Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки 01.04.02 Прикладная математика и информатика
Отделение школы (НОЦ) Отделение информационных технологий

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Тема работы
Разработка алгоритма минимизации риска перехода в неблагоприятное состояние организма человека

УДК 004.421:612.014.4-044.57

Студент

Группа	ФИО	Подпись	Дата
8БМ61	Кисатов Марат Александрович		

Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ИШИТР	Гергет Ольга Михайловна	к. т. н.		

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Старший преподаватель ШИП	Шаповалова Наталья Владимировна			

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент ИШНКБ	Авдеева Ирина Ивановна			

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Профессор ИШИТР	Коваль Тамара Васильевна	д. ф.-м. н.		

Томск – 2018 г.

Планируемые результаты обучения по направлению подготовки 01.04.02 Прикладная математика и информатика

Код результата	Результат обучения
1	**2**
Общекультурные компетенции	
ОК-1	Способность к абстрактному мышлению, анализу, синтезу.
ОК-2	Готовность действовать в нестандартных ситуациях, нести социальную и этическую ответственность за принятые решения.
ОК-3	Готовность к саморазвитию, самореализации, использованию творческого потенциала.
Общепрофессиональные компетенции	
ОПК-1	Готовность к коммуникации в устной и письменной формах на государственном языке Российской Федерации и иностранном языке для решения задач профессиональной деятельности.
ОПК-2	Готовность руководить коллективом в сфере своей профессиональной деятельности, толерантно воспринимая социальные, этнические, конфессиональные и культурные различия.
ОПК-3	Способность самостоятельно приобретать с помощью информационных технологий и использовать в практической деятельности новые знания и умения, в том числе, в новых областях знаний, непосредственно не связанных со сферой деятельности, расширять и углублять своё научное мировоззрение.
ОПК-4	Способность использовать и применять углубленные знания в области прикладной математики и информатики.

Продолжение таблицы

1	2
ОПК-5	Способность использовать углублённые знания правовых и этических норм при оценке последствий своей профессиональной деятельности, при разработке и осуществлении социально значимых проектов.
Профессиональные компетенции	
ПК-1	Способность проводить научные исследования и получать новые научные и прикладные результаты самостоятельно и в составе научного коллектива.
ПК-2	Способность разрабатывать концептуальные и теоретические модели решаемых научных проблем и задач.
ПК-3	Способность углубленного анализа проблем, постановки и обоснования задач научной и проектно-технологической деятельности.
ПК-4	Способность разрабатывать концептуальные и теоретические модели решаемых задач проектной и производственно-технологической деятельности.
ПК-5	Способность управлять проектами, планировать научно-исследовательскую деятельность, анализировать риски, управлять командой проекта.
ПК-6	Способность организовывать процессы корпоративного обучения на основе технологий и развития корпоративных баз знаний.
ПК-7	Способность разрабатывать и оптимизировать бизнес-планы научно-прикладных проектов

Окончание таблицы

1	2
ПК-8	Способность разрабатывать корпоративные стандарты и профили функциональной стандартизации приложений, систем, информационной инфраструктуры.
ПК-9	Способность к преподаванию математических дисциплин и информатики в образовательных организациях основного общего, среднего общего, среднего профессионального и высшего образования.
ПК-10	Способность разрабатывать учебно-методические комплексы для электронного и мобильного обучения.
ПК-11	Способность разрабатывать аналитические обзоры состояния области прикладной математики и информационных технологий.
ПК-12	Способность к взаимодействию в рамках международных проектов и сетевых сообществ.
ПК-13	Способность осознавать корпоративную политику в области повышения социальной ответственности бизнеса перед обществом, принимать участие в её развитии.

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Школа ____
Направление подготовки (специальность)_____
Отделение школы (НОЦ)_____

УТВЕРЖДАЮ:
Руководитель ООП
_____ _____ _____
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

Магистерской диссертации

(бакалаврской работы, дипломного проекта/работы, магистерской диссертации)

Студенту:

Группа	**ФИО**
8БМ61	Кисатову Марату Александровичу

Тема работы:

Разработка алгоритма минимизации риска перехода в неблагоприятное состояние организма человека	
Утверждена приказом директора (дата, номер)	

Срок сдачи студентом выполненной работы:	

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к особенностям функционирования (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.).*	Данные из гор. Больницы №4. (г. Томск) о состоянии здоровья 152 человек.

Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе).*	Вычисление интегрального показателя для оценки состояния оргнизма; Провести сравнительный анализ алгоритмов поиска оптимального управления (генетический алгоритм, алгоритм симуляции отжига, полный перебор); С помощью алгоритма поиска оптимального управления подобрать управляющее воздействие, позволяющее минимизировать отклонение функционального состояния от нормы; Оценить риск перехода в неблагоприятное функциональное состояние с учетом выбранного воздействия.
Перечень графического материала *(с точным указанием обязательных чертежей)*	

Консультанты по разделам выпускной квалификационной работы

(с указанием разделов)

Раздел	Консультант
Финансовый менеджмент	Шаповалова Наталья Владимировна
Ресурсоэффективность и ресурсосбережение	Шаповалова Наталья Владимировна
Социальная ответственность	Авдеева Ирина Ивановна
Английский язык	Комиссарова Ольга Валентиновна

Названия разделов, которые должны быть написаны на русском и иностранном языках:

Основные методы при решении задачи

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ИШИТР	Гергет Ольга Михайловна	к.т.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8БМ61	Кисатов Марат Александрович		

РЕФЕРАТ

Выпускная квалификационная работа 159с., 48 рис., 26 табл., 32 источника, 5 прил.

Ключевые слова: риск, показатель, алгоритм, воздействие, оптимум, оценка, состояние.

Объектом исследования является методика минимизации риска перехода в неблагоприятное состояние организма человека.

Цель работы – оценить риск перехода в неблагоприятное состояние организма человека и подобрать управляющее воздействие, чтобы его минимизировать. Исследовать применимость различных алгоритмов (генетический алгоритм, отжиг и полный перебор) к данной задаче.

В процессе исследования проводилось изучение методов поиска оптимального управляющего воздействия для рассматриваемого примера.

В результате исследования проведен сравнительный анализ алгоритмов поиска оптимального решения и выбран лучший из них. Все рассматриваемые алгоритмы реализованы программно в среде визуального программирования на языке высокого уровня.

Основные конструктивные, технологические и эксплуатационные характеристики: высокий современный научный уровень, высокая степень новизны и оригинальности.

Область применения: интеллектуальная поддержка принятия решений на основе медицинских данных.

ОПРЕДЕЛЕНИЯ

В данной работе применены следующие термины с соответствующими определениями:

Оптимальное управление — это задача проектирования системы, обеспечивающей для заданного объекта управления или процесса закон управления или управляющую последовательность воздействий, обеспечивающих максимум или минимум заданной совокупности критериев качества системы.

Оптимальное решение — решение, которое по тем или иным признакам предпочтительнее других.

Целевая функция — вещественная или целочисленная функция нескольких переменных, подлежащая оптимизации (минимизации или максимизации) в целях решения некоторой оптимизационной задачи.

Сокращения

НС – нейронная сеть

ЦПТ – центральная предельная теорема

Оглавление

Введение

Мы существуем среди систем, да и общество, человек всего лишь одна из подсистем окружающего мира. Наши знания и взаимодействие с окружающей нас средой всегда опосредованы через ту или иную систему, а наш посредник – это знание законов и принципов функционирования систем.

Объектами активных исследований уже долгое время являются развивающиеся открытые системы, находящиеся в неравновесном состоянии относительно окружающей среды. В развитии таких систем особый интерес вызывают ситуации, в которых протекают их переходы в качественно новые состояния [1].

Развивающиеся системы – это системы, функционирующие и развивающиеся в условиях воздействия внешних факторов и взаимодействия с ними.

Открытой системой называются системы, отличительной особенностью которых является способность обмениваться со средой массой, энергией и информацией. Любой организм является высокоорганизованной открытой неравновесной системой, примером которой главным образом выступает человек. Человек рождается и развивается в определенной внешней среде, которая активно взаимодействует с ним.

Одной из важнейших задач общества является забота о здоровье населения. Профилактика заболеваний, формирование здорового образа жизни по сей день является актуальной проблемой.

Реакция организма человека на внешнее воздействие зависит от функционального состояния, уровня функционирования и степени напряжения [1].

Цель данной работы – оценить функциональное состояние организма человека; выявить ситуации разладки (перехода в кризисное состояние); подобрать управляющее воздействие, позволяющее минимизировать отклонения интегрального показателя, характеризующего функциональное состояние организма в целом, от нормы. Для достижения поставленной цели

будут рассматриваются алгоритмы поиска оптимального управляющего воздействия, среди которых два эвристических (алгоритм симуляции отжига и генетический алгоритм) и еще один – алгоритм полного перебора. Для полного перебора используется поиск в ширину.

Для достижения поставленной цели были получены данные из гор. Больницы №4. (г. Томск) о состоянии здоровья 152 человек. В качестве входных данных на вход нейронной сети были поданы нормированные значения показателей крови (гемоглобин, эритроциты, ретикулоциты и т.д.) каждого пациента из рассматриваемой группы людей, проходивших своё лечение в больнице в течение определённого временного интервала. Этот набор характеристик в совокупности указывает на состояние здоровья человека.

1 Обзор литературы

Сложность количественной оценки состояния сложной системы, такой как организм человека, объясняется следующими факторами: [2]

- Индивидуальные особенности каждого человека;
- Многообразие внешних воздействий;

Существенной особенностью медико-биологических показателей является то, что при изменениях состояния биосистемы ряд из них находятся в пределах статистической нормы, в то время как другие выходят за пределы нормы. Это вносит большие трудности при оценки состояния биосистемы. Поэтому рассматриваются обобщенные интегральные показатели, которые построены по совокупности многомерных данных.

В данной работе для оценки степени отклонения состояния биосистемы от уровня нормального функционирования рассматривается подход, предложенный Н.В Бокучаевой и Г.В.Мамасахлисовым [3], который заключается в рассмотрении информационной меры Кульбака как меры предпочтительности поведения биосистемы:

$$I(t) = \int_t P_0(x) Ln \frac{P_0(x)}{P(x,t)} dx \,,$$

где $\boldsymbol{P_0(x)} -$ плотность вероятности текущего “равновесного” состояния;

$\boldsymbol{P(x,t)} -$ плотность вероятности нахождения биологической системы в состоянии x_t;

$\boldsymbol{x(t) = (x_1, \dots, x_n, t)} -$ зависящие от времени переменные, характеризующие состояние биологической системы.

Данный критерий позволяет оценить отклонение текущего состояния пациента от “предпочтительного”.

В работе В.А. Фокина в качестве оценки состояния биосистемы рассматривается показатель, который учитывает взаимное расположение объектов в областях.[4] На рисунке ниже представлена ситуация, когда

пренебрежение взаимного расположения объектов в областях S_1, S_2, S_3 приводит к одинаковой оценке близости объекта $\vec{x}$ к этим областям.

