МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФГБОУ ВО «Тверской государственный университет»

Факультет прикладной математики и кибернетики

Направление«Прикладная информатика»

Профиль «Прикладная информатика в экономике»

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**(ВЫПУСКНАЯ РАБОТА БАКАЛАВРА)**

Сетевое планирование в условиях неопределённости

Автор:

Руденок Михаил Александрович

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Научный руководитель:

д-р физ.-мат. наук, проф.

Соломаха Геннадий Михайлович

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Допущена к защите:

Руководитель ООП:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С.М. Дудаков

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_2021 г.

Тверь – 2021

Оглавление

[**Введение** 2](#_Toc73952064)

[**1.** **Сетевое планирование и правила построения сетевого графика** 2](#_Toc73952065)

[1.1. Основные понятия сетевой модели 2](#_Toc73952066)

[1.2. Правила построения сетевой модели 2](#_Toc73952067)

[1.3. Правила упорядочивания сетевого графика 2](#_Toc73952068)

[**2. Сетевое планирование в условиях неопределённости** 2](#_Toc73952069)

[2.1. Понятие и виды неопределённости в сетевом графе. 2](#_Toc73952070)

[2.2 Сравнительный анализ методов сетевого планирования. 2](#_Toc73952071)

[**3. Программная реализация сетевого графа** 2](#_Toc73952072)

[**Заключение** 2](#_Toc73952073)

[**Список литературы** 2](#_Toc73952074)

[**ПРИЛОЖЕНИЕ** 2](#_Toc73952075)

**Введение**

Перед началом планирования сложных процессов люди обычно используют план для того, чтобы структурировать и оценить время исполнения работы в целом и каждого отдельного подэтапа выполнения работы. Цель сетевого планирования - построить модель реализации некоторого проекта наиболее логично, правильно, эффективно, с наименьшими затратами. Для реализации такого подхода необходимо сформировать комплекс работ, задать их очередность, определить необходимые ресурсы, сформулировать задачи, которые должны быть решены в заданной последовательности для достижения конечной цели проекта. В итоге необходимо добиться сокращения до минимума продолжительности проекта и затрат на его выполнение. Благодаря сетевой модели можно понять какие именно работы необходимо произвести для получения желаемого результата, сколько людей будет задействовано, какое количество ресурсов необходимо и сколько это примерно займет по времени.

Очевидно, что в реальной жизни для планирования столь сложной работы не всё всегда точно известно. Например, не ясно сможет ли соисполнитель в требуемый срок поставить некоторый материал, или предоставить определенную услугу. По неизвестным причинам может произойти задержка перевозки груза. Наконец, может произойти и форсмажор, которого. скорее всего, нельзя запланировать заранее. Иными словами, в реальной жизни всегда имеет место элемент неопределенности. Не учитывать его нельзя, ибо рано или поздно произойдет провал запланированного дела, с соответствующими материальными, и иными, издержками и потерями, вплоть до полного банкротства предприятия.

По указанным причинам тема работы: «Сетевое планирование в условиях неопределенности» является чрезвычайно актуальной.

Цель работы исследование вопросов сетевого планирования в условиях неопределённости.

В рамках поставленной цели, в работе решаются следующие задачи:

* исследование и проведение сравнительного анализа существующих методов сетевого планирования;
* анализ существующих моделей сетевого планирования;
* анализ известных подходов к сетевому планированию в условиях неопределенности;
* разработка алгоритма и компьютерной программы, реализующей оптимальное время выполнения процесса сетевого планирования в условиях неопределенности;
* разработка имитационной модели для проверки исследований;
* анализ результатов моделирования.

**Объектом исследования -** Сетевое планирование.

**Предметом исследования -**благоустройства города Тверь.

В первой главе определено содержание методов сетевого планирования, правила построения и упорядочивания, временные параметры сетевого графика.

Во второй главе даны основные определения сетевого графика в условиях неопределённости, проанализированы основные модели сетевого планирования, в частности детерминированные модели (Ганта и метод критического пути) и вероятностные модели (метод оценки и анализа программ, метод имитационного моделирования или метод Монте-Карло, метод графической оценки и анализа программ) и проведён их сравнительный анализ.

В третьей главе описана и разработана с использованием пакета прикладных программ MATLAB для более точной оценки времени каждой отдельной работы и времени выполнения всего сетевого плана.

1. **Сетевое планирование и правила построения сетевого графика**

Сетевое планирование (сетевой анализ) — совокупность прикладных методов [управления проектами](https://ru.wikipedia.org/wiki/%D0%A3%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B0%D0%BC%D0%B8), обеспечивающая планирование, анализ сроков выполнения частей проектов, с увязкой выполнения различных работ и процессов во времени.

* 1. Основные понятия сетевой модели

**Сетевая модель** — это план выполнения определенной совокупности зависимых работ, данный в особенной форме сети.

**Сетевой график** – это графическое представление сетевой модели

Понятие работа применяется в СПУ во всеобъемлющим смысле.

* **Действительная работа** – это длительный процесс, для которого нужны определенные траты на материалы. Для каждой действительной работы должен быть определен ответственный и она должна быть точно описана
* **Ожидание** – это длительный процесс, для которого не требуются траты на труд.
* **Зависимость** – это логическая связь между определёнными работами, не требующая затрат на труд, ресурсы или время выполнения. Она показывает, что одна работа прямо зависит от вывода другой работы.

**Событие** – это момент окончание любого из процессов, показывающий определённый этап реализации проекта. Так же событие может быть, как результатом отдельной работы, так и итоговым результатом нескольких работ. Событие совершится только если все предыдущие работы уже закончены. Последующие работы начнутся только, когда событие произойдет. У события нет продолжительности работ и совершается оно без задержки.

Так же стоит отметить исходное и завершающее события.

Исходное событие не имеет предшествующих работ и событий.

У завершающего события нет последующих работ и событий.

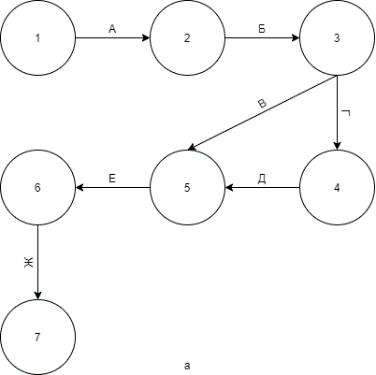
Все события на сетевом графике изображаются кружками

Работы – стрелками

Изображение выглядит как текст

Автоматически созданное описание

Рис. 1. Пример реализации события в сетевом графе



Изображение выглядит как текст, часы

Автоматически созданное описание

Рис. 2

На рисунке 2 , а изображен сетевой график задачи моделирования и построения оптимального плана. Для решения данной задачи необходимо использовать следующие работы :

А – Формулировка проблемы исследования

Б – Постройка математическую модель

В – Сбор информации

Г – Выбрать метод решения задачи

Д – Запрограммировать и отладить программу

Е – Подсчитать оптимальный план

Ж – Отдать результаты работы заказчику

События обозначены цифрами на графике к ним приводит выполнения соответствующих работ.

На графике видно, что работы В и Г можно начать выполнять независимо друг от друга, но только после окончания события 3 то есть после выполнения работ А и Б. Работу Д -после окончания события 4, когда закончены работы А,Б,В,Г , а работу Е выполнить после установления события 5 то есть при выполненных всех предыдущих работ А,Б,В,Г,Д

Сеть называется структурной если в сетевой модели нет числовых оценок.

* 1. Правила построения сетевой модели

Сетевые графики строятся на первом этапе планирования. Изначально составляется процесс и перечь работ и событий, рассчитывают их логические связи и последовательность, в которой они будут выполнятся. С их помощью оценивается продолжительность работ. Далее составляется сетевой график. После построения сетевого графика считают параметры события и работ, выбирают резервы времени и критический путь. Далее проводят анализ и оптимизацию сетевого графика. При надобности график вычеркивается и заново составляется с учётом пересчитанных параметров событий и работ.

Правила, которые необходимо соблюдать при построении графика:

1. Не должно быть “Тупиковых” событий в сетевой модели. То есть событий, из которых не выходит ни одна из работ, кроме завершающего события

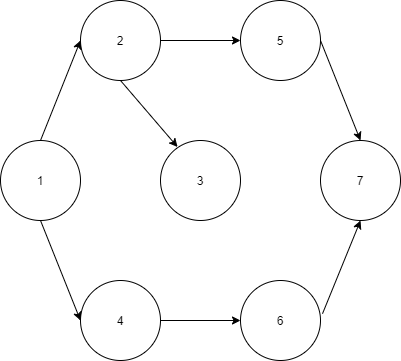


Рис.3. Сетевая модель с “тупиковым” событием

1. Не должно быть “хвостовых” событий в сетевом графике (если это только не исходное), которым не предшествует ни одна из работ.

