддержка нескольких языков. Основная библиотека классов. Единая для всех языков библиотека классов [19][20].

– Visual F#

Для языка F# имеются различные шаблоны элементов и проектов. Это .NET Core и .NET Standard. Так конструктор имеется из нескольких страниц свойств проекта. Обычно все страницы, которые используется для F#, являются подмножеством страниц.

Теперь рассмотрим структуру среды Microsoft Visual Studio. В неё входят:

- отладчик кода;
- редактор форм;
- веб-редактор;
- дизайнер классов;
- редактор исходного кода (рефакторинг кода и т.д.).

Основные преимущества Microsoft Visual Studio:

- универсальностью разработки;
- поддержка большого количества языков программирования;
- контроль за выполнение кода;
- меньше кода для написания;
- интуитивный стиль кодирования;

Основные недостатки Microsoft Visual Studio:

- много места для хранения данных;
- требовательность к ресурсам;
- цена продукта;
- сложность при самостоятельном освоении материала [21][22][23].

Процесс создания приложения в Microsoft Visual Studio:

– создание нового проекта;

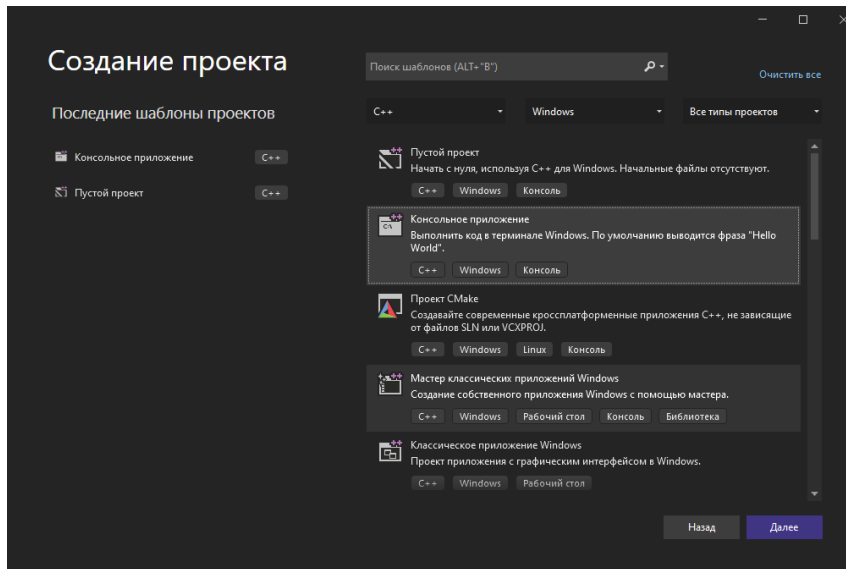


Рисунок 10 – Создание проекта

– добавление библиотек (в настоящее время имеется более 1700 библиотек в каталоге);

– написание кода;