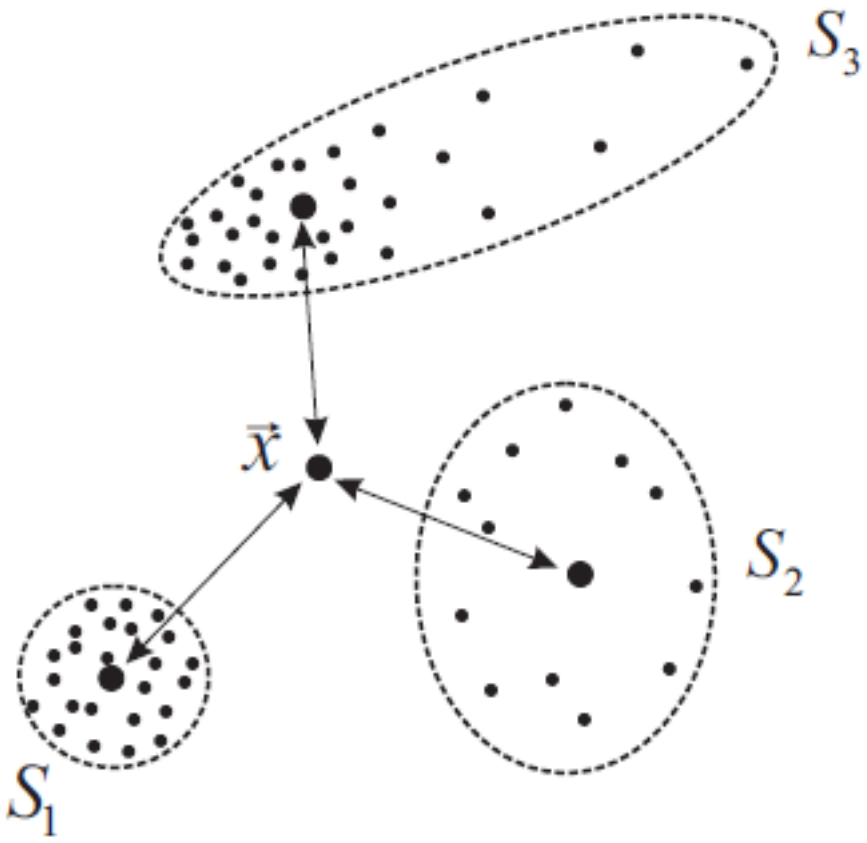


Рис. 1.1. Взаимное расположение объектов в областях

Во избежание такого случая мера близости рассматриваемого объекта к референсному состоянию нормируется на меру компактности то области, которую занимают объекты референсного состояния.

Таким образом, близость рассматриваемого объекта $\vec{x}$ к «эталонному» состоянию можно оценить следующей формулой:[5]

$$I_{S_0}(\vec{x}) = \frac{1}{2mN_{S_0}} \sum_{i=1}^{N_{S_0}} d_M(\vec{x}, \overrightarrow{x_i})$$

где

$d_M(\vec{x}, \overrightarrow{x_i})$ – расстояние Махаланобиса;

$m -$ количество показателей объекта;

$N_{S_0} -$ количество объектов референсного состояния.

В настоящей работе для поиска оптимального управляющего воздействия рассматривается алгоритм полного перебора (*breadth- first search*), который решает задачу поиска кратчайшего пути во взвешенном ориентированном графе. В нашей трактовке кратчайший путь – это такая последовательность управляющих воздействий во времени, которая доставляет функционалу минимальное значение. Применение алгоритма поиска в ширину, реализующего полный перебор, позволяет получить оптимальное решение задачи для рассматриваемого примера. Тем самым появляется возможность исследовать применимость генетического алгоритма и алгоритма симуляции отжига для рассматриваемого примера. [6]

В работе Кочетова Ю.А. рассматривается простейший вариант генетического алгоритма с различными вариантами скрещивания, среди которых, наибольший интерес представляет параметрический оператор скрещивания похожий на одноточечный кроссовер. Однако, в отличие от одноточечного оператора, разделитель выбирается не произвольно, а в соответствии со следующей формулой:[7]

$$k = \frac{F(u^l)}{F(u^l) + F(u^r)} \in (0, 1)\ ,$$

где $u^l -$ первый родитель;

$u^r -$ второй родитель;

$F -$ функция приспособляемости;

k принадлежит интервалу от 0 до 1.

Так же для генетического алгоритма рассматриваются два различных варианта селекции. Первый из них, наиболее знаменитый – это рулеточная селекция. Второй вариант селекции на каждой итерации подразумевает выбор лучшего

родителя в плане приспособляемости, а ему в пару – произвольно выбранного, отличного от него, чтобы избежать вырождаемости. [8]

Еще один из рассматриваемых алгоритмов решения задачи поиска оптимума – это алгоритм симуляции отжига, особенность этого алгоритма заключается в вероятностном переходе к «худшему» решению. Эта особенность защищает алгоритм от проваливания в локальный экстремум. Этот алгоритм, так же, как и генетический алгоритм, активно используется в нейронных сетях[9]. В работе А.С. Лопатина рассматриваются различные варианты отжига[10]. В данной работе рассматриваются Больцмановский отжиг, отжиг Коши и быстрый отжиг. Их отличие заключается в законе изменения температуры и способа выбора допустимого решения на основе предыдущего для возможного перехода к нему. Функция выбора допустимого решения для Больцмановского отжига и быстрого отжига моделируется с помощью стандартного нормального распределения и распределения Коши, соответственно. [10]

2 Постановка задачи

2.1 Содержательная постановка задачи

Была поставлена следующая задача: имеется таблица данных (обследований больных). Все пациенты, зафиксированные в этой таблице, разделены на группы лечения. Необходимо подобрать алгоритм, который будет генерировать группу лечения для каждого пациента, исходя из его показателей крови. Для решения задачи нужно определить, каким закономерностям подчиняются данные в таблице. С этой целью для каждого пациента в таблице данных строится интегральный показатель в соответствующий момент времени. Следует заметить, что размер таблицы – по восемь признаков на каждый из трех моментов времени и девяносто объектов. Данные таблицы фиксировались до лечения пациентов, после лечения и через месяц после лечения.

В качестве объекта исследования рассматривается организм человека (функциональное состояние беременной женщины) [5]. Предмет исследования – методы оценки функционального состояния на основе нейронных сетей, интегральных критериев и алгоритмов поиска оптимального решения. Необходимо построить модель, которая принимает на вход данные о здоровье человека и выбирает наиболее подходящий процесс лечения, то есть сообщает о группе, в которой будет лечиться пациент.

Первым шагом к решению задачи является оценка состояния биосистемы (организма человека). Для оценки состояния биосистемы используются интегральные критерии.

Следующий шаг – построение модели на основе интегральных критериев для прогнозирования интегрального показателя в последующие

моменты времени. В качестве модели рассматриваются нейронные сети с одним внутренним слоем.

Таким образом, для каждого пациента модель может построить набор интегральных кривых, соответствующих разным группам лечения. Эти интегральные кривые мы сравниваем с интегральной кривой нормального функционирования и выдаем ответ, какая из построенных кривых является наилучшей.

2.2 Математическая постановка задачи.

Функциональное состояние организма в каждый момент времени характеризуется интегральным показателем:

$$I_{\text{адапт}_i} = f(x_1, \dots, x_m), \ \ i = 1, \dots, N \qquad (1)$$

где $N -$ количество объектов (пациентов) в группе, $f -$ функция от значений признаков.

Кроме того, для каждого объекта исследования задано управляющее воздействие:

$$U = \{u_1, u_2, u_3, \dots\} \qquad (2)$$

Считаем, что в любой момент времени заданы желаемые значения интегрального критерия:

$$I_{\text{кр}}(t), \ \ t \in [t_0, t_k] \qquad (3)$$

Также задана функция потерь $- C_t$, рассчитывается как разность между желаемым значением функционального состояния и реальным:

$$C_t = I_{\text{кр}} - I_{\text{адапт}} \tag{4}$$

В этом случае функционал оптимизации может быть представлен в виде:

$$J(x;u) = \frac{1}{k}\sum_{i=t_0}^{t_k}\left(I_{\text{адапт}_i} - I_{\text{кр}_i}\right)^2 \;\rightarrow\; min\,, \tag{5}$$

где $J(x;u)-$ целевая функция, $I_{\text{адапт}_i}-$ прогнозные значения интегрального показателя.

3 Основные методы при решении задачи

3.1 Интегральный показатель

Одним из перспективных подходов для решения задачи оценки состояния биосистемы (организма человека) является применение энтропийных методов моделирования сложных систем.

Данный подход к анализу состояния биосистемы позволяет обойти множество таких свойств биосистемы как:

- нелинейность биосистемы;
- неполнота описаний состояния биосистемы;
- большая вариабельность и разнородная метрика параметров биосистемы;
- состояние биосистемы не может быть описано монотонной функцией;
- биосистемы подвержена большому числу состояний.

В данной работе для оценки степени отклонения состояния биосистемы от уровня нормального функционирования рассматривается подход, предложенный Н.В Бокучаевой и Г.В.Мамасахлисовым [3], который заключается в рассмотрении информационной меры Кульбака как меры предпочтительности поведения биосистемы:

$$I(t) = \int_t P_0(x) Ln \frac{P_0(x)}{P(x,t)} dx \ ,$$

где $\boldsymbol{P_0(x)} -$ плотность вероятности текущего “равновесного” состояния;

$\boldsymbol{P(x,t)} -$ плотность вероятности нахождения биологической системы в состоянии x_t;

$\boldsymbol{x(t) = (x_1, \dots, x_n, t)} -$ зависящие от времени переменные, характеризующие состояние биологической системы.

Данный критерий позволяет оценить отклонение (4) текущего состояния пациента (1) от “предпочтительного” состояния (3). Приняв в качестве “предпочтительного” состояния биосистемы состояние, при котором значения всех переменных состояния биосистемы заданы среднеквадратическим значениям физиологической нормы, в качестве критерия оценки текущего состояния биосистемы может быть использовано следующее выражение:

$$I = \frac{1}{n}\sum_{j=1}^{n} P_{0j} Ln \frac{P_{0j}}{P_j},$$

где $\boldsymbol{P_{0j}}$ – “предпочтительная” вероятность состояния;

$\boldsymbol{P_j}$ – вероятность того, что значение переменной состояния X соответствует “норме”.

n – количество показателей, характеризующих состояние биосистемы.

В формуле для интегрального критерия предполагается, что “предпочтительная” вероятность состояния P_{0j} равна единице. Таким образом, выражение для I приобретает вид:

$$I = \frac{1}{n}\sum_{j=1}^{n} Ln \frac{1}{P_j},$$

где $\boldsymbol{P_j}$ выражается в следующем виде:

$$P_j = P\left(\left|x_j - x_0\right| < \delta\right) = 2\Phi\left(\frac{\delta}{\sigma}\right) - 1,$$

Ф – функция Лапласа;

$\boldsymbol{\delta}$ – величина отклонения признака от его среднего значения этого признака:

$$\delta = \left|x_j - x_{\text{норм}}\right|$$

$\boldsymbol{\sigma}$ – среднеквадратическое отклонение признака:

3.2 Алгоритм поиска в ширину для выбора управляющего воздействия

Имеется следующая ситуация:
В каждый из рассматриваемых моментов времени имеются на выбор три управляющих воздействия. Причем результат управляющего воздействия в первый момент времени зависит от набора показателей крови. В остальные же моменты времени результат воздействия зависит от воздействия, выбранного в предыдущий момент времени.