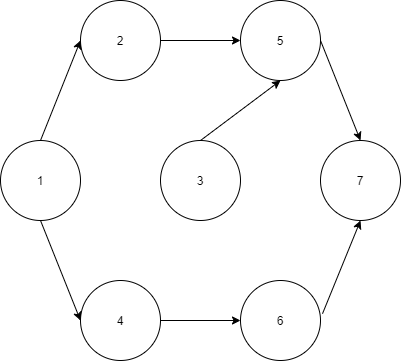


Рис.4. Сетевая модель с “хвостовым” событием

1. Не должно быть замкнутых контуров и петель, то есть путей, соединяющих определённые события с ними же.

Изображение выглядит как текст, спортивная игра

Автоматически созданное описание

Рис.5. Сетевая модель с замкнутым событием и петлёй

1. Каждые два события должны быть непосредственно связаны не более одной стрелкой-работой. При нарушении данного правила может произойти путаница, так как две разные работы имеют одно и то же обозначение. Для избежание данной проблемы рекомендуется вводить фиктивные события. Данные события не требуют времени выполнения и обозначаются пунктирными линиями

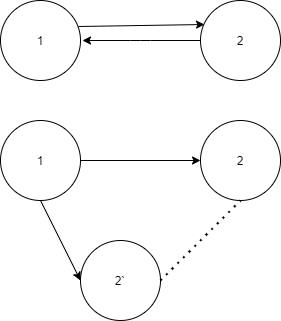


Рис.6. Сетевая модель с нарушением в построении и его исправленная копия с вводом фиктивного события

1. Рекомендуется в сети иметь одно начальное и одно заключительное событие. Если в получившейся сети не так, то следует вводить фиктивную работу.

Фиктивная работа так же может помочь при отражении зависимости событий, не связанных с реальными работами. Или для того, чтобы отражать реальную отсрочку и ожидание.

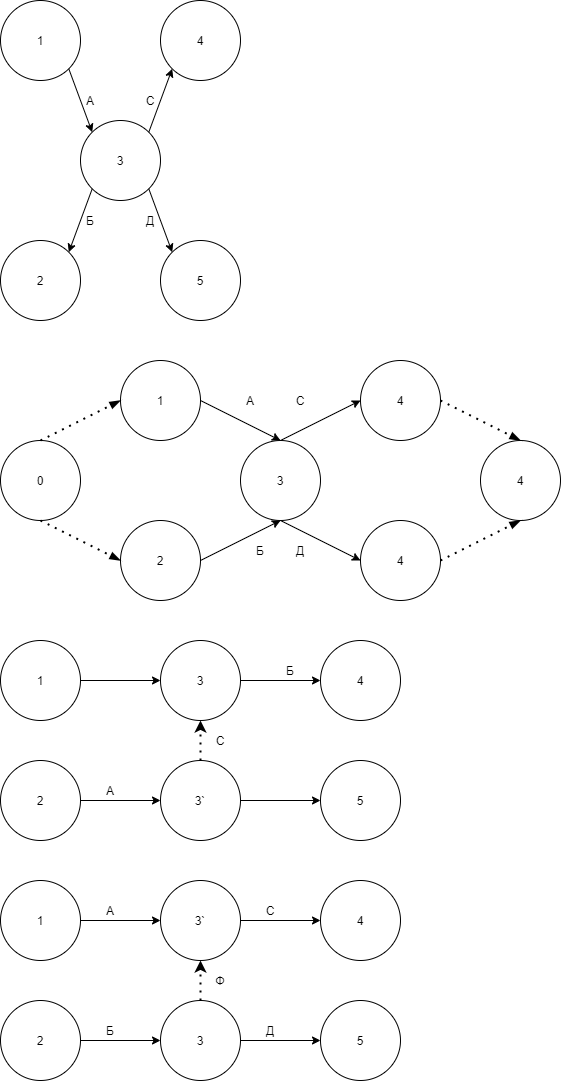


Рис.7. Сетевая модель без использования рекомендации и с использованием

1.3. Правила упорядочивания сетевого графика

Допустим, что при реализации определённого проекта имеется 12 событий:0,1,2,3,4,5,6,7,8,9,10,11 и 24 и работы, с которыми они связаны:

(0,1), (0,2), (0,3), (1,2), (1,4), (1,5), (2,3), (1, 5), (2, 3), (2, 5), (2, 7), (3, 6), (3, 7), (3, 10), (4, 8), (5, 8), (5, 7), (6, 10), (7, 6), (7, 8), (7, 9), (7, 10), (8, 9), (9, 11), (10, 9), (10, 11).Нужно составить и упорядочить сетевой график.

Начальным событием сетевого графика значится событие 0 (так как данному события не предшествует ни одна из работ), а оканчивающим – событие 11 (так как за данным событием не следует ни одна из работ). Полагая на сетевом графике изменение времени слева направо, поместим событие 0 в левую часть графика, а событие 11 в правую часть, предварительно расположив между ними промежуточные события в соответствии их номерам. Работы-стрелки будут связывать наши события в соответствие с перечнем работ.

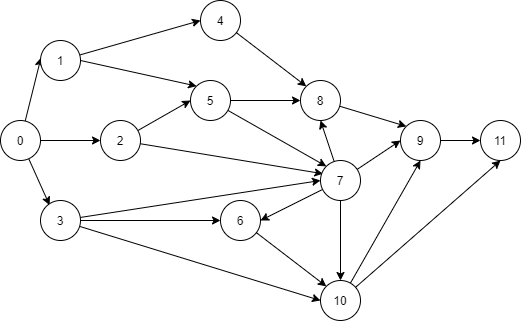


Рис.8 Построенный сетевой график

Упорядочение сетевого графика состоит в том, чтобы расположить события и работы в таком порядке, чтобы для любой работы предшествующее ей событие, расположенное левее, и имеет меньший номер по сравнению с событием, завершающим эту работу.

Иными словами, в упорядоченном сетевом графике все работы-стрелки идут слева направо: от события с наименьшим номером к событиям с большим номером.

Для начала разобьём сетевой график на пару вертикальных слоев (их мы обведём пунктирной линией и обозначим римскими цифрами)

В первый слой поместим начальное событие 0 и вычеркнем его из графика и все работы-стрелки, выходящие из него. Без события I и без входящих стрелок у нас образуется II слой. Мысленно вычеркнув событие 1 и все выходящие работы, увидим, что события 4 и 2 образуют III слой. Продолжаем данную процедуру вычеркивая и получая слои: IV с событиями 5 и 3, V с событием 7, VI с событием 8 и 6, VII с событием 10,

VIII с событием 9 и IX с событием 11

Изображение выглядит как белый, черный, цепь, пар

Автоматически созданное описание

Рис. 9 сетевой график с вертикальными слоями

На данном графике видим, что изначальная нумерация событий не совсем корректная так как событие 6 в VI слое имеет номер меньший, чем событие 7. Исправив нумерацию событий в соответствии с их расположением на графике сможем получить упорядоченный график.

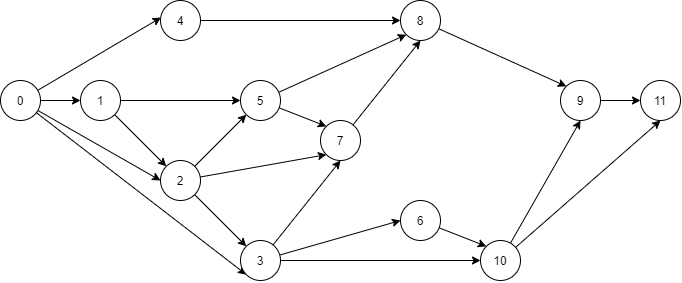


Рис. 10 Упорядоченный график

**Путь** –любая последовательность работ, в которой заключительное событие каждой работы совпадает с начальным событием следующей за ней работы. Есть различные пути сетевого графика, но наибольший интерес видеться полный путь L – любой путь, который начало которого совпадает с изначальным событием сети, а конец – с конечным.

**Критический путь** – это наиболее длительный путь в сетевом графике. Данное определение распространяется и на работы и события, расположенные на этом пути.

Выводы.

1. Сетевое планирование представляет собой совокупность прикладных методов управления проектами, обеспечивающая планирование, анализ сроков выполнения частей проектов, с увязкой выполнения различных работ и процессов во времени.

2. Основными правилами построения сетевого графика являются:

- сетевые графики строятся на первом этапе планирования;

- при построении графика не должно быть “Тупиковых” событий то есть событий, из которых не выходит ни одна из работ, кроме завершающего события;

- не должно быть “хвостовых” событий

- не должно быть замкнутых контуров и петель

- каждые два события должны быть непосредственно связаны не более одной стрелкой-работой;

3. Сетевой график должен быть упорядочен по определённым правилам:

- начальному событию сетевого графика не должно предшествовать ни одна из работ;

- расположить события и работы следует в таком порядке, чтобы для любой работы предшествующее ей событие, располагалось левее, и имело меньший номер по сравнению с событием, завершающим эту работу;

**2. Сетевое планирование в условиях неопределённости**

2.1. Понятие и виды неопределённости в сетевом графе.

Неопределённость — это отсутствие или недостаток информации о чём-либо. Неопределенность в моделировании имеет свою специфику. Следует различать ошибку моделирования и неопределенность.