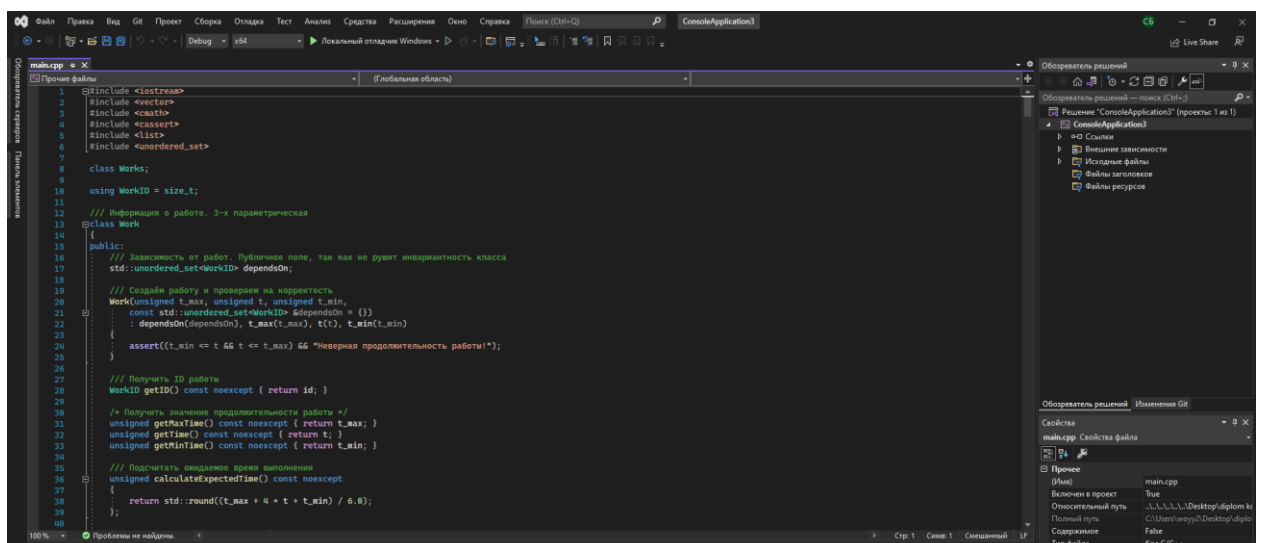
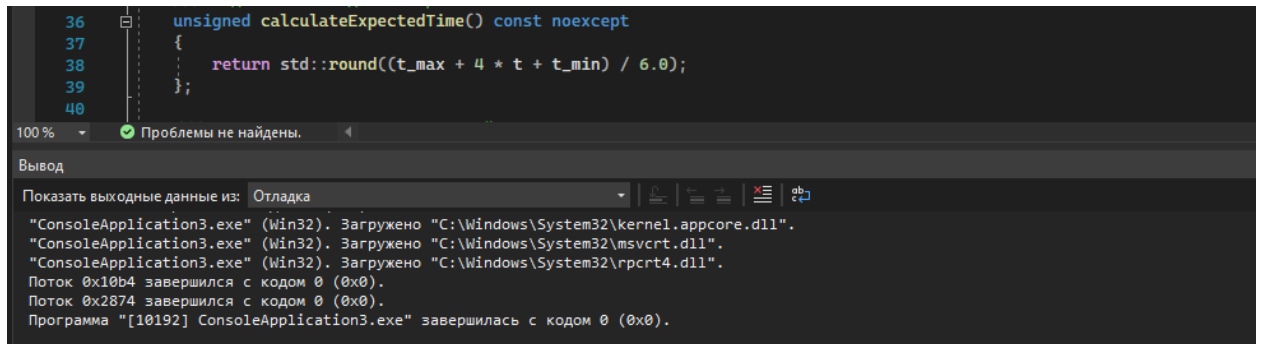


Рисунок 11 – Код программы

– сборка (компиляция и компоновка);



```
36     unsigned calculateExpectedTime() const noexcept
37     {
38         return std::round((t_max + 4 * t + t_min) / 6.0);
39     };
40
```

100 % Проблемы не найдены.

Вывод

Показать выходные данные из: Отладка

"ConsoleApplication3.exe" (Win32). Загружено "C:\Windows\System32\kernel.appcore.dll".
"ConsoleApplication3.exe" (Win32). Загружено "C:\Windows\System32\msvcrt.dll".
"ConsoleApplication3.exe" (Win32). Загружено "C:\Windows\System32\rpcrt4.dll".
Поток 0x10b4 завершился с кодом 0 (0x0).
Поток 0x2874 завершился с кодом 0 (0x0).
Программа "[10192] ConsoleApplication3.exe" завершилась с кодом 0 (0x0).