Если эту постановку структурировать, то выбор управляющего воздействия в последующие моменты времени может быть представлен в виде дерева $T = \langle V, E \rangle$:

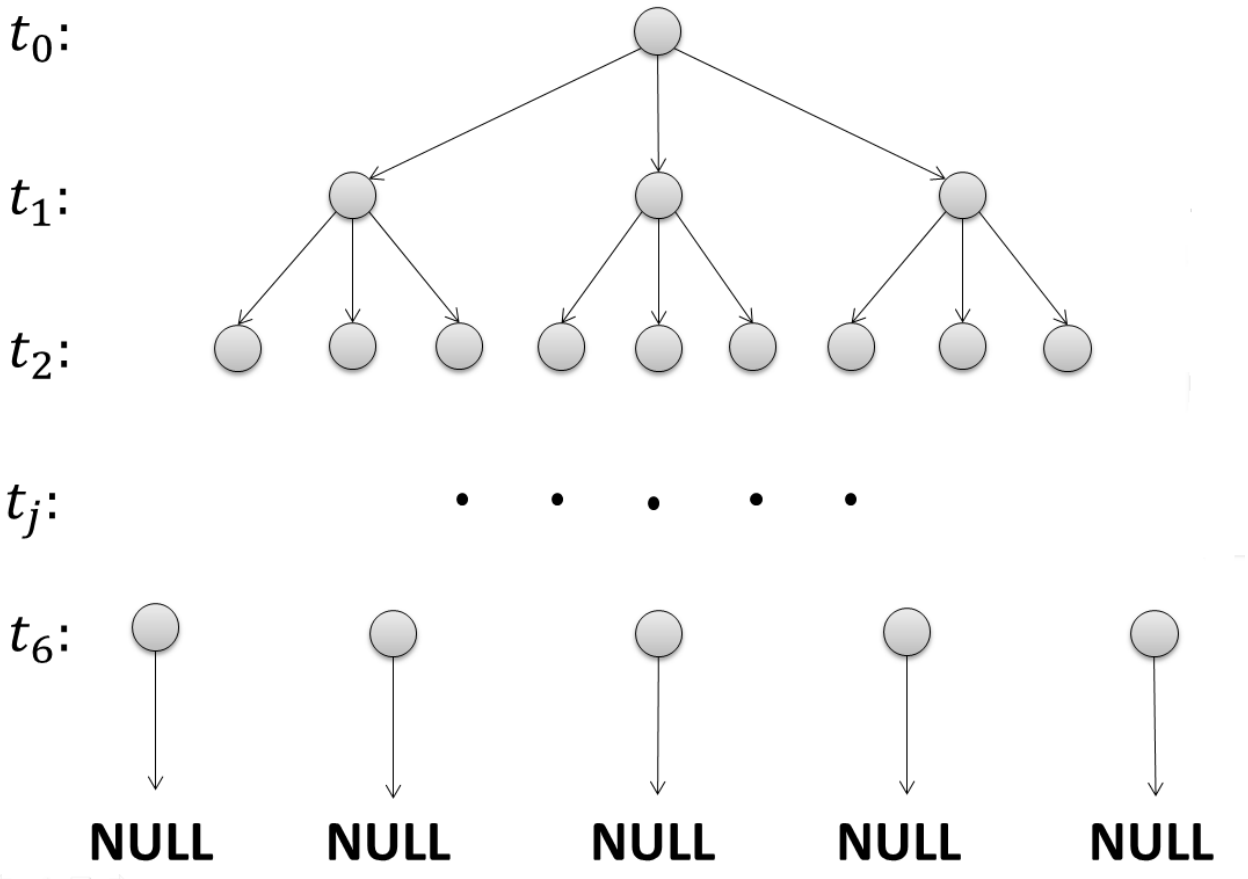


Рисунок 2 – Структура дерева для данной решаемой задачи

Дерево $T = \langle V, E \rangle$ имеет семь уровней. На нулевом уровне располагается корень дерева. Это вершина, на которой содержатся значения показателей крови пациента. На последующих уровнях располагаются вершины–воздействия.

Каждая вершина имеет одного родителя (предыдущее воздействие) и три дочерних элемента. Более того каждая вершина умеет вычислять вклад соответствующего управляющего воздействия в целевую функцию с учетом родительского элемента. Также для каждой вершины определены флаг, который сигнализирует о ее посещении и пара номеров, указывающая позицию вершины во времени и номера управляющего воздействия.

Структура вершины программно выглядит следующим образом:

```
struct Node {
    bool used;
    double dist;
    Node * parent;
    std::vector<Node*> childs;
    std::pair<int, int> position;

    Node(Node * _parent) {
        parent = _parent;
        used = false;
        for (int i = 0; i < childsAmount; i++) {
            childs.push_back( NULL );
        }
    }
};
```

Каждая вершина нижнего уровня ссылается на пустой элемент ($NULL$), который играет роль заглушки.

Цель обхода дерева $T = \langle V, E \rangle$ состоит в том, чтобы посетить все элементы, вычислить соответствующее слагаемое целевой функции (путь до этой вершины от корня). Очевидно, что вершины нижнего уровня представляют собой конечное значение функционала $J(x; u)$.

Сформулируем алгоритм обхода в ширину:

1. Выбрать начальную вершину v_0 и поместить ее в очередь.
2. Пока очередь не пуста, делать
   2.1. Извлечь из очереди вершину, пометить ее как посещенную и вычислить длину пути до этой вершины.

- Если эта вершина нижнего уровня, то сравнить с минимальным значением целевой функции и вернуться в пункт 2.
- Иначе поместить в очередь все ее дочерние элементы.

3. Если очередь пуста, то все вершины дерева просмотрены, значения функционала на нижнем уровне вычислены и найден минимальный элемент.

Таким образом, алгоритм поиска в ширину находит вершину дерева, значение функционала которой является минимальным. Остается найти минимальный путь, то есть тот путь, который приводит в эту вершину (доставляет функционалу $J(x;u)$ минимальное значение). Для этого применяется обратный ход алгоритма, который заключается в следующем:

1. Поместить в стек найденную минимальную вершину нижнего уровня.
2. Делать до тех пор, пока не дойдем до корня дерева
   2.1. Поместить вершину−родителя в стек;
   2.2. Вернуть рекурсивно вершину−родителя.

Таким образом, стек будет состоять из тех вершин, которые образуют минимальный путь.

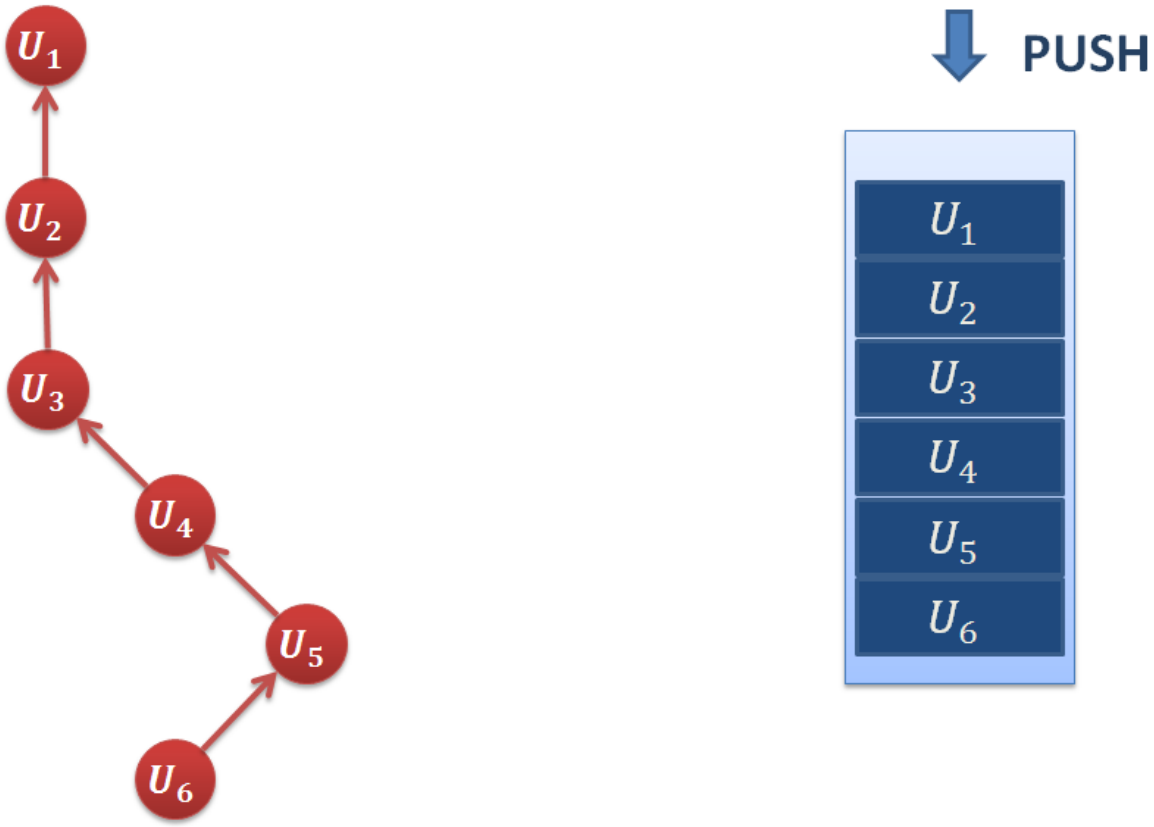


Рисунок 3 – Структура оптимального решения

После развертывания этого стека (последовательное извлечение элементов), получается оптимальное решение задачи:

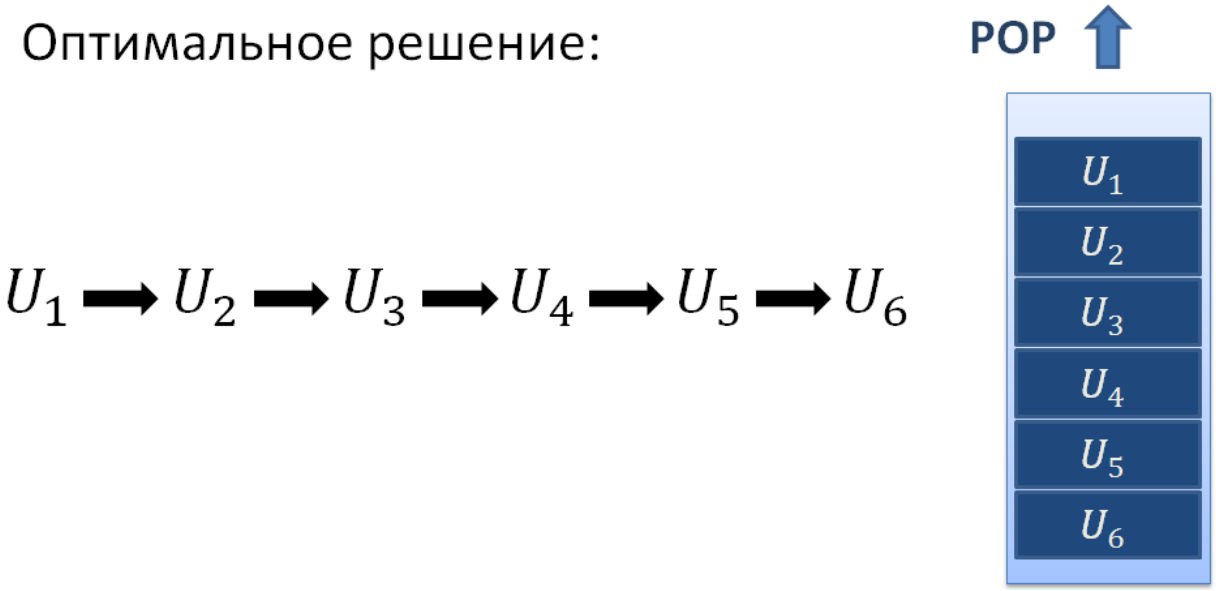


Рисунок 4 – Оптимальное решение задачи методом перебора

Таким образом, на конкретном примере, который представляет собой набор показателей крови человека, и наборе управляющих воздействий, применяется алгоритм поиска в ширину, который реализует полный перебор допустимых решений. Этот алгоритм находит оптимальное решение задачи, то есть тот набор воздействий, который доставляет функционалу (5) минимальное значение. Это оптимальное решение будет использовано для исследования применимости различных вариантов генетического алгоритма, а так же отжига Коши, быстрого отжига и Больцмановского отжига к рассматриваемой задачи.

3.3 Генетический алгоритм

Генетический алгоритм – это эвристический алгоритм поиска. Этот алгоритм используется для решения задач оптимизации и моделирования путем случайного подбора, комбинирования и вариации искомых параметров с

использованием механизмов, аналогичных естественному отбору в живой природе. В генетическом алгоритме используются методы естественной эволюции, такие как наследование, мутация, отбор и кроссинговер. Отличительной особенностью генетического алгоритма является лишь акцент на оператор скрещивания, который производит операцию рекомбинации решений – кандидатов, роль которой аналогична роли скрещивания в природе.

За счет того факта, что данный алгоритм самообучающийся, спектр его применения очень широк:

- Задачи на графы (задачи коммивояжера, раскраска, нахождение паросочетаний)
- Задачи компоновки
- Составление расписания
- Игровые стратегии
- Аппроксимация функций
- Биоинформатика
- Задачи о назначениях
- Оптимизация функций
- Настройка и обучение искусственных нейронных сетей

Генетические алгоритмы оперируют совокупностью особей (популяцией). Каждая особь представляет собой вектор, кодирующий одно из решений задачи.

$$u = [u_1, u_2, \dots, u_n]$$

Этот вектор u принято называть хромосомой, а его компоненты u_i – генами.