Источники ошибок моделирования: сознательно введенные исследователем (например, аппроксимация математической модели) и возникшие в процессе моделирования.

Неопределенность вызвана недетерминированными источниками, например, внешней средой, лицом, принимающим решение, форсмажорными обстоятельствами. Анализ неопределенности при моделировании крайне важная задача, обеспечивающая получение требуемого качества работы.

Факторы, влияющие на неопределённость:

• Неполнота данных — это неточность информации о параметрах, ситуациях, требующих выбора оптимального решения. Когда невозможно адекватно и точного учёта информации.

• Фактор случайности — это фактор, возникновение которого невозможно предусмотреть заранее даже в вероятностной оценке.

В сетевом планировании существует несколько типов неопределённости:

• статистическая неопределённость;

• стохастическая неопределённость.

Статистическая неопределённость – это неопределённость возникает в связи с неопределенными свойствами исходной информации, необходима для выполнения расчётов по определенному типу данных.

При стохастической неопределённости известны только вероятностные характеристики входящих в задачу случайных факторов.

Неопределенности могут возникать по причинам нечетко сформулированных целей функционирования системы, неопределенности определения её структуры, неопределенности исходных данных.

Все виды неопределенностей необходимо по возможности компенсировать. Для этого на этапе планирования:

- следует исключить недостижимые цели и принять все возможные меры для четкого определения слабо определённых промежуточных целей;

- учесть вероятности выхода из строя физических элементов системы: возможность поломки машин, механизмов, аварий и т.п.;

- учесть возможность изменения ситуации в системе верхнего уровня (прогнозирование внешней среды). Так, например, следует учесть возможнотсь повышения цен на топливо, если такая тендениця имеет место систематически, инфляцию используемой валюты, принятие законодательных актов, изменяющих ценообразование, налоговую политику;

- следует учесть предсказуемость клиентов и исполнителей.

2.2 Сравнительный анализ методов сетевого планирования.

Методы в сетевом планировании разделяются на два типа, это детерминированные и вероятностные.

Детерминированные, это те модели в которых продолжительность и последовательность работ задана однозначно. Одна из самых популярных детерминированных диаграмм является диаграмма Ганта и метод критического пути.

Вероятностные модели, это такие модели, где продолжительность работ точна неизвестна или могут возникнуть непредвиденные ситуации такие как: погодные условия, ненадежные поставщики, результат научного эксперимента. Вероятностные модели делятся на два типа: неальтернативные и альтернативные.

Неальтернативные модели, это модели в которых зафиксирована последовательность выполнения работ, а продолжительность работ определяется функциями распределения.

Альтернативные модели, это модели в которых продолжительность и связи между всеми или некоторыми работами носят вероятностный характер.

Детерминированные модели имеет смысл применять, когда продолжительность и последовательность работ в сетевом графе задана однозначно.

Рассмотрим три распространенных метода вероятностного сетевого планирования:

* метод оценки и анализа программ (PERT);
* метод имитационного моделирования или метод Монте-Карло;
* метод графической оценки и анализа программ (GERT).

2.2.1 Метод оценки и анализа программ.

Метод оценки и анализа программ (PERT) – используется для очень масштабных и сложных проектов. Он был разработан для упрощения планирования на бумаге и составления графиков больших и сложных проектов. Метод нацелен на анализ времени выполнения для каждой отдельной задачи и для определения минимального времени для выполнения проекта полностью.

Диаграмма представляет собой некоторое множество точек-вершин с соединяющими их ориентированными дугами. Любой дуге, которая рассматривается, как работа прописывают две характеристики это объём выделенных на данную работу ресурсов и длительность. Каждая вершина интерпретируется как событие окончания работ дугам которые входят в неё и одновременно начала работ из исходящих дуг. Таким образом следует, что ни одна работа не начнётся пока не будут выполнены все предшествующие ей. В каждой диаграмме должно быть начало — это вершина без входящих дуг и окончание – это вершина без исходящих дуг. У всех остальных должны быть и те и другие дуги.

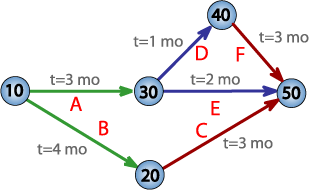


Рис.11. Пример диаграммы PERT. Продолжительностью 7 месяцев. С 5 промежуточными точками.

2.2.2 Метод имитационного моделирования.

Метод имитационного моделирования используется, когда известны статистические параметры модели. Тогда выполняется многократный прогон модели и идёт подсчёт основных характеристик модели. Благодаря большому количеству испытаний мы можем более точно определить закономерности моделируемой сети. Но имитационное моделирование не может предоставить решение в том виде как это представлено в аналитических моделях. Они служат для анализа системы в условиях, которые ей задал экспериментатор. Данный вид моделирования является одним из методов решения поставленной задачи и для его выбора необходима определённых обоснований. Например:

* Когда возникает надобность получить информацию о системе, которую невозможно получить в известных источниках
* Когда происходит экспериментирование на реальной системе если необходимо обеспечить соответствие между моделью и реальными условиями.

Критерии, которым должна удовлетворять хорошая модель:

* простота и понятность;
* надежность;
* удобна в управлении и обращении;
* адаптивность.

Этапы использования имитационного моделирования:

1. Сначала выделяют основные черты модели.
2. Создают и модифицируют имитационную модель.
3. Эксплуатируют и выводят результаты модели.

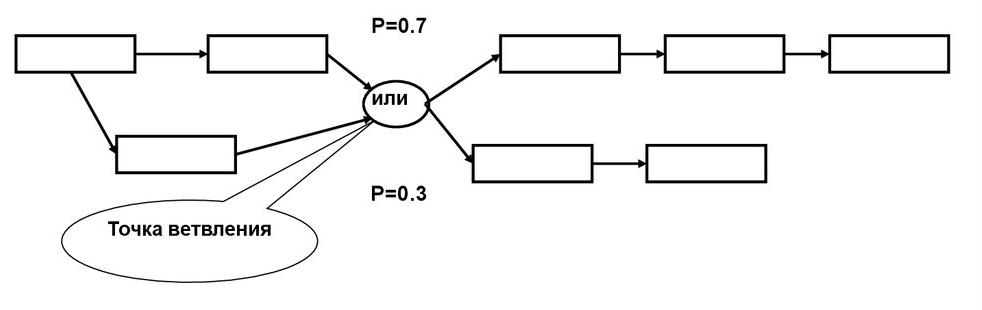
2.2.3 Метод графической оценки и анализа программ**.**

Метод оценки и анализа программ (PERT) используется тогда, когда последующие задачи могут начинаться после выполнения некоторого числа предшествующих задач, причем не все работы на сетевой модели должны быть завершены для завершения проекта. Данный способ помогает упростить сложные процессы производства, когда сложно или невозможно определить какие именно работы и в какой последовательности должны быть выполнены для достижения результата.

В GERT каждое действие описано стрелкой. Для соединения действий, определение типа и условий отношений используют узлы. Каждая задача имеет два параметра: продолжительность и вероятность появления.

Так же в GERT применяются три логических оператора которые касаются действий, идущих на узел:

* XOR - альтернатива (есть только один путь)
* OR - альтернатива (может быть выполнен один или несколько путей)
* И - должны быть выполнены все пути

Рис.12.Пример GERT. С логическим оператором OR

Выводы.

1. Неопределённость — это отсутствие или недостаток информации о чём-либо. Факторами, влияющими на неопределённость являются: неполнота данных, фактор случайности,

2. В сетевом планировании существует несколько типов неопределённости: статистическая неопределённость и стохастическая неопределённость. Все виды неопределенностей необходимо по возможности компенсировать.

3. В сетевом планировании применяют два типа методов: детерминированные и вероятностные. Детерминированные, это те модели в которых продолжительность и последовательность работ задана однозначно. Вероятностные модели, это такие модели, где продолжительность работ точна неизвестна или могут возникнуть непредвиденные ситуации

4. Наиболее часто применяются три распространенных метода вероятностного сетевого планирования:

* метод оценки и анализа программ (PERT);
* метод имитационного моделирования или метод Монте-Карло;
* метод графической оценки и анализа программ (GERT).

**3. Программная реализация сетевого графа**

3.1. Описание модели

Не всегда так происходит, что время выполнения каждой работы точно известно и может принимать лишь одно из возможных значений. Иначе говоря, продолжительность работ *t*(*i,j*) является случайной величиной, определённая своим законом распределения, числовыми характеристиками, своим средним или математическим ожиданием и дисперсией .

Предположим, что в нашем проекте используется на выбор пользователя:

* равномерное распределение;
* наихудший вариант.

Равномерное распределение применяют тогда, когда среди принимаемых случайных величинами значений нет предпочтительных. Обычно считают, что события с одинаковой вероятностью могут происходить в любой момент времени.