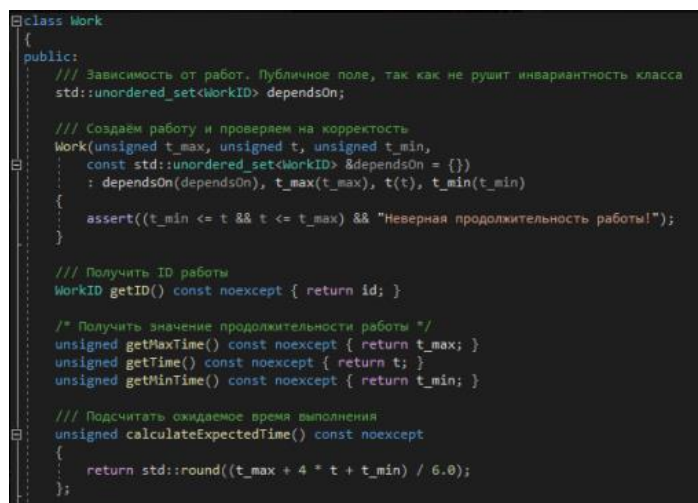
Рисунок 12 – Сборка

– анализ полученных результатов. Проверка на их выполнимость и правильностью полученных результатов [8][19][20][21].

Приступим к реализации программы.

Подключив все необходимые библиотеки среды, приступаем к написанию основных классов программы.

На рисунке 13 представлен Class Work.



```
class Work
{
public:
    /// Зависимость от работ. Публичное поле, так как не рушит инвариантность класса
    std::unordered_set<WorkID> dependsOn;

    /// Создаем работу и проверяем на корректность
    Work(unsigned t_max, unsigned t, unsigned t_min,
        const std::unordered_set<WorkID> &dependsOn = {})
        : dependsOn(dependsOn), t_max(t_max), t(t), t_min(t_min)
    {
        assert((t_min <= t && t <= t_max) && "Неверная продолжительность работы!");
    }

    /// Получить ID работы
    WorkID getID() const noexcept { return id; }

    /* Получить значение продолжительности работы */
    unsigned getMaxTime() const noexcept { return t_max; }
    unsigned getTime() const noexcept { return t; }
    unsigned getMinTime() const noexcept { return t_min; }

    /// Подсчитать ожидаемое время выполнения
    unsigned calculateExpectedTime() const noexcept
    {
        return std::round((t_max + 4 * t + t_min) / 6.0);
    }
};
```

Рисунок 13 – Class Work

Class Work реализует информацию о работе. Он содержит создание работ и проверка их на корректность, получение ID работы, получение значения продолжительности работы, подсчитывает ожидаемое время выполнения, подсчитывает дисперсию случайной величины, устанавливает уникальный ID работы.

На рисунке 14 представлен Class Works.

```
/// Таблица работ. Нужна, чтобы правильно разрешать зависимости работ
class Works
{
public:
    Works() = default;

    /// Добавляем работу
    void addWork(Work work) noexcept
    {
        /// Меняем ID на правильное
        work.id = works.size() + 1;
        works.emplace_back(std::move(work));
    }

    /// Получить работу по ID (считая с 1)
    const Work *getWork(WorkID id) const noexcept
    {
        if (1 <= id && id <= works.size()) { return &works[id-1]; }
        return nullptr;
    }
};
```

Рисунок 14 – Class Works

Class Works реализует создание таблицы работ. Он содержит добавление работ, получение работ по ID.

На рисунке 15 изображён Class Net.

```
/// Сетевой график
class Net
{
public:
    /// Начало или конец задач
    struct Event
    {
        unsigned early_start = 0; /// Раннее время начала
        /// Индекс предыдущего события,
        /// которое повлияло на раннее время начала
        EventIndex prev_index = 0;
    };

    /// Часть критического пути
    struct CriticalTask
    {
        TaskIndex index; /// Индекс задачи
        bool is_fake = false; /// Является ли задача фальшивой?
    };
    CriticalTask criticalTask; /// Часть критического пути, что привела в это событие

    std::unordered_set<TaskIndex> out; /// Реальные индексы исходящих задач
    std::unordered_set<TaskIndex> in; /// Реальные индексы входящих задач
    std::unordered_set<FakeTaskIndex> fake_out; /// Индексы фальшивых исходящих задач
    std::unordered_set<FakeTaskIndex> fake_in; /// Индексы фальшивых входящих задач
};
```