Каждая хромосома обладает функцией приспособленности. Эта функция позволяет понять, насколько рассматриваемое решение хорошее. За счет этой функции особи делятся на два типа:

- Самые приспособленные (наиболее подходящие решения). Эти особи остаются в популяции и получают возможность участвовать в скрещивании и давать потомство.
- Наименее приспособленные (плохие решения), постепенно удаляются из популяции (вымирают) и не дают потомства.

Таким образом, происходит естественный отбор, в результате которого приспособленность нового поколения все время в среднем становится выше предыдущего. Постановка задачи генетического алгоритма выглядит следующим образом:

Предполагается, что задана некоторая функция $f(u)$, где $u \in U$. Множество U дискретно, состоит из конечного числа элементов:

$$U = \bigcup_{s \in S} u_s \ , S \subset R$$

Необходимо найти такой элемент u множества U, который бы минимизировал исходную функцию:

$$\min_{u \in U} f(u)$$

Функция f может быть как функцией одной переменной $(f: R^1 \to R)$, так функцией нескольких переменных $(f: R^n \to R)$.

В рассматриваемой задаче размерность функции f зависит от количества рассматриваемых моментов времени, т.е. каждый элемент из множества допустимых решений U состоит из шести компонент.

Таким образом, в качестве функции f рассматривается функционал $J(x; u)$, где $u = (u_1, \dots, u_6)$, $u_i -$ управляющее воздействие в момент времени i.

Общая схема генетического алгоритма может быть представлена в виде:

1. Выбрать начальную популяцию P и запомнить рекорд $F^* = \min\limits_{i=1,\dots,k} f(S_i)$
2. Пока не выполнен критерий остановки делать следующее:
   2.1. Выбрать «родителей» S_{i_1}, S_{i_2} из популяции.

2.2. Применить к S_{i_1}, S_{i_2} оператор скрещивания и получить новое решение $S^{'}$.

2.3. Применить к $S^{'}$ оператор мутации и получить решение $S^{''}$.

2.4. Применить к $S^{''}$ оператор локального улучшения и получить новое решение $S^{'''}$.

2.5. Если $f(S^{'''}) < F^*$, то сменить рекорд $F^* := f(S^{'''})$.

2.6. Добавить $S^{'''}$ в популяцию и удалить из нее наихудшее решение.

Формирование начальной популяции заключается в выборе решений (особей) из множества допустимых решений U. Также определяется размер популяции. Результат работы генетического алгоритма напрямую зависит от выбора начальной популяции и от ее размера. Элементы из множества U в популяцию заносятся произвольно.[9]

Алгоритм формирования популяции из n элементов:

1. Повторить n раз:
   1.1. Положить $i = 1$.
   1.2. Повторить m раз:
      1.2.1. Реализовать случайную равномерно распределенную величину $\xi(u_i) \in S, i = 1 \dots m$.
      1.2.2. Выбрать соответствующий элемент u_{i_ξ}, который представляет собой $i-$ й ген решения, соответствующий реализации ξ.
      1.2.3. Увеличить порядковый номер гена на единицу: $i := i + 1$.
   1.3. Записать полученную хромосому $u = \left[u_{1_\xi}, \dots, u_{n_\xi}\right]$ в популяцию.

$n-$ размер популяции;

$m-$ длина хромосомы;

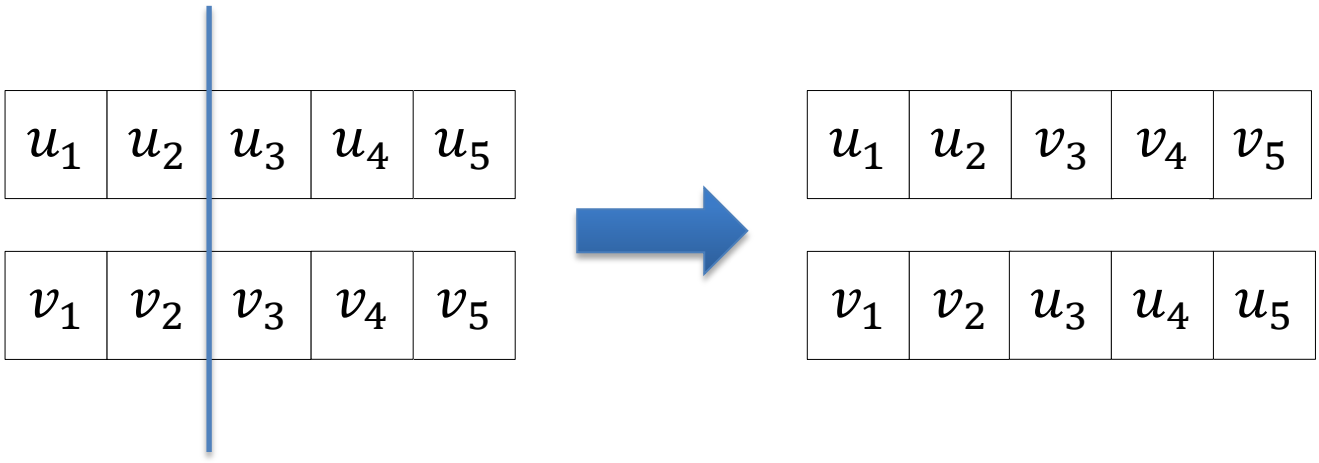
Оператор скрещивания представляет собой скрещивание двух объектов. Имеются два решения $u^1, u^2 \in U$ и процедура, которая позволяет

сделать алгебраическую операцию $u^1 æ_d u^2$, где d принадлежит некоторому семейству индексов ($d \in D \subset R$). В генетическом алгоритме операция скрещивания рассматривается параметрически (параметр d регулирует способ скрещивания) в отличие от размножения биологических особей, где половина генов берется от отца, а другая половина от матери. В данной работе параметр $d -$ случайное число от 1 до m.

Существуют различные вариации оператора скрещивания. В работе рассматриваются: [10]

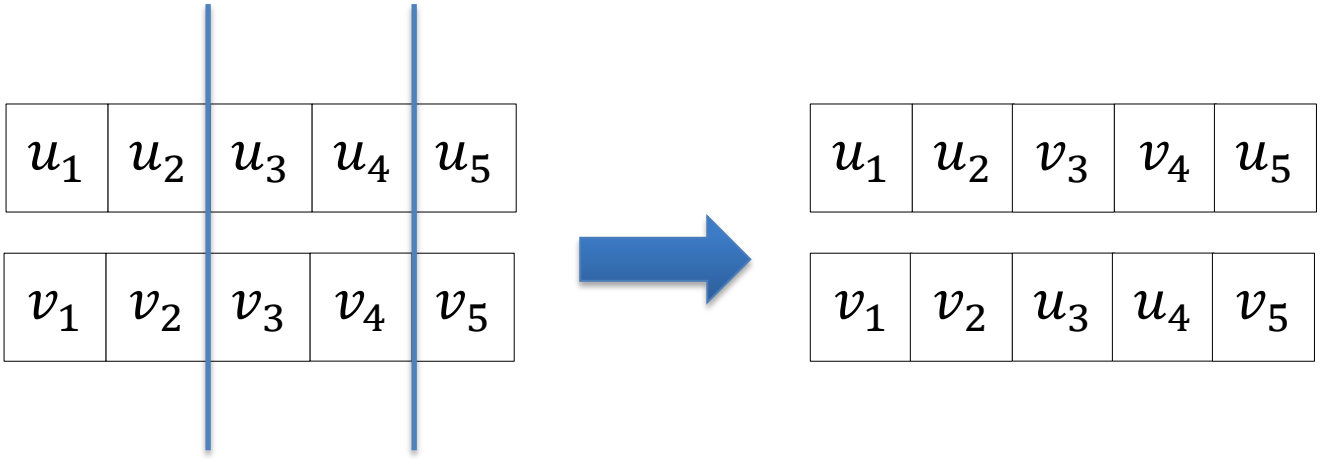
- **одноточечный оператор скрещивания**

  параметр d оператора скрещивания выбирается случайно. Результатом оператора $æ_d$ являются два потомка. Один из этих потомков присваивает себе часть генов первого родителя, которая находится левее разделителя d, и часть генов второго родителя, которая находится правее разделителя d. Другой же потомок, наоборот, первую часть генов берет у второго родителя, оставшуюся часть – у первого. Таким образом, формируются два потомка. Каждый из этих потомков имеет одинаковый шанс быть выбранным ($p = 0.5$).



- **двухточечный оператор скрещивания**

случайным образом выбираются два разделителя. Так же, как в случае одноточечного оператора скрещивания, результатом $æ_d$ будут две особи. Одна особь возьмет часть генов у первого родителя до первого разделителя и после второго разделителя, другую часть – у второго родителя между разделителями. Вторая особь возьмет часть генов у второго родителя до первого разделителя и после второго разделителя, оставшуюся часть – у первого родителя между разделителями. Далее, с одинаковой вероятностью выбирается один из двух потомков.



- **оператор скрещивания, зависящий от родителей**

  Рассматривается еще один вариант оператора скрещивания, похожий на одноточечный оператор. Однако, в отличие от одноточечного оператора, разделитель выбирается не произвольно, а в соответствии со следующей формулой:[5]

$$k = \frac{F(u^l)}{F(u^l) + F(u^r)} \in (0,1)\,,$$

где u^l – первый родитель;

u^r – второй родитель;

F – функция приспособляемости;

k принадлежит интервалу от 0 до 1.

Значение разделителя d высчитывается следующим образом: интервал $(0,1)$ делится на шесть равных частей. Параметр d будет равен порядковому номеру части интервала $(0,1)$, в который попало значение k.

Оператор мутации предполагает случайное изменение генов в решении. Этот оператор представляет собой отображение из множества допустимых решений в множество допустимых решений:

$$g_m : U \to U\,, m \in M$$

Оператор мутации является вероятностным. С определенной вероятностью исходное решение, которое подвергается мутации, может измениться как в лучшую сторону (увеличится приспособляемость), так и в худшую (приспособляемость уменьшится). Также благодаря этому оператору генетический алгоритм защищен от проваливания в локальный экстремум (локальный минимум). Оператор мутации может быть реализован следующими способами:

- с малой вероятностью $p < \frac{1}{n}$ в каждой координате значение $u_i \in \{0,1\}$ заменяется на противоположное $1 - u_i$.

- Если же в решении требуется сохранить $\sum_{i \in I} u_i = p$, то случайным образом выбирается координата i_1, что $u_{i_1} = 1$ и координата i_2, такая, что $u_{i_2} = 0$ и производится замена $u_{i_1} := 0,\ u_{i_2} := 1$.