Определение: Непрерывная случайная величина X имеет равномерное распределение на интервале [a,b] если на интервале плотность распределения случайной величины постоянна, а за его пределами равна нулю[16]

(1)

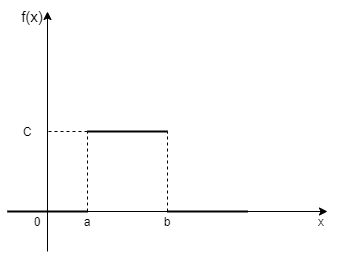


Рисунок 13. - График плотности f(x) равномерного распределения

Функция распределения *F*(*x*) для равномерного распределения имеет вид[16]

(2)

График функции  F(x) представлен на рисунке 14.

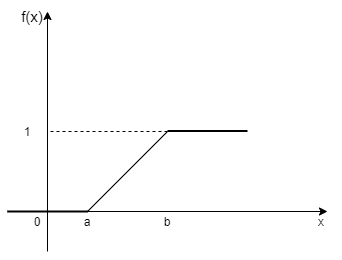


Рис.14 График функции  F(x) равномерного распределения

Вероятность попадания случайной величины X имеющие равномерное распределение на отрезке , являющегося частью участка [a,b] находится по формуле[16]

. (3)

Если же мы используем метод гарантированного результата или наихудший вариант, то этап имитационного моделирования пропускается и будем считать, что все работы будут выполнятся максимально допустимое время. Данный метод может быть полезен, чтобы оценить максимальное время требующиеся на выполнения каждой отдельной работы и всего проекта в целом.

Когда мы привели модель к детерминированному виду нужно убедиться, что все работы будут выполнены в срок. Для этого делают оценку вероятности успеть закончить проект в срок[1]:

 (4)

На рисунке\_\_ показан результат применения данной оценки (в заштрихованной области)

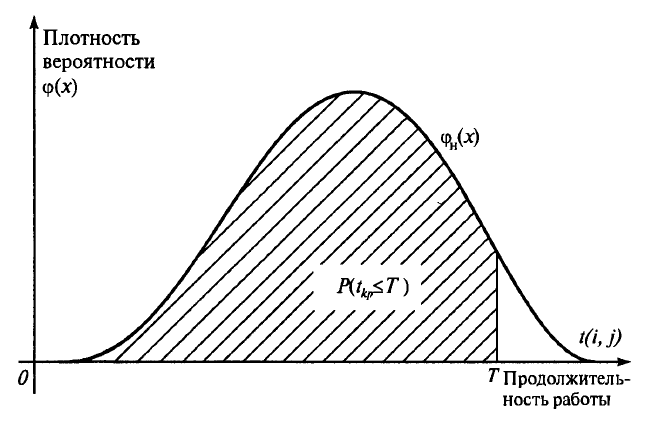


Рис.15 результат применения P(tкр≤T)

где: tкр –длина критического пути;

Ф – значения интеграла вероятности Лапласа;

*Т* - Максимальный срок выполнения проекта;

σкр - среднеквадратичное отклонение длины критического пути.

Если в задаче нам неизвестно *Т*, то его можно найти по формуле[1]:

 (5)

где: - нормальное отклонение случайной величины, определяемая с помощью функции Лапласа.

Если нам неизвестно σ2кр, то можно его найти по формуле[1]:

 (6)

 (7)

 (8)

Аналогично находится по формуле и σ2.

Если *P*(*t*кр≤*T*) ≤ 0,3 то можно сказать c высокой вероятностью, что работа не будет закончена в срок и нужно прибегать к перераспределению ресурсов, пересмотр состава работ, оптимизации сетевого графа.

Если 0,3 < *P*(*t*кр≤*T*) < 0,8 то можно сказать о средней вероятности успеха закончить работы в срок.

Если *P*(tкр≤*T*) ≥ 0,8 то достаточно очевидно, что с высокой вероятностью работа будет закончена в срок.

В проекте используются основные статистические точечные оценки:

* оценка генеральной средней продолжительности работы[27]

(9)

где =элемент выборки, ; n – объём выборки;

* выборочная дисперсия[27]

(10)

* несмещённая оценка дисперсии[27]

(11)

* среднеквадратичное отклонение[27]

(12)

* смещённая оценка СКО[27]

(13)

Так же в проекте построен доверительный интервал с выбранным уровнем статистической значимости. Так как у нас закон распределения, по которому мы моделируем значения не нормальный, то используем *t* - распределение. Данная методика имеет более широкий доверительный интервал по сравнению с методикой, базирующейся на *Z*-оценке.  
 Для расчета нижнего и верхнего предела доверительного интервала на основании *t*-распределения применяют следующие две формулы[27]:

 (14)

 (15)

где: – оценка генеральной средней продолжительности работы;

*α* – уровень статистической значимости;

*n* – объём выборки;

– Критерий Стьюдента для уровня значимости и количества степеней свободы (*n*-1);

– Среднеквадратичное отклонение.

3.2. Программная реализация

Вышеописанная модель реализована с использованием пакета прикладных программ для решения технических задач и вычислений – MATLAB.

Входными данными программы является: заполненный документ формата .csv. В документе заполняются первые четыре столбца:

* А – номер работы (number\_work);
* B – опорная работа (support\_work);
* C – ранний срок окончания работы (er\_time);
* D – поздний срок выполнения работы (lr\_time), если работа точно определена, то ставится “0”.

Выходными данными являются: оптимальное время выполнения для каждой отдельной работы и всей работы в целом, оценка вероятности выполнения всей работы в срок. Так же программа выдает упорядоченные данные из документа, выдает количество дней для выполнения всей работы и возможные простои на некритических путях. Для визуализации данных строится сетевой граф с выделенным критическим путём полученных из документа.

Блок-схема программы показана на рисунке 16

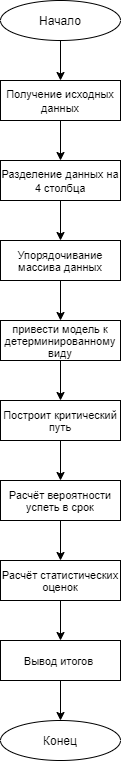


Рис.16 Блок-схема программы

Схема системы по упорядочиванию массива данных, приведение к детерминированному виду, построению критического пути, расчёт вероятности, расчёт статистических оценок и вывод итогов. Некоторые блоки более детально расписаны на последующих блок-схемах:

Блок-схема сортировки данных массива по первому столбцу представлена на Рис.17

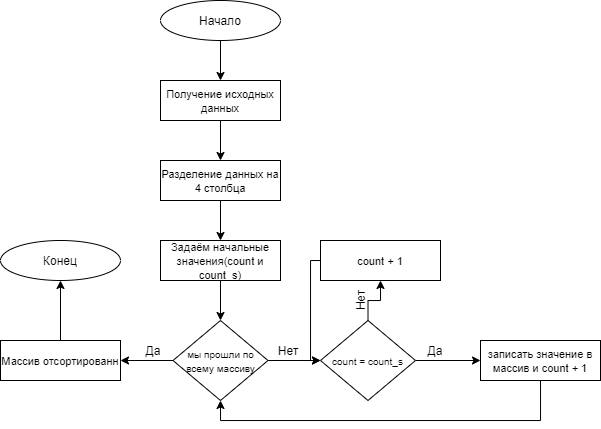


Рис.17 Блок-схема сортировки данных.

В блок-схеме используются следующие обозначения:

* Count – счётчик для перемещения по массиву данных
* Count\_s – счётчик для поиска нужных элементов в массиве

Блок-схема привидения данных к детерминированному виду представлена на Рис.18

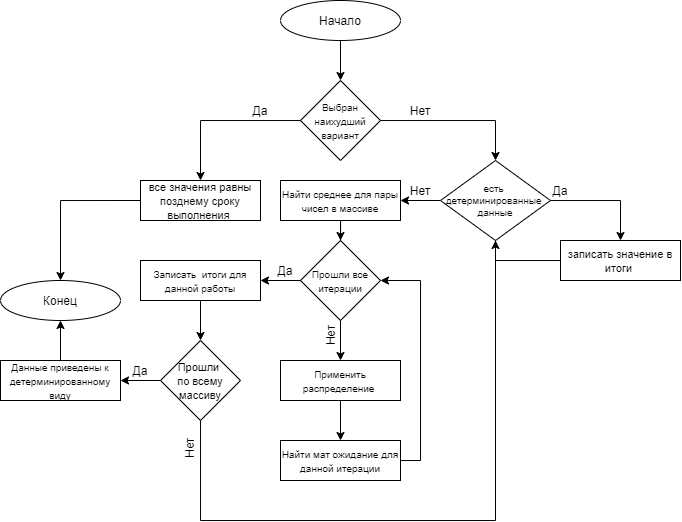


Рис.18Блок-схема привидения данных к детерминированному виду.