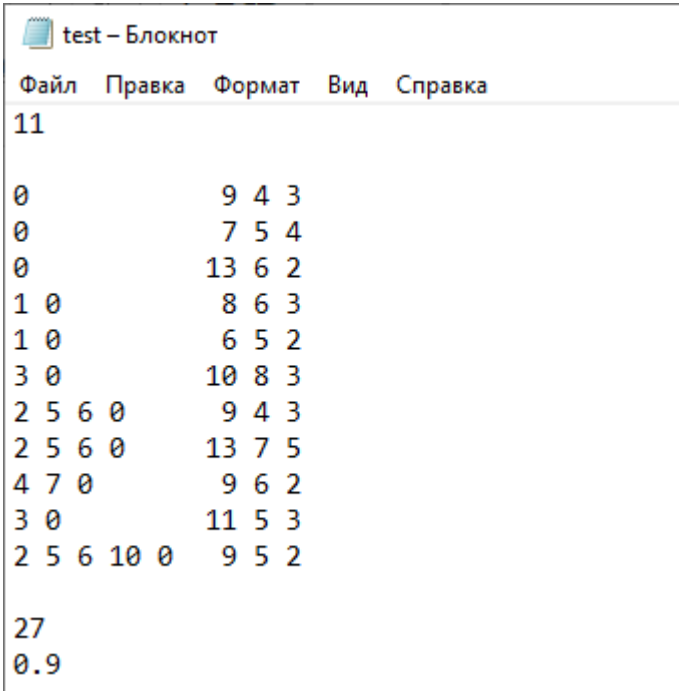
Рисунок 15 – Class Net

Class Net реализует сетевой график. Он содержит получение критического пути, создание сети, перевод ID работы в индекс задачи, получение событий начала задачи, получение событий конца задачи, подсчёт критического пути. Пример кода представлен в приложении А.

Приступим к запуску программы:

При запуске программы нам необходимо ввести входные данные для расчёта задачи: количество работ, наиболее вероятная оценка продолжительности работ, оптимистическая оценка продолжительности работ, пессимистическая оценка продолжительности работ, директивный срок выполнения проекта, заданная надёжность.

На рисунке 16 представлен ввод данных через текстовый документ.



```
test - Блокнот
Файл  Правка  Формат  Вид  Справка
11
0          9 4 3
0          7 5 4
0          13 6 2
1 0        8 6 3
1 0        6 5 2
3 0        10 8 3
2 5 6 0    9 4 3
2 5 6 0    13 7 5
4 7 0      9 6 2
3 0        11 5 3
2 5 6 10 0 9 5 2

27
0.9
```

Рисунок 16 – Ввод данных

На рисунке 17 представлен Вывод программы.


```
Консоль отладки Microsoft Visual Studio
Critical time: 26
Critical path: b3 b6 b7 b9
Critical path dispersion: 7.08333
Critical path standard deviation: 2.66145
Enter directive time: 27
Enter accuracy: 0.9
Probability: 60%
Z: 1.29
Max dT: 3.43327
Max time with 0.9 accuracy: 30
```

Рисунок 17 – Вывод программы

На выходе программы выводит критическое время и критический путь, а также вероятность выполнения проекта в заданный срок. Также выводит время выполнения проекта с заданной надёжностью.

3.3 Проверка программы на аналогах решения задачи анализа и оптимизации сетевых графиков

Чтобы проверить разработанную программу на точность значений, рассмотрим пример на приведённом выше аналоге.

На рисунке 18 представлен ввод данных.

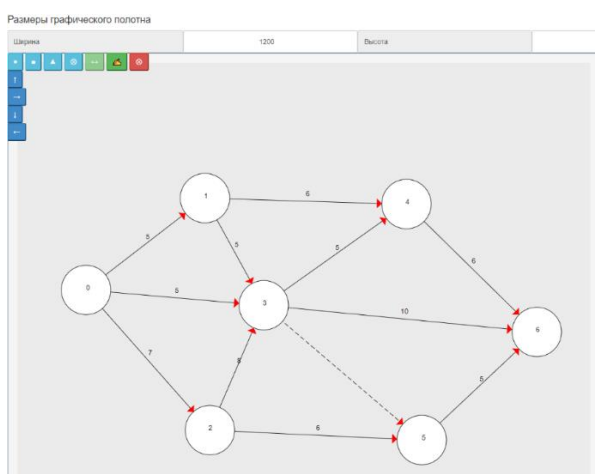


Рисунок 18 – Ввод данных

Ввод данных происходит путём построения сетевого графика, который мы строили во второй главе.