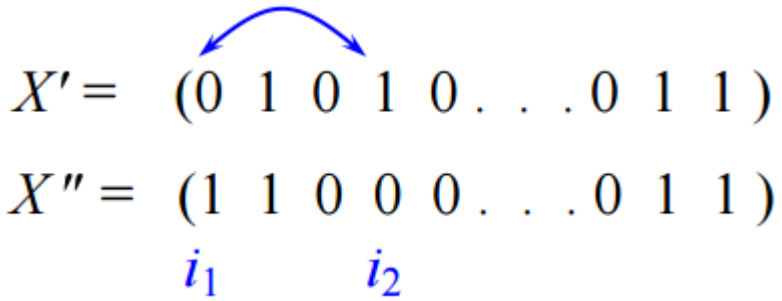
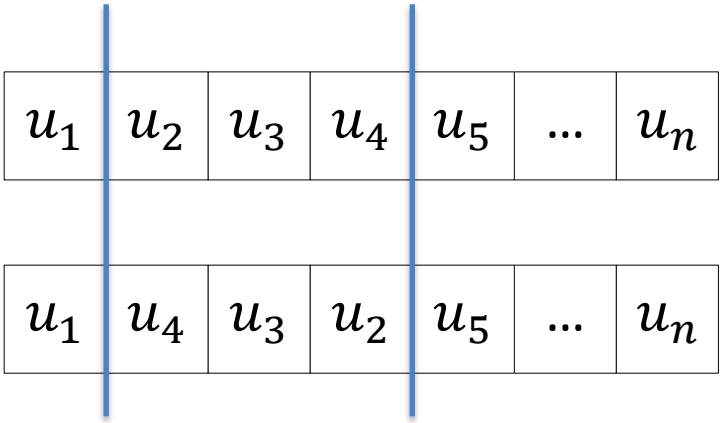


Рисунок 5 – Результат оператора мутации

Оператор инверсии работает с одной хромосомой. Этот оператор меняет последовательность генов в хромосоме между двумя границами. В настоящей работе границы инверсии выбираются случайным образом:



Селекция хромосом заключается в выборе (по рассчитанным значениям функции приспособленности) тех хромосом, которые будут участвовать в создании потомков для следующей популяции, т.е. для очередного поколения. Такой выбор производится согласно принципу естественного отбора, по которому наибольшие шансы на участии в создании новых особей имеют хромосомы с наибольшим значение функции приспособленности. В работе рассматриваются следующие виды селекции:

- **Метод рулетки**

  Отбираются хромосомы путем запуска рулетки. Рулетка поделена на сектора, которые соответствуют отдельным хромосомам. Причем величина сектора зависит от функции приспособляемости особи и в случае решении задачи минимизации может быть вычислена по следующей формуле:

$$P_{sel}(i) = \frac{1}{N-1}\left(1 - \frac{f(i)}{\sum_{i=1}^{n} f(i)}\right)$$

  Данная формула пригодна для задачи минимизации. Размер каждого сектора обозначается в процентах. Величина всей рулетки $-$ 100%.

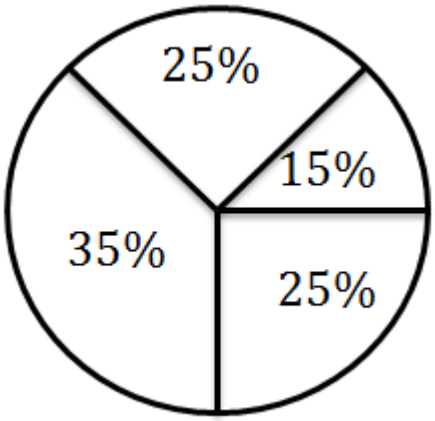


Рисунок 6 – Взаимное расположение объектов в областях

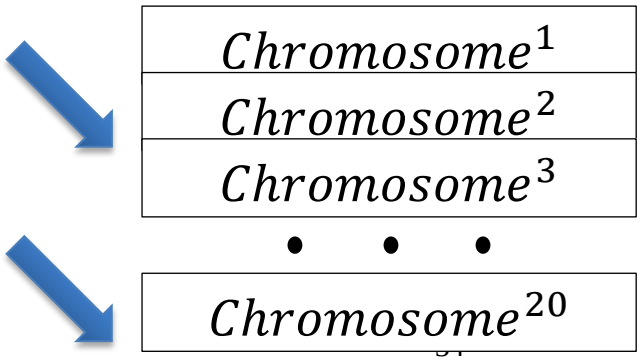
При таком отборе члены популяции с более высокой приспособленностью имеют больший сектор и, соответственно, больший шанс быть выбранными, нежели те хромосомы, функция приспособленности которых минимальна.

- **Пропорциональная селекция** (*proportional selection*) :

  Из популяции случайным образом выбираются два родителя. Для решения S_i вероятность быть выбранным обратно пропорциональна значению целевой функции $f(S_i)$. Родители для скрещивания выбираются до тех пор, пока они не будут различны.

- **Случайно выбранный + наиболее удаленный от него**:

  Из популяции случайным образом выбирается хромосома (первый родитель). Далее, выбирается та хромосома, которая занимает самую удаленную позицию от первой.



- **Лучший + случайно выбранный**

  В качестве первого родителя выбирается та особь, которая имеет рекордную функцию приспособляемости (минимальное значение фитнесс-функции). Второй родитель выбирается случайным образом. Чтобы избежать вырождения (два одинаковых родителя), вторая хромосома должна отличаться от первой. [6]

Рассмотрим вопрос, связанный с критерием остановки генетического алгоритма. Существуют различные варианты критериев остановки:

1. Достижение алгоритмом определенного числа итераций;
2. Истечение времени, отпущенного на работу алгоритма;
3. Достижение оптимального решения;
4. Схождение популяции.
5. На основе производных

Можно установить для работы алгоритма определенное число итераций. Тогда существует вероятность, что алгоритм попросту не сможет за отпущенное ему число поколений найти оптимальное решение. С другой стороны, алгоритму может потребоваться гораздо меньше итераций для нахождения оптимума и оставшуюся часть итераций он проработает вхолостую. То же самое можно сказать про такой критерий остановки как задание времени на работу. Если же оптимальное решение задачи известно наперед и необходимо проверить, какой из вариантов генетического алгоритма справляется с задачей быстрее, то можно воспользоваться критерием остановки под номером 3. Для решения задач на поиск экстремума оптимальное решение неизвестно наперед. Хотелось бы понимать, когда нужно остановить работу генетического алгоритма. В данной главе рассматривается последний вариант критерия остановки.

Под схождением понимается такое состояние популяции, когда ни операция кроссовера, ни операция мутации не вносят изменения в генетическое разнообразие популяции в течение нескольких поколений.

При такой трактовке схождения популяции возникают следующие проблемы:

1. Неизвестно количество поколений достаточное для отслеживания неизменности популяции;
2. Нет четкого определения сошедшейся популяции.

Поэтому такой подход неприемлем для задач оптимизации. Поэтому предлагается в качестве условия остановки использовать метрику, оценивающую различие особей в популяции. Каждая особь в популяции представляет собой допустимое решение и описывается своим генотипом и фенотипом. Под генотипом понимается запись решения в закодированном виде (с помощью 0 и 1), а фенотип представляет собой декодированное решение. Исходя из такого описания возможно определение двух метрик, с помощью которых можно оценить различие особей, для генотипа и для фенотипа. Для сравнения генотипов особей используется Хеммингово расстояние, которое равно количеству различающихся позиций в хромосомах. Для сравнения фенотипов используется евклидово расстояние.

Хемингово расстояние следующей формулой:

$$\rho_H(X_i, X_j) = \sum_{l=1}^{m} |x_{il} - x_{jl}|$$

$\rho_H(X_i, X_j)$ равно числу несовпадений значений соответствующих признаков в X_i и X_j.

Вводится характеристическая метрика популяции, которая будет оценивать различие всех особей в популяции. Эта метрика рассчитывается по следующей формуле:

$$\bar{e} = \frac{1}{N}\left(\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\sum_{l=1}^{m}|x_{il} - x_{jl}|\right)\right)$$

Однако если рассчитывать расстояние на каждой итерации, алгоритм будет довольно требователен к вычислительным ресурсам, поэтому

предлагается использовать абсолютное значение разности средних приспособленностей популяции:

$$\left|f_{\text{ср}}^{l+1} - f_{\text{ср}}^{l}\right|$$

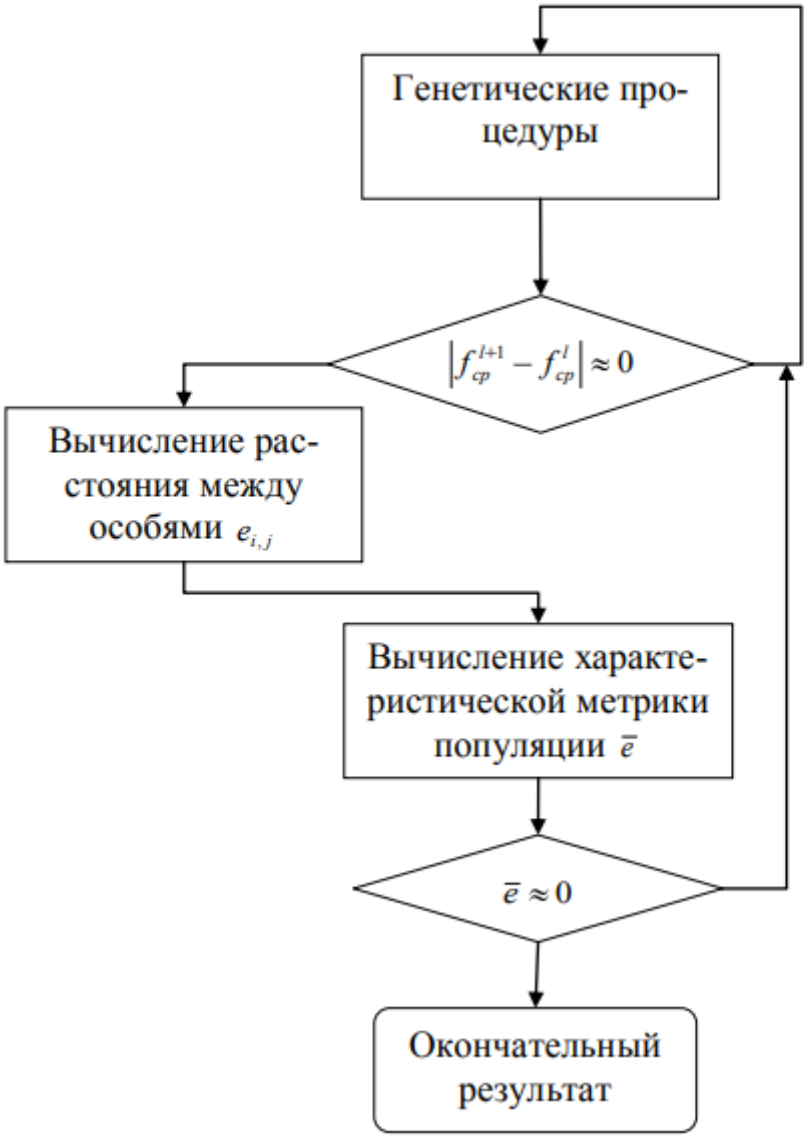


Рисунок 7 – Неполная таблица начальных данных

Если же эта величина не превосходит наперед заданного малого ε или же вовсе равна нулю, то переходим к вычислению расстояния между особями популяции вычисления характеристической метрики популяции. Таким образом, критерий остановки состоит из двух уровней проверок.

Рассмотрим еще вариант останова 5, который основывается на вычислении первой и второй производной.

Как известно, если градиент в какой-либо точке равен нулю, то эта точка является критической. Поэтому оценим норму градиента наперед заданным малым ε_1:

$$\left\|\nabla J\left(X^{(N+1)}\right)\right\|_2 = \left\|\left(\frac{\partial J}{\partial u_1}, \dots, \frac{\partial J}{\partial u_6}\right)\left(X^{(N+1)}\right)\right\|_2 = \left(\sum_{i=1}^{6}\left[\frac{\partial J}{\partial u_i}\left(X^{(N+1)}\right)\right]^2\right)^{\frac{1}{2}} < \varepsilon_1$$

Где $X^{(N+1)} = (u_1, \dots, u_6) -$ лучшее решение в $(N+1) -$ й популяции

Однако это условия не является достаточным для существования экстремума в этой точке(в «седловой точке» градиент обращается в ноль, но она не является экстремумом). Достаточным же условием является знакоопределенность гессиана функционала J, а точнее его квадратичной формы:

$$H(u) = \sum_{i=1}^{6}\sum_{j=1}^{6}\frac{\partial^2 J}{\partial u_i \partial u_j}u_i u_j$$

Если квадратичная форма остается положительно определенной при переходе через критическую точку, то алгоритм находится возле минимума функции. Если же квадратичная форма отрицательно определена, то мы имеем максимум. Если же квадратичная форма является не знакоопределенной, то критическая точка является седловой.