Блок-схема нахождения развилок для графа представлена на Рис.19

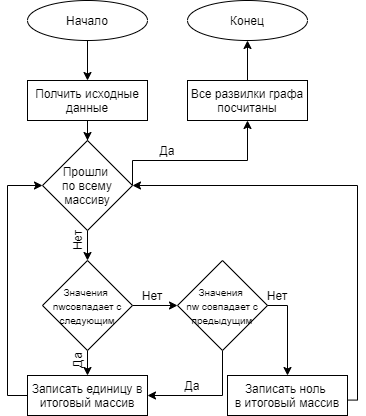


Рис.19 Блок-схема нахождения развилок для графа (используется в модуле нахождения критического пути)

Блок-схема нахождения критического пути в графе представлена на Рис.20

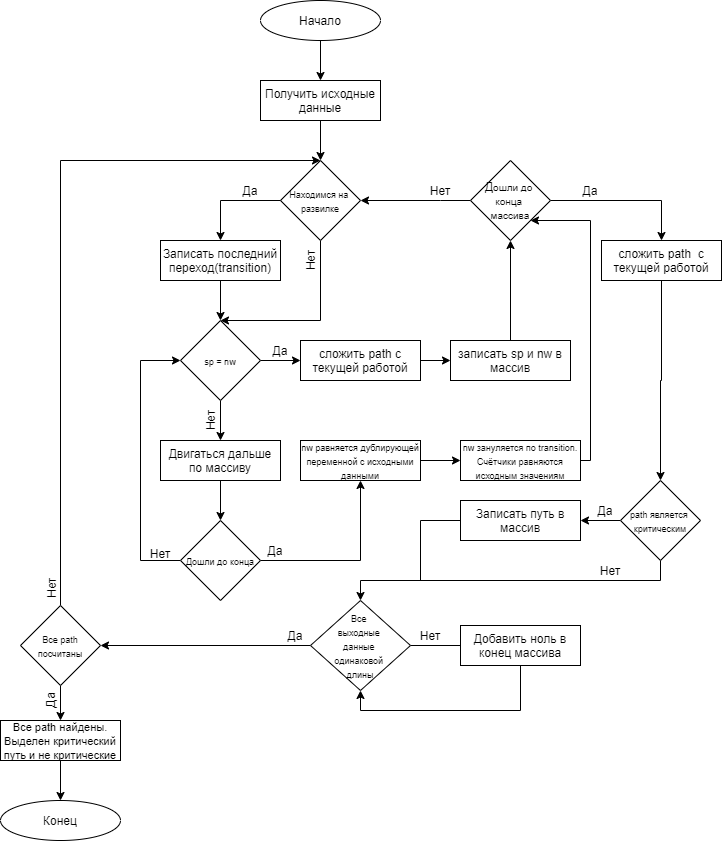


Рис.20 Блок-схема нахождения критического пути в графе

В блок-схеме используются следующие обозначения:

* sp → support\_work – данные из второго столбика документа .csv
* nw →number\_work – данные из первого столбика документа .csv
* transition – равняется значению счётчику, когда делаем переход по развилке
* path – значения по которому мы суммируем работы идя по пути

Блок-схема расчёта вероятности успеть вовремя закончить все работы представлена на Рис.21

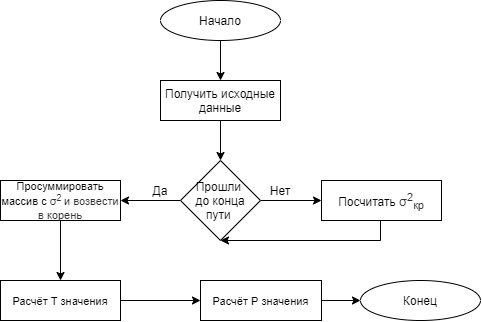


Рис.21 Блок-схема расчёта вероятности

3.3. Вычислительный эксперимент

В качестве примера для вычислительного эксперимента были взяты данные с официального сайта администрации города Тверь. Документ, по которому выполнен вычислительный эксперимент называется: “Постановление Администрации Города Твери от 30.03.2021 №371\_371\_30\_03\_2021”. В данном документе описывается план по санитарной обработке и благоустройству территории города Твери в 2021 году. В перечень входят:

* санитарная очистка;
* уборка объектов потребительского рынка и примыкающих к ним территориям;
* ремонт малых архитектурных форм;
* ликвидация навалов мусора;
* ремонт контейнерных площадок;
* уборка аварийных деревьев, сучьев и поросли;
* посадка деревьев и кустарников.

У данного плана есть срок выполнения всей работы с 30 марта 2021 по 28 мая 2021. Так и у каждой отдельной работы есть определённый промежуток выполнения работ. Список всех работ и их промежуток исполнения представлен на таблице 1.

Таблица 1. – Исходные данные для моделирования

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 1 | 6 |
| 2 | 3 | 10 | 0 |
| 3 | 4 | 7 | 0 |
| 4 | 5 | 7 | 0 |
| 5 | 6 | 21 | 0 |
| 6 | 12 | 6 | 0 |
| 2 | 7 | 18 | 34 |
| 7 | 12 | 11 | 17 |
| 2 | 8 | 14 | 21 |
| 8 | 9 | 15 | 21 |
| 9 | 12 | 7 | 10 |
| 2 | 12 | 28 | 51 |
| 2 | 10 | 14 | 30 |
| 10 | 12 | 14 | 20 |
| 2 | 11 | 7 | 14 |
| 11 | 12 | 7 | 15 |
| 12 | 13 | 6 | 12 |

Данная таблица переносится в файл с расширением .csv. Файлу присваивается имя и переносится в корневой каталог с проектом.

Код программы представлен в приложении 1.

Пользовательский интерфейс представлен на Рис.22

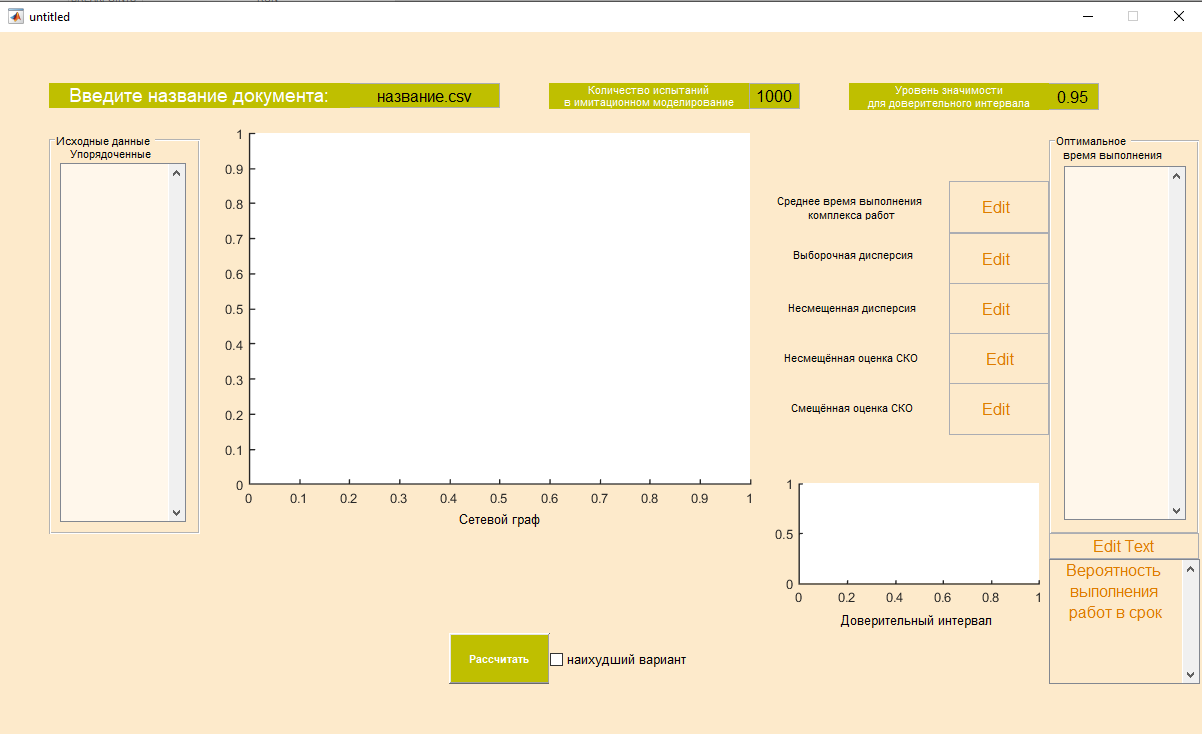


Рис.22 Пользовательский интерфейс

При первом запуске пользователю нужно ввести название документа с расширением .csv. При необходимости можно выбрать количество испытаний и уровень значимости для доверительного интервала. Как альтернатива Имитационному моделированию можно выбрать “наихудший вариант”. Когда мы ввели все необходимые входные переменные нажимаем кнопку “Рассчитать”. Результат продемонстрирован на Рис.23