На рисунке 19 изображены параметры сетевого графика.

Рисунок 19 – Параметры сетевого графика

Также выделяются параметры сетевого графика, которые нам необходимы.

Расчёт критического времени:

$i=0$ из этого следует, что $t_p=0$, рассчитаем и для других начальных событий

$$i=1 \quad t_p(0) + t(0,1) = 0+5 = 5;$$

$$i=2 \quad t_p(0) + t(0,2) = 0+7 = 7;$$

$$i=3 \quad \max(t_p(0) + t(0,3); t_p(1)+t(1,3); t_p(2)+t(2,3)) = \max(0+5; 5+5; 7+8) = 15;$$

$$i=4 \quad \max(t_p(1) + t(1,4); t_p(3)+t(3,4)) = \max(5+6; 15+5) = 20;$$

$$i=5 \quad \max(t_p(2) + t(2,5); t_p(3)+t(3,5)) = \max(7+6; 15+0) = 15;$$

$$i=6 \quad \max(t_p(3) + t(3,6); t_p(4)+t(4,6); t_p(5)+t(5,6)) = \max(15+10; 20+6; 15+5) = 26;$$

Критическое время составляет 26. Мы убедились, что критическое время совпадает [5].

Таблица 8 – Расчет резерва времени событий

Номер события	Сроки свершения событий - ранний	Сроки свершения событий - поздний	Резерв времени
0	0	0	0
1	5	10	5
2	7	7	0
3	15	15	0
4	20	20	0
5	15	21	6
6	26	26	0

Проведем анализ сетевой модели по времени.

Таблица 9 – Исходные данные

Работа	Предшествующие работы	Продолжительность
0,1	0	5
0,2	0	7
0,3	0	5
1,3	1	5
1,4	1	6
2,3	1	8
2,5	1	6
3,4	3	5
3,5	3	0
3,6	3	10
4,6	2	6
5,6	2	5

Таблица 10 – Расчет ранних и поздних сроков, а также резерва времени

Ранние сроки		Поздние сроки		Резервы времени	
Начало	Окончание	Начало	Окончание	Полный	Независимый
0	5	5	10	5	0
0	7	0	7	0	0
0	5	10	15	10	10

Продолжение таблицы 10:

Ранние сроки		Поздние сроки		Резервы времени	
Начало	Окончание	Начало	Окончание	Полный	Независимый
5	10	10	15	5	0
5	11	14	20	9	4
7	15	7	15	0	0
7	13	15	21	8	2
15	20	15	20	0	0
15	15	21	21	6	0
15	25	16	26	1	1
20	26	20	26	0	0
15	20	21	26	6	0

Таким образом, критический путь у работы будет следующий: 0 →2 →3 →4 →6

Мы убедились, что программа работает правильно. Она позволяет решать задачи анализа и оптимизации сетевого графика в условиях неопределённости.

Выводы к главе 3

Третья глава ВКР посвящена анализу и обзору сервисов и инструментов необходимых для реализации программного приложения оптимизации сетевых проектов. В качестве средств разработки выбрана среда Microsoft Visual Studio. Реализованное приложение позволяет определять критическое время и топологию критического пути, а также вычислять вероятностные характеристики сетевых проектов.

Заключение

Выпускная квалификационная работа была посвящена проблеме анализа и оптимизации сетевых графиков в условиях неопределённости.

В процессе работы над ВКР решены следующие задачи:

- изучены математические методы решения сетевых задач;
- выполнена оптимизация сети в условиях неопределённости и оценены её основные параметры;
- построен алгоритм решения сетевой задачи в условиях неопределённости и реализован программный модуль.

Все задачи в выпускной квалификационной работе были выполнены.