3.4 Алгоритм симуляции отжига ($SA -$ алгоритм)

Следующий метод выбора управляющего воздействия (2), который рассматривается в этой работе, это алгоритм симуляции отжига (SA-метод). Этот метод предназначен для нахождения оптимумов (максимумов, минимумов) функций, заданных на различных структурах. Как и генетический алгоритм этот метод относится к эвристическим методам нахождения оптимума. Поэтому метод отжига не всегда может давать гарантированный

результат. С другой стороны, он позволяет решать $NP-$ полные задачи за довольно небольшое количество шагов.

Постановка задачи для метода имитации отжига ничем не отличается от постановки задачи для генетического алгоритма и выглядит следующим образом:

Дано множество допустимых решений U. Предполагается, что это множество не пусто. Также на этом множестве U определена некоторая функция $f(u), u \in U$ (т.е. каждому элементу множества U сопоставляется некоторое число). Требуется найти минимум этой функции $f(u)$, поэтому предполагается, что

$$\inf_{u \in U} f(u) > -\infty$$

То есть функция $f(u)$ должна быть ограниченной снизу (если $f(u)$ не ограничена снизу, то задача о поиске минимума становится бессмысленной). Более того, не ограничивая общности, полагается, что функция f на множестве U является неотрицательной: $f(u) \geq 0 \; \forall u \in U$.

Требуется найти такой элемент $u^* \in U$ что $f(u^*) = \min_{u \in U} f(u)$.

Для конструирования метода имитации отжига потребуется ввести некоторые понятия:

$S-$ набор индексов (множество является конечным или счетным). Рассматривается случайная величина ξ, которая принимает значения из множества S. Также рассматривается множество преобразований множества U в себя:

$$g_s : U \to U, \qquad s \in S$$

Предполагается, что g_s является малым преобразованием:

$$\exists \, M > 0 : \; \forall s \in S \; \rho(g_s(u), u) \leq M$$

где $\rho -$ это некоторая метрика на множестве S.

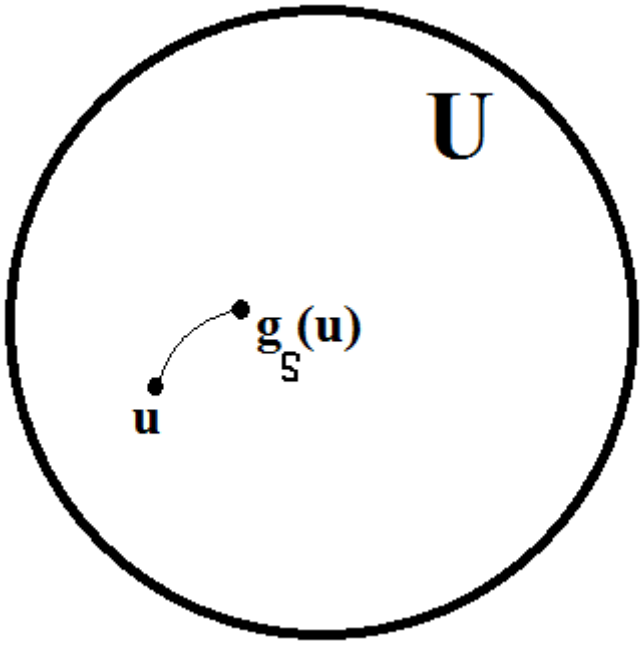


Рисунок 8 – Преобразование функции на множестве допустимых решений

SA – метод использует отображение, которое принято называть температурой:

$$T : S \to R$$

эта функция сопоставляет номеру итерации соответствующее вещественное число. Также эта функция обязательно должна быть убывающей.

Вводится еще одно отображение, которое называется функцией энергии:

$$E : U \to R$$

Функция энергии каждому элементу из множества U допустимых решений ставит в соответствие некоторое вещественное число.

Ниже приведен алгоритм метода SA:

1. $k = 0$
2. Устанавливаются максимальная t_{max} и минимальная (конечная) t_{min} температуры
3. Задается начальное состояние $u_0 \in U$
4. Пока температура не достигнет минимального значения делать
   4.1. $\tilde{u} = g_{\xi}(u_k)$, где ξ – случайная величина
   4.2. $\Delta = f(\tilde{u}) - f(u_k)$
   4.3. Если $\Delta \leq 0$ то $u_{k+1} = \tilde{u}$

4.4. Если $\Delta > 0$ то $\eta \sim R(0,1) -$ случайная величина. Если $\eta < \exp\left\{-\frac{\Delta}{t_k}\right\}$, то $u_{k+1} = \tilde{u}$ иначе $u_{k+1} = u_k$

4.5. $k = k + 1$

4.6. $t_{k+1} = T(k)$

t_{max} выбирается произвольно и должна соизмеряться с той функцией, которую необходимо минимизировать. Начальное состояние u_0 также выбирается произвольно (может быть выбрано случайное значение или фиксированное).

Отличительной особенностью данного алгоритма от градиентных алгоритмов является то, что он может делать не оптимальные шаги (принимать худшие решения), в то время как последние худшие решения всегда отбрасывают. Поскольку $SA -$ алгоритм с определенной вероятностью принимает худшие решения, его называют стохастическим (вероятностным). Эта особенность – двигаться не только в сторону улучшения решения, но и немного отступать назад, защищает алгоритм от проваливания в локальный минимум.

Как и в случае генетических алгоритмов, у $SA -$ алгоритма также существует множество различных реализаций, которые отличаются друг от друга способом генерации новых состояний и функцией изменения (понижения) температуры.

Вероятностный переход к худшему решению осуществляется по следующему принципу: генерируется случайное число в диапазоне от 0 до 1. Если это число находится в интервале $\left[0,\ \exp\left\{-\frac{\Delta}{T_k}\right\}\right]$, то решение принимается (алгоритм отступает назад). В противном случае алгоритм остается на месте. Можно сделать следующий вывод: чем больше температура, тем больше экспонента $\exp\left\{-\frac{\Delta}{T_k}\right\}$ и соответственно больше вероятность принятия худшего решения. Однако температура со временем снижается и большой она остается только на первых итерациях. Поэтому у алгоритма есть возможность «погулять» по множеству допустимых решений на первых итерациях. Со

временем вероятность принятия худшего решения уменьшается в соответствии с выбранным законом изменения температуры и алгоритм все больше вынужден принимать только лучшие решения.

4 Результаты моделирования

В качестве исходных данных рассматривается следующая таблица, которая представлена ниже в неполном виде:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	0	1	118	135	129	3,48	4,20	3,83	7			4,40		6,00	15,0		11,5	49
2	0	2	108	110	117	3,90	4,10	4,20	10			5,30		8,00	13,2		14,7	57
3	0	3	116	110	118	3,26	3,46	3,75	11			5,80		7,00	14,8		12,9	47
4	0	4	130	106	113	4,40	3,57	4,40	12			4,70		6,20	12,7		11,9	60
5	0	5	114	120	115	3,31	3,80	4,05	9			10,60		12,00	11,0		21,7	65
6	0	6	112	114	120	3,35	3,16	4,30	10			9,70		10,00	15,8		22,7	55
7	0	7	111	104	118	3,76	3,73	4,01	8			4,90		5,70	19,9		17,0	58
8	0	8	112	115	117	3,67	3,65	3,90	5			4,60		4,90	22,9		19,8	57
9	0	9	112	113	110	3,70	3,70	3,32	4			8,60		9,00	8,4		10,0	53
10	0	10	113	115	110	4,11	4,10	4,00	8			7,30		8,60	10,5		12,0	43
11	0	11	130	121	121	4,10	3,94	3,77	9			7,60		7,98	10,4		10,0	59
12	0	12	118	135	129	3,48	4,20	3,90	9			6,70		6,90	13,6		16,0	65
13	0	13	113	110	117	3,90	4,10	4,20	10			5,80		6,50	19,5		17,0	49
14	0	14	112	110	118	3,26	3,46	3,75	11			6,50		9,00	17,0		12,7	51
15	0	15	130	106	113	4,40	3,53	3,50	4			8,50		12,00	16,9		17,1	55
16	0	16	114	120	115	3,31	3,80	4,03	11			6,05		6,20	14,9		12,9	62
17	0	17	110	104	120	3,35	3,16	4,30	16			7,20		11,00	18,1		24,7	68
18	0	18	111	104	112	3,76	3,73	3,18	6			6,90		9,30	12,7		21,8	51
19	0	19	112	115	117	3,67	3,65	3,90	8			7,80		9,70	14,5		27,9	58
20	0	20	112	113	105	3,70	3,70	3,32	9			9,60		9,80	21,8		21,0	49
21	0	21	121	119	117	4,11	4,10	4,00	9			6,90		8,90	7,9		9,1	54
22	0	22	130	121	121	4,10	3,94	3,77	10			13,50		8,94	21,4		20,4	64
23	0	23	118	135	129	3,48	4,20	3,90	2			8,60		9,54	20,1		23,0	53
24	0	24	108	110	117	3,90	4,10	4,20	12			17,40		8,95	20,9		21,1	51
25	0	25	109	110	118	3,26	3,46	3,75	11			6,90		7,55	12,6		10,9	50
26	0	26	130	106	113	4,40	3,57	4,00	9			8,40		9,90	25,9		22,8	55
27	0	27	114	120	115	3,31	3,80	4,03	13			9,50		12,80	20,5		21,9	64
28	0	28	111	109	114	3,76	3,73	3,18	11			8,30		9,12	10,7		10,1	67
29	0	29	112	115	117	3,67	3,65	3,90	8			6,98		8,92	15,6		13,7	66
30	0	30	112	113	135	3,70	3,70	3,32	6			12,54		12,30	17,5		19,7	63
31	**1**	**31**	**105**	**109**	**102**	**3,50**	**3,94**	**3,21**	**2**	**10**	**5**	**8,90**		**9,12**	**5,2**	**8,3**	**5,9**	**66**
32	**1**	**32**	**103**	**112**	**100**	**3,53**	**3,70**	**4,50**	**5**	**9**	**8**	**9,23**		**11,90**	**3,2**	**12,0**	**9,0**	**69**
33	**1**	**33**	**107**	**115**	**112**	**3,60**	**3,63**	**3,21**	**9**	**8**	**10**	**8,32**		**9,13**	**6,4**	**7,1**	**6,3**	**71**
34	**1**	**34**	**103**	**120**	**104**	**3,70**	**3,20**	**3,32**	**8**	**7**	**13**	**7,31**		**9,40**	**4,9**	**11,0**	**7,0**	**75**
35	**1**	**35**	**105**	**106**	**109**	**3,43**	**3,50**	**3,62**	**12**	**10**	**9**	**10,30**		**12,50**	**5,0**	**10,5**	**8,4**	**62**
36	**1**	**36**	**107**	**111**	**107**	**3,04**	**3,40**	**3,10**	**3**	**9**	**7**	**14,20**		**15,00**	**5,2**	**6,9**	**6,7**	**78**
37	**1**	**37**	**102**	**106**	**105**	**4,28**	**3,60**	**3,92**	**5**	**11**	**9**	**9,43**		**9,30**	**3,2**	**8,0**	**6,7**	**81**
38	**1**	**38**	**103**	**107**	**103**	**3,28**	**3,59**	**4,00**	**7**	**9**	**8**	**18,20**		**16,00**	**5,8**	**7,7**	**9,8**	**59**
39	**1**	**39**	**106**	**118**	**110**	**4,05**	**3,47**	**3,80**	**9**	**12**	**11**	**14,80**		**12,40**	**5,9**	**12,0**	**11,5**	**81**
40	**1**	**40**	**100**	**107**	**110**	**3,83**	**3,60**	**3,60**	**4**	**11**	**6**	**9,04**		**11,70**	**4,9**	**7,4**	**4,9**	**76**
41	**1**	**41**	**104**	**114**	**100**	**3,40**	**3,50**	**3,80**	**2**	**5**	**13**	**13,94**		**12,00**	**3,3**	**5,9**	**6,6**	**69**

Рисунок 9 – Неполная таблица начальных данных

Первый столбец этой таблицы – порядковые номера пациентов (всего их 92 в этой таблице). Последующие 24 столбца – это признаки. Рассматриваются 8 типов признаков, по три признака того же типа на каждый из трех моментов времени – до лечения, после лечения и через месяц после лечения. Все пациенты в таблице делятся на три группы:

- группа контроля;
- группа, проходившая свое лечение с мексидолом;
- группа, проходившая свое лечение без мексидола;

Нужно отметить, что мы располагаем небольшим числом пациентов, что может нехорошо сказываться на обучении нейронной сети ввиду маленькой

входной даты. Также мы оперируем небольшим числом групп. Группа контроля также участвует в обучении нейронной сети, поэтому мы имеем всего три группы для исследования задачи. Еще одним существенным недостатком является отсутствие некоторых столбцов в таблице. Предполагается, что значения соответствующих признаков в пропущенных столбцах не меняются с течением времени, поэтому пробелы заполняются теми же значениями, что и в момент времени «до лечения».