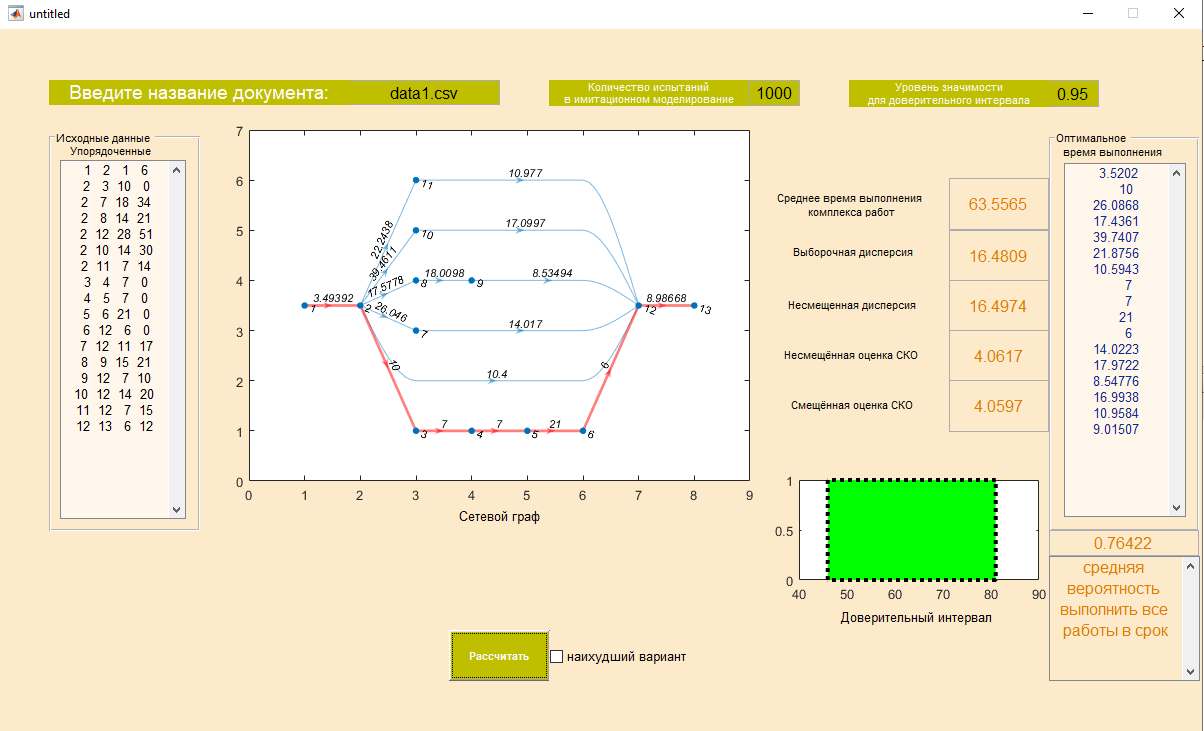


Рис.23 Пользовательский интерфейс с выходными данными

На рисунке 23 показан результат работы программы.

Выходные данные. Упорядоченные данные по первому столбцу (левая колонка), построенный сетевой граф с выделенным критическим путём. Посчитанные статистические оценки такие как: среднее время выполнения всего комплекса работ, выборочная дисперсия, несмещённая дисперсия, несмещённая оценка СКО, смещённая оценка СКО и построение доверительного интервала. Рассчитанное оптимальное время выполнение для каждой работы (правый колонка), вероятность успеть вовремя (значение под правой колонкой), сделан вывод по полученной вероятности.

Если мы хотим пересчитать по наихудшему варианту, то нужно нажать на соответствующею галочку и нажать кнопку “Рассчитать”. Данный пример показан на Рис.24

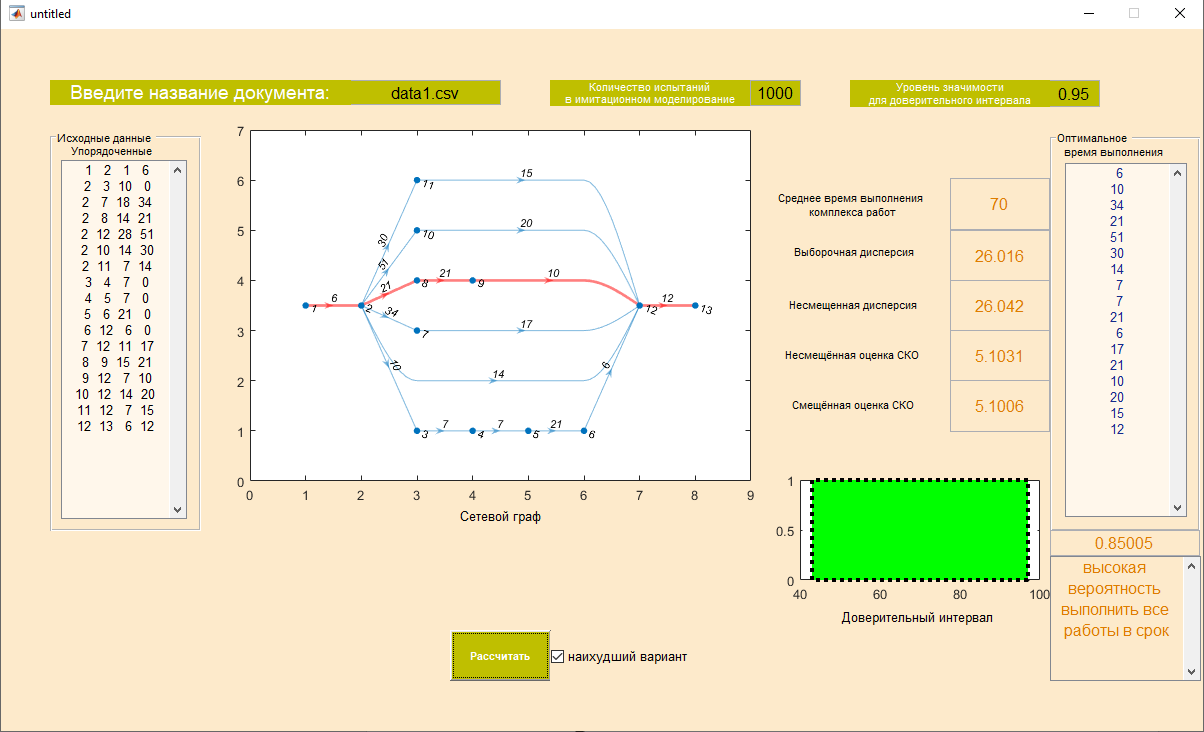


Рис.\_\_ Пользовательский интерфейс с выходными данными по наихудшему варианту

На рисунке 24 показан результат работы программы по наихудшему варианту. Выходные данные те же, когда мы используем метод имитационного моделирования.

Выводы.

1. Программная реализация сетевого графа предусматривает: Описание, программную реализацию модели и проведение вычислительного эксперимента.

2. В описании модели предусмотрен выбор варианта распределения для моделирования процесса – вариант равномерного распределения времени выполнения работ.

3. Для нахождения вероятности того, что все работы будут выполнены в срок, использовано выражение интеграла вероятностей Лапласа (4) .

4. При выполнении вычислений получены численные статистические оценки генеральной средней продолжительности работы, дисперсии, среднеквадратического отклонения и смещённой оценка СКО.

5. Вычислены значения верхней и нижней границ доверительного интервала, в пределы которого с доверительной вероятностью 0,95 будет попадать вычисленное значение времени выполнения всего комплекса работ.

6. Разработана компьютерная программа с использованием пакета прикладных программ Матлаб, приведенная в приложении №1.

**Заключение**

Целью работы «Сетевое планирование в условиях неопределенности» явилось исследование вопросов сетевого планирования в условиях неопределённости.

Актуальность работы определяется следующими обстоятельствами:

- для достижения высоких результатов деятельности практически во всех сферах применяется сетевое планирование;

- для качественного выполнение сетевого планирования в реальной жизни известны далеко не все факторы, имеет место неопределенность.

В интересах решения поставленных задач выполнено:

- исследование вопросов содержания и параметров сетевого планирования;

- Проанализированы три основные метода сетевого планирования и вскрыто влияние факторов неопределенности на процесс;

- разработана компьютерная программа сетевого планирования с использованием пакета прикладных программ Матлаб;

- выполнено моделирование сетевого планирования благоустройства города Твери в 2021 году в условиях неопределённости, проведен анализ результатов моделирования.

Программа позволяет:

- получать оптимальное время выполнения каждой отдельной работы и среднее время всего комплекса работ;

- вероятность закончить все работы в установленный срок;

- основные статистические оценки: генеральной средней продолжительности работы, выборочной дисперсии, несмещённой оценки дисперсии, среднеквадратического отклонения, смещённой оценки СКО.

Для визуализации моделируемых данных в программе реализован сетевой граф с выделением на нём критического пути.

Программа гибкая, может работать с различными исходными данными, определять статистически достаточное число испытаний в модели, определять уровень значимости.

Для сравнения варианта разработанной модели с «Наихудшим вариантом» разработан и запрограммирован альтернативный вариант модели.

Таким образом, цель работы достигнута полностью, поставленные задачи решены. Разработанную программу можно применять в различных областях деятельности людей: строительство объектов, освоение нового производства продукции в промышленных/торговых компаниях, реструктуризация производства.