В первой главе выпускной квалификационной работы представлен алгоритм нахождения критического времени выполнения проекта, а также изложен метод построения топологии критического пути. Кроме того, в первой главе приведено обоснование того, что основной параметр сетевой задачи – время выполнения отдельной работы подчиняется β -распределению.

Во второй главе решена конкретная сетевая задача в предположении, что время выполнения отдельной задачи подчиняется β -распределению, а случайные величины критическое время и критический путь распределены нормально. Найдено ожидаемое время выполнения проекта, определена вероятность выполнения проекта не позднее заданного срока, найден интервал гарантированного времени выполнения проекта, определён максимально возможный срок выполнения проекта с заданной надёжностью.

В третьей главе разработана программа для построения критического пути, а также для вычисления критического времени и вероятностных характеристик основных параметров сети, выполнен анализ и обзор сервисов и инструментов необходимых для реализации программного приложения. В качестве средств разработки использовалась среда Microsoft Visual Studio на языке C++.

Список используемой литературы и используемых источников

1. Анализ и оптимизация сетевого графика [Электронный ресурс]. URL: <https://lektsii.org/8-29566.html>
2. Илларионов, В.А. Сетевое планирование: учебное пособие / Казань 2013. 45 с.
3. Кремер Н.Ш. Исследование операций в экономике. Москва. 2002. – 413 с.
4. Лафоре.Р. Объектно-ориентированное программирование в С++ 4 издание. 2004. – 924 с.
5. Онлайн-калькулятор по сетевому планированию [Электронный ресурс]. URL: https://math.semestr.ru/setm/setm_manual.php
6. Плескунов М. А. Задачи сетевого планирования: учебное пособие / Екатеринбург: Изд-во Урал. ун-та, 2014. 92 с.
7. Расчёт и анализ стохастических сетей [Электронный ресурс]. URL: https://bstudy.net/815849/ekonomika/raschet_analiz_stohasticheskikh_setey
8. Руководство по Visual Studio [Электронный ресурс]. URL: <https://docs.microsoft.com/ru-ru/cpp/get-started/?view=msvc-170>
9. Сетевые модели в условиях неопределённости [Электронный ресурс]. URL: https://studref.com/548856/ekonomika/setevye_modeli_usloviyah_neopredelennosti
10. Сетевое планирование / Вероятностные сети [Электронный ресурс]. URL: <https://www.bibliofond.ru/view.aspx?id=663171#text>
11. Сетевое планирование [Электронный ресурс]. URL: <http://upr-proektom.ru/setevoe-planirovanie>
12. Сетевое планирование в условиях неопределённости [Электронный ресурс]. URL: <https://www.stud24.ru/mathematic-modelling/setevoe-planirovanie-v-usloviyah-neopredelennosti/97257-291136-page4.html>
13. Сиддхартха Рао. Освой язык самостоятельно: учебное пособие. Москва, Санкт-Петербург. 2013. - 651 с.

14. Сидорина Т. Visual Studio C++ - учебное пособие. Санкт-Петербург, 2009. 843 с.
15. Сосина Н.А. Исследование операций / Электронное учебное пособие. В 2-х частях. Часть II / Тольятти: ФГБОУ ВО «Тольяттинский государственный университет», 2022г.
16. Спиридонов Э.С., Телятникова Н. А. Сетевое планирование в строительстве: Учебное пособие, М.: РУТ (МИИТ), 2018. 75 с.
17. Таирова, Е.В. Методы сетевого планирования в организации комплексов работ: учебное пособие / Е.В. Таирова. – Иркутск: ИрГУПС, 2007. – 95 с.
18. 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>
19. C++ in Visual Studio [Электронный ресурс]. URL: <https://docs.microsoft.com/en-us/cpp/overview/visual-cpp-in-visual-studio?view=msvc-170>
20. C++ solved programming programs [Электронный ресурс]. URL: <https://www.includehelp.com/cpp-programming-examples-solved-cpp-programs.aspx>
21. Graph Emplementation in C++ Using Adjacency List [Электронный ресурс]. URL: <https://www.softwaretestinghelp.com/graph-implementation-cpp/>
22. Graph Emplementation in C++ [Электронный ресурс]. URL: <https://stackoverflow.com/questions/5493474/graph-implementation-c>
23. Network Planning [Электронный ресурс]. URL: <https://www.manageengine.com/network-monitoring/network-planning.html>