Ниже представлены типы признаков и виды рассматриваемых групп:

A – группы:
0 – контроль (1-30)
1 – без мексидола (31-61)
2 – с мексидолом (62-92)
B – № протокола
C – Hb до лечения (г/л)
D – Hb после лечения (г/ л)
E – Hb ч\з мес лечения (г/ л)
F – эритроциты до лечения (х1012 л)
G – эритроциты после лечения (х1012 л)
H – эритроциты ч\з мес лечения (х1012 л)
I – ретикулоциты до лечения (%0)
J – ретикулоциты после лечения (%0)
K – ретикулоциты ч\з мес лечения (%0)
L – лейкоциты до лечения (х109 л)
M – лейкоциты после лечения (х109 л)
N – лейкоциты ч\з мес лечения (х109 л)
O – сыв. Fe до лечения (мкм/л)
P – сыв. Fe после лечения (мкм/л)
Q – сыв. Fe ч\з мес лечения (мкм/л)
R – ОЖСС до лечения (мкм/л)
S – ОЖСС после лечения (мкм/л)
T – ОЖСС ч\з мес лечения (мкм/л)
U – трансферрин до лечения (мг/дл)
V – трансферрин после лечения (мг/дл)
W – трансферрин ч\з мес лечения (мг/дл)
X – sTfR до лечения (рецепторы к трансферрину) (мкг/мл)
Y – sTfR после лечения (мкг/мл)
Z – sTfR ч\з мес лечения (мкг/мл)

В группе контроля:
N – лейкоциты ч\з мес
Q – сыв. Fe ч\з мес
T – ОЖСС ч\з мес
W – трансферрин ч\з мес
Z – sTfR ч\з мес

4.1 Рассчитанные значения интегральных показателей

Для каждой группы лечения построены интегральные показатели.

Значения интегрального показателя на основе информационной меры Кульбака представлен на рисунке ниже:

	Группа 0			Группа 1			Группа 2	
до лечения	после лечения	ч/з месяц после	до лечения	после лечения	ч/з месяц после	до лечения	после лечения	ч/з месяц после
0,68	0,44	0,62	0,34	0,80	0,46	0,28	0,93	0,75
1,06	1,05	1,12	0,49	1,46	0,32	0,18	0,61	1,04
1,07	0,75	1,03	1,23	1,31	0,62	1,15	1,55	0,43
0,42	0,55	0,46	0,95	0,66	0,38	0,17	1,21	1,06
1,07	1,22	1,40	0,64	0,75	1,49	1,12	1,42	0,33
1,35	1,56	0,81	0,26	0,97	0,44	0,30	1,18	0,96
0,99	0,93	1,56	0,39	0,35	1,10	0,48	0,67	1,50
0,88	1,05	0,84	0,59	0,82	0,38	0,95	0,95	0,69
0,91	0,85	0,59	0,61	0,48	0,40	0,26	0,48	0,78
1,48	1,74	0,99	0,44	1,10	1,19	0,52	1,01	0,63
1,20	1,33	1,61	0,25	0,69	0,32	0,56	0,90	0,86
1,13	0,89	1,37	0,59	0,59	0,45	0,38	0,85	1,47
0,95	0,83	1,45	0,17	0,52	0,43	0,36	0,41	0,90
0,72	0,74	1,66	0,19	0,59	0,25	0,07	0,60	1,66
1,40	1,50	1,09	0,24	1,18	0,39	0,18	0,52	0,95
0,91	1,07	0,70	0,27	0,92	0,25	0,45	0,51	0,42
0,66	0,58	0,59	0,11	0,33	0,32	1,07	1,54	1,12
1,32	1,25	0,39	0,11	0,39	0,38	0,28	0,82	0,60
1,43	1,60	1,17	0,16	0,69	1,08	0,24	0,56	0,72
1,47	1,42	1,33	0,48	1,29	0,42	0,99	1,86	0,84
1,10	1,14	1,63	0,26	0,52	0,67	0,52	0,61	1,07
0,44	0,57	1,12	0,28	0,43	0,77	0,32	0,94	0,54
0,67	0,43	0,72	0,62	0,79	0,40	0,20	1,66	0,76
0,38	0,37	1,34	0,13	0,87	0,90	0,88	1,27	1,27
0,52	0,59	1,17	0,24	0,56	0,48	0,45	0,86	0,60
1,09	1,20	0,93	0,29	0,07	0,46	0,41	0,47	0,68
0,79	0,95	0,50	0,23	0,45	0,62	0,95	0,87	0,57
1,58	1,58	0,71	0,22	0,27	0,96	0,14	1,13	0,40
1,39	1,56	1,48	0,38	1,46	0,88	0,41	0,38	1,19
0,97	0,90	0,88	0,40	0,55	1,02	0,52	0,72	0,91

Рисунок 10 – Интегральные показатели на основе меры Кульбака

В формуле для вычисления данного критерия $P_j = 2\Phi\left(\frac{\delta}{\sigma}\right) - 1$, в числителе аргумента функции Лапласа величину δ среднего значения признака $x_{\text{норм}}$ можно высчитывать двумя способами:

1. Как среднее между максимальным и минимальным возможными значениями признака;
2. Как среднее значение среди всех пациентов рассматриваемой группы по соответствующему признаку.

Референсные значения лабораторных показателей беременных женщин представлены в виде таблицы:

Таблица 1 – Показатели крови

Показатель	**Референсные значения**
Общий анализ крови	**женщины**
Гемоглобин (HGB), g/dl	110-150
Эритроциты (RBC) x 10^6 /mm^3	3,5-6,1
Гематокрит (HCT), %	33 - 52
Ретикулоциты %	0,59 – 2,07
Лейкоциты (WBC) x 10^3 /mm^3	4,0 – 12,0
ОЖСС, мкмоль\л	50-85
Трансферрин, мг/дл	200-360
Рецепторы к трансферрину мг\л	1,9 – 4.4
Сывороточное железо, мкмоль\л	12-27

Представленные на рисунке интегральные показатели рассчитаны по пункту 2.

Далее в работе будут рассматриваться именно данные интегральных показателей, полученные с использованием информационной меры Кульбака.

В качестве эталонного состояния предлагается рассмотреть интегральную кривую, полученную из известных референсных значений показателей крови. Таким образом, для каждого показателя берется среднее между максимальным и минимальным значением соответствующего показателя крови.

Мы располагаем данными интегральных показателей только в трех точках: до лечения, после лечения и через месяц после лечения. По этим трем

точка строится сплайн-функция $I(t)$, которая обладает следующими условиями:[6]

1. $I(t) \in C^2[t_0, t_2]$ – два раза непрерывно дифференцируема;
2. $I(t_i) = I_{\text{адапт}_i}\ \forall\ i = 1, 2, 3$ – совпадает с узлами интерполяции
3. $I(t)$ является многочленом третьей степени на каждом подинтервале.

4.2 Модель нейронной сети

Наша цель – спрогнозировать значение интегрального показателя в смоделированные моменты времени $(t_1, t_2, t_3, t_4, t_5, t_6)$ на основе известных показателей крови в нулевой момент времени. С этой целью для каждой группы лечения строиться своя модель нейронной сети.[6] Каждая модель нейронной сети представляет собой определенное управляющее воздействие.

Построим модель нейронной сети для второй группы лечение (с мексидолом). В каждой группе данных 30 человек. Для обучения нейронной сети этого слишком мало. Поэтому следует увеличить выборку в 30 раз. Для этого к этой выборке данных применяется $bootstrap-$ преобразование.[7] Таким образом, мы уже имеем выборку из 900 примеров, которую нейронная сеть разбивает на три группы – обучающую, которая состоит из 630 примеров, валидационную (135 примеров) и тестовую (135 примеров).

Ниже представлены результаты обучения нейронной сети в среде Matlab: [8]