**Список литературы**

1. Н. И. Новицкий, Сетевое планирование и управление производством. Москва, ООО “Новое знание” 2004–159 с.
2. Багрова И.А., Шахпаронян А.П. Имитационное моделирование. – М.: Тверь,2014. – 160 с.
3. Кремер Н. Ш. и др. Исследование операций в экономике. – М.: ЮНИТИ,2008. – 407 с.
4. Г.Л. Бродецкий, Системный анализ в логистике. Выбор в условиях неопределенности. М.: Academia, 2010–336 стр.
5. Экономико-математические методы и прикладные модели: Учеб. Пособие для вузов/ В.В. Федосеев, А.Н. Гармаш, Д.М. Дайитбегов и др.; Под ред В.В. Федосеева. – М.: ЮНИТИ, 2001. – 391 с.
6. Дюбуа Д., Прад А. Теория возможностей. Приложения к представлению знаний в информатике. Пер. с фр. – М.: Радио и связь, 1990. – 288 с.
7. Schruben L.W. Simulation Modeling with Event Graphs. Communications of the ACM. 26: 957-963.
8. Подготовка графиков в MATLAB/Habr. Электрон. дан. – [Б.м., 2013]. – Режим доступа: <https://habr.com/ru/post/195842/> – Загл. с экрана.
9. Построение графиков в MATLAB/Mathworks. Электрон. дан. – [Б.м., 2021]. – Режим доступа: [https://www.mathworks.com/help/matlab/ref/plot.html.](https://www.mathworks.com/help/matlab/ref/plot.html) – Загл. с экрана.
10. Сборник полезных средств для настройки графики в MATLAB/Habr. . Электрон.дан.– [Б.м., 2013]. – Режим доступа: [https://habr.com/ru/post/174697](https://habr.com/ru/post/174697/)– Загл. с экрана.
11. Построение графов в MATLAB/Mathworks.Электрон.дан. [Б.м.,2021].–Режим\_доступа: <https://www.mathworks.com/help/matlab/ref/graph.html>– Загл. с экрана.
12. Объявление функции её имя входные и выходные данные в MATLAB/Mathworks.Электрон.дан.[Б.м.,2021].–Режим\_доступа: <https://www.mathworks.com/help/matlab/ref/function.html> – Загл. с экрана.
13. Создание двумерного массива в MATLAB/CodeLessons.com Электрон.дан.[Б.м.,2018].–Режим\_доступа: <https://coderlessons.com/tutorials/kompiuternoe-programmirovanie/uznaite-matlab/matlab-massivy> –Загл. с экрана.
14. Создание многомерного массива в MATLAB/Mathworks. Электрон.дан.[Б.м.,2021].–Режим\_доступа: <https://www.mathworks.com/help/matlab/math/multidimensional-arrays.html> –Загл. с экрана.
15. Выделение критического пути (highlight) в MATLAB/Mathworks. Электрон.дан.[Б.м.,2021].–Режим\_доступа: <https://www.mathworks.com/help/matlab/ref/matlab.graphics.chart.primitive.graphplot.highlight.html> – Загл. с экрана.
16. Непрерывное равномерное распределение/Wikipedia. Электрон.дан.[Б.м.,2021].–Режим\_доступа:

<https://ru.wikipedia.org/wiki/Непрерывное_равномерное_распределение#:~:text=Непреры́вное%20равноме́рное%20распределе́ние%20в%20теории,этом%20промежутке%20почти%20всюду%20постоянна>

1. Изменение макета графа в MATLAB/Mathworks.Электрон.дан.[Б.м.,2021].–Режим\_доступа: <https://www.mathworks.com/help/matlab/ref/matlab.graphics.chart.primitive.graphplot.layout.html> – Загл. с экрана.
2. Поиск индексов и значений не нулевых элементов в MATLAB/Mathworks.Электрон.дан.[Б.м.,2021].–Режим\_доступа: <https://www.mathworks.com/help/matlab/ref/find.html> – Загл. с экрана.
3. А.Ю.Михайлов, Организация строительства. Календарное и сетевое планирование Москва; Вологда: Инфа-Инженерия, 2020. –300с.
4. Нахождение длины массива в MATLAB/Mathworks.Электрон.дан.[Б.м.,2021].–Режим\_доступа: <https://www.mathworks.com/help/matlab/ref/length.html> – Загл. с экрана.
5. Загрузка данных в MATLAB/Mathworks.Электрон.дан.[Б.м.,2021].–Режим\_доступа: <https://www.mathworks.com/help/matlab/ref/importdata.html> – Загл. с экрана.
6. Создание пользовательского приложения с использованием MATLAB/Mathworks.Электрон.дан.[Б.м.,2021].–Режим\_доступа: <https://www.mathworks.com/help/matlab/creating_guis/about-the-simple-guide-gui-example.html> – Загл. с экрана.
7. Переход в GUIDE “Edit Text” от однострочного отображения к многострочному отображению данных Stackoverflow/ Stackoverflow.Электрон.дан.[Б.м.,2014].–Режим\_доступа: <https://stackoverflow.com/questions/22287726/single-and-multi-line-edit-controls> – Загл. с экрана.
8. Е. А. Бабкин, И. Ю. Пикалов, Имитационное моделирование сетевых моделей. Статья Курский государственный университет 2019. –8с.
9. Изучение PERT диаграмм Habr/Habr.Электрон.дан.[Б.м.,2016].–Режим\_доступа: <https://habr.com/ru/post/318738/> – Загл. с экрана.
10. Нахождение функции ошибки. [Б.м.,2021].–Режим \_доступа: <https://www.mathworks.com/help/matlab/ref/erf.html#d123e362543>– Загл. с экрана.
11. Е.С. Вентцель. Теория вероятностей. Издательство «Наука», М. , 1969 – 564 с.
12. Сетевое планирование.Википедия.Электрон.дан.[Б.м.,2021].–Режим\_доступа: <https://ru.wikipedia.org/wiki/Сетевое_планирование>. – Загл. с экрана.

**ПРИЛОЖЕНИЕ**

**Код программы**

**function data\_sort** = d\_sort(var)

m = importdata(var);

number\_work = m(:,1);

support\_work = m(:,2);

er\_time = m(:,3);

lr\_time = m(:,4);

count = 1;

count\_s = 1;

count\_len = 1;

len = length(number\_work);

while true

while len >= count

if (number\_work(count) == count\_s)

e = [number\_work(count),support\_work(count),er\_time(count),lr\_time(count)];

data\_sort(count\_len,:) = [e];

count\_len = count\_len + 1;

number\_work(count) = 0;

else

count = count + 1;

end

end

count\_s = count\_s + 1;

count = 1;

if count\_len > len

break;

end

end

end

**function time\_avg** = choose\_path(var,var1,var2)

if(var1 == 1)

time\_avg = worst\_case(var);

else

time\_avg = wghts(var,var2);

end

end

**function rasp** = IM(var,var2)

data\_sort = d\_sort(var);

er\_time = data\_sort(:,3);

**Продолжение приложения**

lr\_time = data\_sort(:,4);

len = length(er\_time);

for i =1:var2

for n=1:len

if (lr\_time(n)==0)

rasp(n,i)=er\_time(n);

continue;

else

rasp(n,i) = unifrnd(er\_time(n),lr\_time(n));

end

end

end

end

**function t\_avg** = worst\_case(var)

datat\_sort = d\_sort(var);

er\_time = datat\_sort(:,3);

lr\_time = datat\_sort(:,4);

len = length(er\_time);

count = 1;

count\_t = 1;

while len >= count

if (lr\_time(count)==0)

t = er\_time(count);

else

t = lr\_time(count);

end

count = count + 1;

t\_avg(count\_t,:)=[t];

count\_t = count\_t + 1;

end

end

**function critical\_path** = critical\_path(var,var1,var2)

data\_sort = d\_sort(var);

weights = choose\_path(var,var1,var2);

number\_work = data\_sort(:,1);

support\_work = data\_sort(:,2);

len = length(number\_work);

i = 1;

c = 0;

brk =0;

tmp =0;

path\_w = 0;

count =1;

count\_s =1;

count\_len = 1;

critial\_tmp = 0;

for n =1:len

**Продолжение приложения**

if(number\_work(n) == 1)

i = i + 1;

end

end

number\_wn = number\_work;

fork = search\_fork(var);

path\_quantity =zeros(sum(fork)+1,1);

while true

if support\_work(count) == max(support\_work)

path\_w = path\_w + weights(count);

e = [number\_work(count),support\_work(count)];

c = c + 1;

data\_tmp(c,:) = [e];

if (path\_w > critial\_tmp)

data\_critical = data\_tmp;

critial\_tmp = path\_w;

end;

critical(count\_len,:)=[path\_w];

count\_len = count\_len + 1;

number\_work(transition) = 0;

if tmp ==0

count = 1;

else

count = i;

end

count\_s = 1;

path\_w = 0;

brk = brk + 1;

c = 0;

x = [data\_tmp];

x = zeros\_data(x,fork);

if(length(x) > length(path\_quantity))

path\_quantity = zeros\_data(path\_quantity,fork);