Приложение А Фрагмент программного кода

```
/// Сетевой график
class Net
{
public:
    /// Начало или конец задач
    struct Event
    {
        unsigned early_start = 0; /// Раннее время начала
        /// Индекс предыдущего события,
        /// которое повлияло на раннее время начала
        EventIndex prev_index = 0;

        /// Часть критического пути
        struct CriticalTask
        {
            TaskIndex index; /// Индекс задачи
            bool is_fake = false; /// Является ли задача фальшивой?
        };
        CriticalTask criticalTask; /// Часть критического пути, что привела в
это событие

        std::unordered_set<TaskIndex> out; /// Реальные индексы исходящих
задач
        std::unordered_set<TaskIndex> in; /// Реальные индексы входящих
задач
        std::unordered_set<FakeTaskIndex> fake_out; /// Индексы
фальшивых исходящих задач
        std::unordered_set<FakeTaskIndex> fake_in; /// Индексы фальшивых
входящих задач
    };
};
```


Продолжение Приложения А

```
};  
    // Однопараметрическая работа. Назвал Task для того, чтобы не  
перепутать  
    struct Task  
    {  
        WorkID id = 0; // Соответствующее ID работы. Если == 0 -> задача  
фальшивая  
        unsigned time = 0; // Время на выполнение задачи  
  
        EventIndex start; // Индекс события начала  
        EventIndex finish; // Индекс события конца  
    };  
    // Создаём сеть из трехпараметрических работ  
Net(const Works &works) : events(1)  
    {  
        for (const auto &work : works)  
        {  
            std::unordered_set<TaskIndex> dependsOn;  
            for (auto id : work.dependsOn)  
            {  
                dependsOn.insert(idToIndex(id));  
            }  
            createTask(work.getID(),                work.calculateExpectedTime(),  
dependsOn);  
        }  
    }  
    /* Все события должны сходиться в 1 финальное */  
    EventIndex finish_index = events.size();  
    events.push_back(Event{ });  
    for (EventIndex index = 0; index < finish_index; ++index)
```

Продолжение Приложения А

```
{
    /// Событие никуда не ведёт (лист в сети)
    if (events[index].fake_out.empty() && events[index].out.empty())
    {
        wireEvents(index, finish_index);
    }
}

/// Перевести ID работы в индекс задачи. Просто уменьшает на 1 id
TaskIndex idToIndex(WorkID id) const noexcept
{
    assert(id > 0 && "Incorrect id");
    return id - 1;
}

/// Получить реальную задачу
const Task *getTask(TaskIndex index) const noexcept
{
    if (index >= tasks.size()) { return nullptr; }
    return &tasks[index];
}

/// Получить фальшивую задачу
const Task *getFakeTask(FakeTaskIndex index) const noexcept
{
    if (index >= fake_tasks.size()) { return nullptr; }
    return &fake_tasks[index];
}

/// Получить событие
const Event *getEvent(EventIndex index) const noexcept
{

```

Продолжение Приложения А

```
if (index >= events.size()) { return nullptr; }
    return &events[index];
}
/// Получить событие начала задачи
const Event *getTaskStartEvent(TaskIndex index) const noexcept
{
    if (index >= tasks.size()) { return nullptr; }
    return &events[tasks[index].start];
}
/// Получить событие конца задачи
const Event *getTaskFinishEvent(TaskIndex index) const noexcept
{
    if (index >= tasks.size()) { return nullptr; }
    return &events[tasks[index].finish];
}
/// Получить событие начала фальшивой задачи
const Event *getFakeTaskStartEvent(FakeTaskIndex index) const
noexcept
{
    if (index >= fake_tasks.size()) { return nullptr; }
    return &events[fake_tasks[index].start];
}
/// Получить событие конца фальшивой задачи
const Event *getFakeTaskFinishEvent(FakeTaskIndex index) const
noexcept
{
    if (index >= fake_tasks.size()) { return nullptr; }
    return &events[fake_tasks[index].finish];
}
```