end

path\_quantity = [path\_quantity,x];

data\_tmp = [];

if (brk == sum(fork))

data\_critical = zeros\_data(data\_critical,fork);

critical = zeros\_critical(critical,fork);

critical\_path = [critical,data\_critical];

break;

end

end

**Продолжение приложения**

if (fork(count) ==0)

while true

if (support\_work(count) == number\_work(count\_s))

path\_w = path\_w + weights(count);

e = [number\_work(count),support\_work(count)];

c = c + 1;

data\_tmp(c,:) = [e];

count = count\_s;

break;

else

count\_s = count\_s + 1;

if (count\_s > len)

number\_wn(transition)=0;

number\_work = number\_wn;

number\_work(transition)=0;

i = i - 1;

count = i;

c = 0;

data\_tmp = [];

tmp =1;

count\_s =1;

path\_w = 0;

break;

end

end

end

else

while true

transition =count;

if (support\_work(count) == number\_work(count\_s))

path\_w = path\_w + weights(count);

e = [number\_work(count),support\_work(count)];

c = c + 1;

data\_tmp(c,:) = [e];

count = count\_s;

break;

else

count\_s = count\_s + 1;

if (count\_s > len)

number\_wn(transition)=0;

number\_work = number\_wn;

count = i;

i = i - 1;

count\_s =1;

path\_w = 0;

c = 0;

data\_tmp = [];

break;

end

end

end

**Продолжение приложения**

end

end

end

**function fork** = search\_fork(var)

data\_sort = d\_sort(var);

number\_work = data\_sort(:,1);

count = 1;

count\_len = 1;

len = length(number\_work);

while true

if (number\_work(count) == number\_work(count + 1))

n = 1;

fork(count\_len,:)=[n];

else

n = 0;

fork(count\_len,:)=[n];

end

count = count + 1;

count\_len = count\_len + 1;

if(count +1 > len)

n = 1;

fork(count\_len,:)=[n];

break;

end

end

end

**function s** = fnd(m,x)

s = m(:,x);

s(find(s==0))=[];

end

**function critical** = zeros\_critical(critical,fork)

count\_d = length(critical);

while true

if (count\_d < sum(fork) + 1)

critical(count\_d + 1,:) =[0];

count\_d = count\_d + 1;

else

break

end

end

end

**Продолжение приложения**

**function data\_critical** = zeros\_data(data\_critical,fork)

count\_d = length(data\_critical);

while true

if (count\_d < sum(fork) + 1)

data\_critical(count\_d + 1,:) =[0,0];

count\_d = count\_d + 1;

else

break

end

end

end

**function cp** =main(var,var1,var2);

weights = choose\_path(var,var1,var2);

data\_sort=d\_sort(var);

number\_work = data\_sort(:,1);

support\_work = data\_sort(:,2);

m = critical\_path(var,var1,var2);

path\_1 = m(:,2);

path\_2 = m(:,3);

path=[path\_1,path\_2];

path(find(path==0))=[];

G = digraph(number\_work,support\_work);

e = plot(G,'EdgeLabel',weights,'Layout','layered','Direction','right');

highlight(e,path,'EdgeColor','r','LineWidth',2);

cp = max(m(:,1));

end

**function P** = probability\_success(var,var1,var2)

im = wghts(var,var2);

worst = worst\_case(var);

count = 1;

count\_s = 1;

m =critical\_path(var,var1,var2);

t\_cp = max(m(:,1));

path\_1 = fnd(m,2);

path\_2 = fnd(m,3);

path=[path\_1+path\_2];

z =1.96;

data\_sort = d\_sort(var);

ds1 = data\_sort(:,1);

ds2 = data\_sort(:,2);

ds = [ds1+ds2];

while true

if (path(count) == ds(count\_s))

sigma\_sq\_cp(count) = ((worst(count\_s) - im(count\_s))/6)^2;

count = count + 1;

else

**Продолжение приложения**

count\_s = count\_s + 1;

end

if count\_s >= length(ds)

sigma\_sq\_cp = sum(sigma\_sq\_cp);

break;

end

end

T = t\_cp +z\*sigma\_sq\_cp;

x = (T -t\_cp)/(sqrt(sigma\_sq\_cp));

P =0.5 + 0.5\*(erf(x)/sqrt(2));

end

**function varargout** = untitled(varargin)

gui\_Singleton = 1;

gui\_State = struct('gui\_Name', mfilename, ...

'gui\_Singleton', gui\_Singleton, ...

'gui\_OpeningFcn', @untitled\_OpeningFcn, ...

'gui\_OutputFcn', @untitled\_OutputFcn, ...

'gui\_LayoutFcn', [] , ...

'gui\_Callback', []);

if nargin && ischar(varargin{1})

gui\_State.gui\_Callback = str2func(varargin{1});

end

if nargout

[varargout{1:nargout}] = gui\_mainfcn(gui\_State, varargin{:});

else

gui\_mainfcn(gui\_State, varargin{:});

end

**function untitled\_OpeningFcn**(hObject, eventdata, handles, varargin)

handles.output = hObject;

guidata(hObject, handles);

**function varargout** = untitled\_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;

**function pushbutton1\_Callback**(hObject, eventdata, handles)

var = get(handles.edit1,'string');

var1 = get(handles.checkbox1,'Value');

var2 = get(handles.edit9,'string');

var2 =str2double(var2);

var3 = get(handles.edit20,'string');

var3 =str2double(var3);

a = main(var,var1,var2);

b = d\_sort(var);

c = choose\_path(var,var1,var2);

d = probability\_success(var,var1,var2);

e = critical\_path(var,var1,var2);

**Продолжение приложения**

e = e(:,1);

e = fnd(e,1);

len =length(e);

disp\_select = sum(e.^2)/var2;

n =var2/(var2-1);

disp\_not\_select =n\*disp\_select;

CKO\_not\_select = sqrt(n \* disp\_select);

CKO\_select = sqrt(disp\_select);

switch var3

case 0.90

var3 = 0.95;

case 0.95

var3 = 0.975;

case 0.99

var3 = 0.995;

otherwise

var3 = 0.975;

end

t\_alpha = tinv(var3,len-1);

interval\_l = round(a - t\_alpha\*(disp\_not\_select /sqrt(len)));

interval\_r = round(a + t\_alpha\*(disp\_not\_select /sqrt(len)));

x= [interval\_l,interval\_l,interval\_r,interval\_r];

y=[0,1,1,0];

area(handles.axes2,x,y,'FaceColor','g','LineStyle',':','LineWidth',3);

set(handles.edit15,'String',num2str(a));

set(handles.edit16,'String',num2str(disp\_select));

set(handles.edit17,'String',num2str(disp\_not\_select));

set(handles.edit18,'String',num2str(CKO\_not\_select));

set(handles.edit19,'String',num2str(CKO\_select));

set(handles.edit3,'String',num2str(b));

set(handles.edit4,'String',num2str(c));

set(handles.edit6,'String',num2str(d));

if(d >= 0.8)

set(handles.edit7,'String','высокая вероятность выполнить все работы в срок');

elseif (d>= 0.6)

set(handles.edit7,'String','средняя вероятность выполнить все работы в срок');

elseif (d > 0.3)

set(handles.edit7,'String','низкая вероятность выполнить все работы в срок');

else

set(handles.edit7,'String','Вероятность выполнения работы в срок крайне мала. Рекомендуется повторить расчёт');

end

**function edit1\_Callback**(hObject, eventdata, handles)

**Продолжение приложения**

**function edit1\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function edit1\_KeyPressFcn**(hObject, eventdata, handles)

**function edit4\_Callback**(hObject, eventdata, handles)

**function edit4\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function edit6\_Callback**(hObject, eventdata, handles)

**function edit6\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function edit2\_Callback**(hObject, eventdata, handles)

**function edit2\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function edit3\_Callback**(hObject, eventdata, handles)

**function edit3\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function checkbox1\_Callback**(hObject, eventdata, handles)

**function edit7\_Callback**(hObject, eventdata, handles)

**function edit7\_CreateFcn**(hObject, eventdata, handles)

**Продолжение приложения**

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function checkbox4\_Callback**(hObject, eventdata, handles)

**function radiobutton1\_Callback**(hObject, eventdata, handles)

**function edit9\_Callback**(hObject, eventdata, handles)

**function edit9\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function edit15\_Callback**(hObject, eventdata, handles)

**function edit15\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function edit16\_Callback**(hObject, eventdata, handles)

**function edit16\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function edit17\_Callback**(hObject, eventdata, handles)

**function edit17\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function** **edit18\_Callback**(hObject, eventdata, handles)

**Продолжение приложения**

**function** **edit18\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function** **edit19\_Callback**(hObject, eventdata, handles)

**function** **edit19\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function** **edit20\_Callback**(hObject, eventdata, handles)

**function** **edit20\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function** **edit21\_Callback**(hObject, eventdata, handles)

**function** **edit21\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

**function** **edit23\_Callback**(hObject, eventdata, handles)

**function** **edit23\_CreateFcn**(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end