Продолжение Приложения А

```
/// Получить событие начала
const Event &getStartEvent() const noexcept
{
    // Гарантированно существует
    return events.front();
}

/// Получить событие конца
const Event &getFinishEvent() const noexcept
{
    // Гарантированно существует
    return events.back();
}

/// Критическое время выполнения
unsigned criticalTime() const noexcept
{
    return events.back().early_start;
}

/// Критический путь
std::vector<Event::CriticalTask> criticalPath() const noexcept
{
    std::vector<Event::CriticalTask> path;
    const Event *event = &events.back();
    while (event->prev_index != 0)
    {
        path.push_back(event->criticalTask);
        if (event->criticalTask.is_fake)
        {
            event = getFakeTaskStartEvent(event->criticalTask.index);
        }
    }
}
```

Продолжение Приложения А

```
else
    {
        event = getTaskStartEvent(event->criticalTask.index);
    }
}
// Не забываем часть пути в начало
path.push_back(event->criticalTask);

// Делаем путь сначала, а не с конца
std::reverse(path.begin(), path.end());
return path;
}

private:
/// События начала и конца
std::vector<Event> events;
/// Реальные задачи
std::vector<Task> tasks;
/// Фальшивые задачи. Нужны для синхронизации реальных задач
std::vector<Task> fake_tasks;
/// Добавить новую задачу в конец
TaskIndex createTask(WorkID id, unsigned time, const
std::unordered_set<TaskIndex> &dependsOn)
{
    TaskIndex index = tasks.size();
    Task task;
    task.id = id;
    task.time = time;
    task.start = resolveDependencies(dependsOn);
    task.finish = createFinishEvent(index, task);
```

Продолжение Приложения А

```
tasks.push_back(std::move(task));
    return index;
}
/// Разрешаем зависимости от других задач
EventIndex resolveDependencies(const std::unordered_set<TaskIndex>
&dependsOn)
{
    /// Нет зависимостей -> задача стартует с 0
    if (dependsOn.empty()) { return 0; }
    /// Только одна зависимость -> стартуем с её окончания
    if (dependsOn.size() == 1) { return tasks[*dependsOn.begin()].finish; }
    /* Много зависимостей -> создаём событие синхронизации */
    EventIndex tasks_done_index = events.size();
    events.push_back(Event{ });
    for (auto task_index : dependsOn)
    {
        wireEvents(tasks[task_index].finish, tasks_done_index);
    }
    return tasks_done_index;
}
/// Совместить 2 события фальшивой работой
void wireEvents(EventIndex start_index, EventIndex finish_index)
{
    /* Создаём фальшивую работу, что не занимает времени */
    TaskIndex fake_task_index = fake_tasks.size();
    Task fake_task;
    fake_task.start = start_index;
    fake_task.finish = finish_index;
    fake_tasks.push_back(std::move(fake_task));
}
```

Продолжение Приложения А

```
Event &start = events[start_index];
    start.fake_out.insert(fake_task_index);
    Event &finish = events[finish_index];
    finish.fake_in.insert(fake_task_index);
    /// Ранний старт определяется как максимум из ранних стартов
    if (finish.early_start < start.early_start)
    {
        finish.early_start = start.early_start;
        finish.prev_index = start_index;
        finish.criticalTask = Event::CriticalTask{
            fake_task_index,
            true
        };
    }
}
/// Создаём событие окончания задачи
EventIndex createFinishEvent(TaskIndex index, const Task &task)
{
    events[task.start].out.insert(index);
    Event finish_event;
    finish_event.early_start = events[task.start].early_start + task.time;
    finish_event.prev_index = task.start;
    finish_event.in = {index};
    finish_event.criticalTask.index = index;
    events.push_back(std::move(finish_event));
    return events.size() - 1;
}
};
```