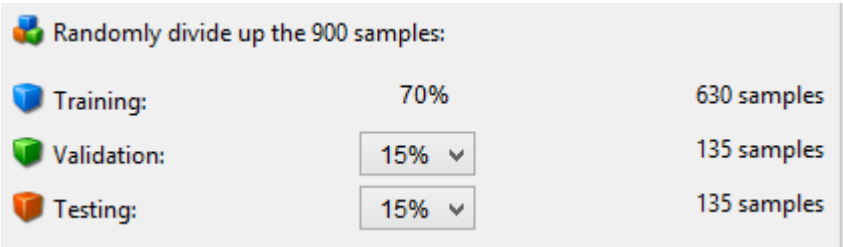


Рисунок 11 – *Размеры тренировочного, валидационного и тесового множеств.*

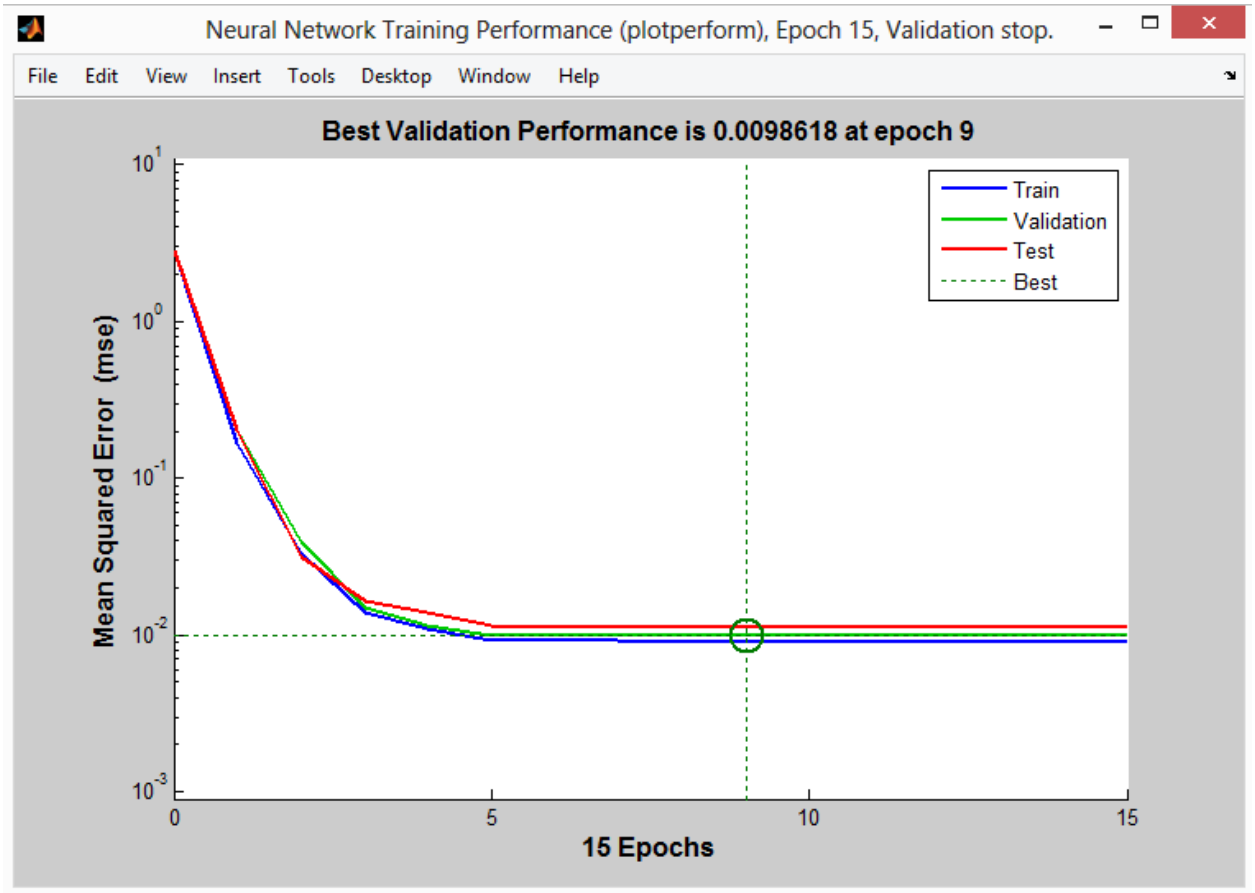


Рисунок 12 – Среднеквадратическая ошибка

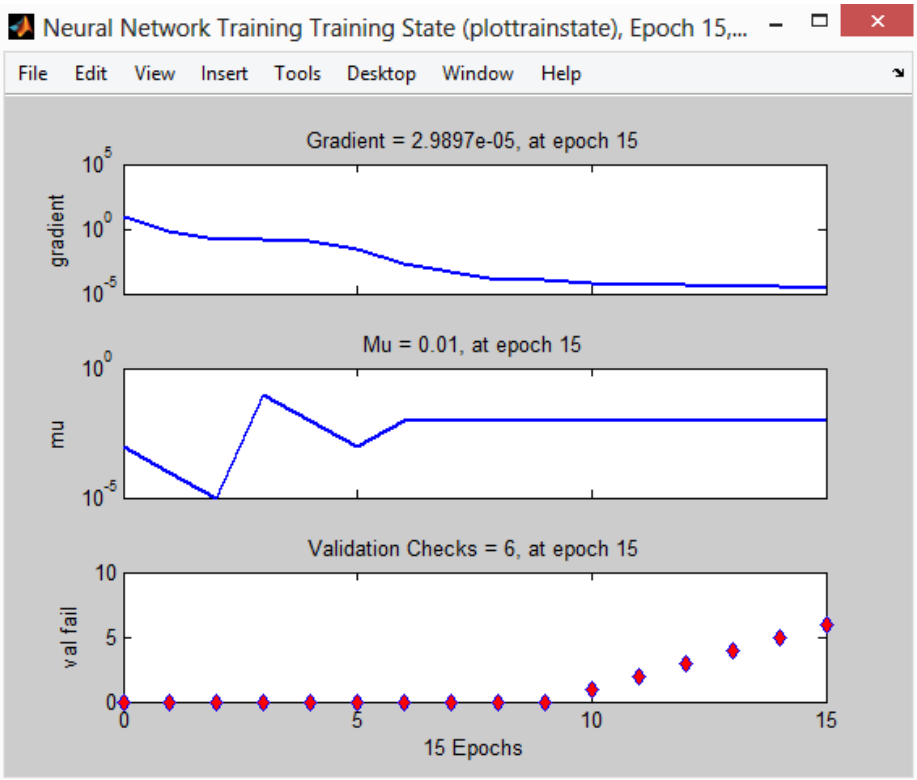


Рисунок 13 – График градиента

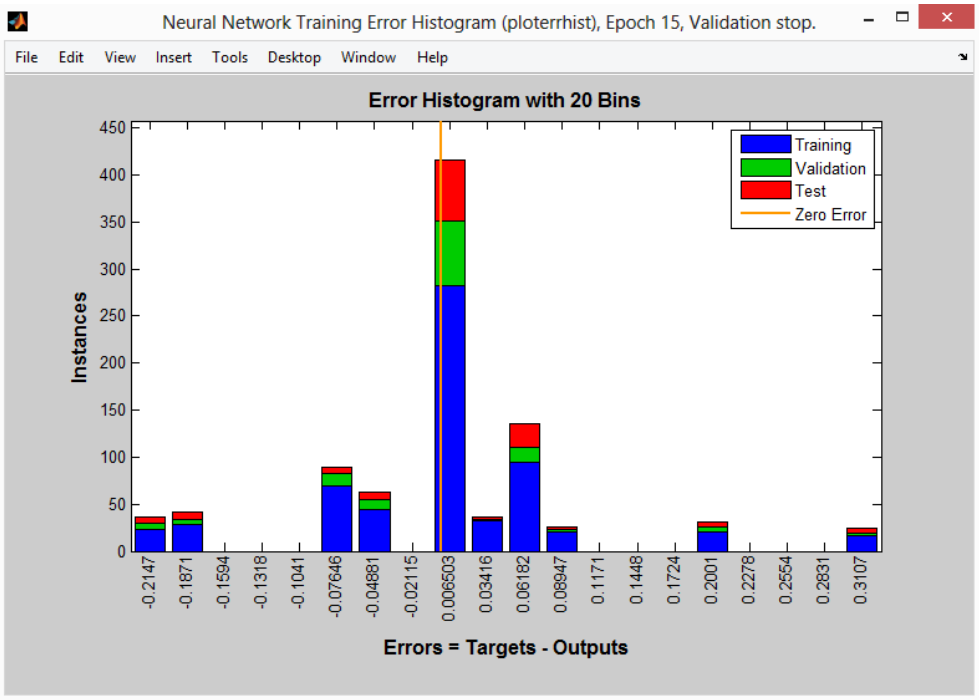


Рисунок 14 – Гистограмма ошибок

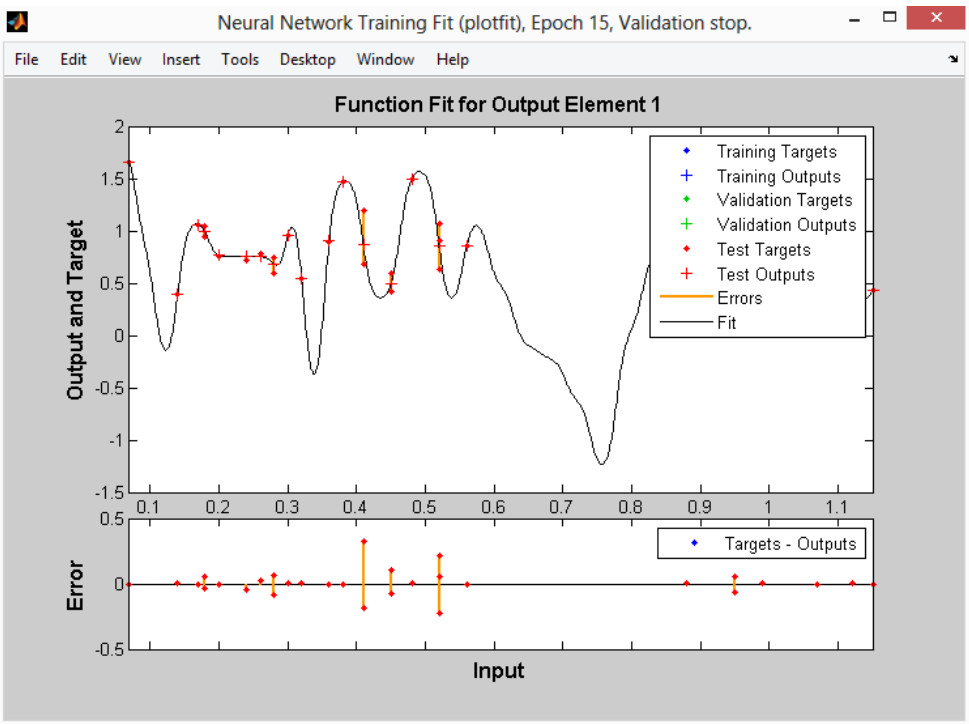


Рисунок 15 – Значения фитнесс-функции

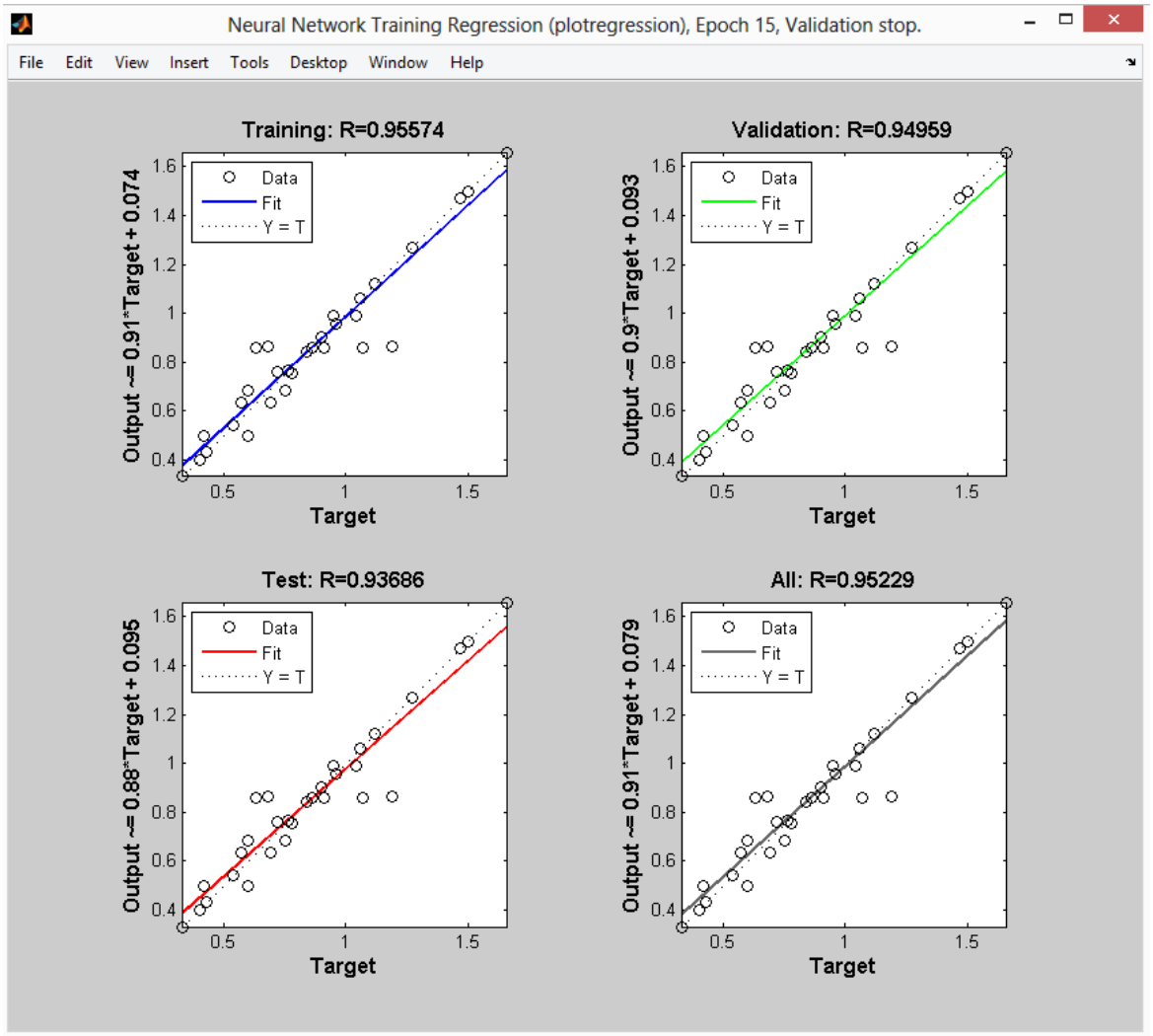


Рисунок 16 – Результаты регрессии.

Таким образом, для каждого из трех управляющих воздействий построена и обучена модель нейронной сети, которая с большой точностью прогнозирует значения интегрального показателя в любой из рассматриваемых моментов времени.

Эти модели будут использоваться для вычисления целевой функции

$$J(x;u) = \frac{1}{k}\sum_{i=t_0}^{t_k}\left(I_{\text{адапт}_i} - I_{\text{кр}_i}\right)^2$$

Где $I_{\text{адапт}_i}$ – прогнозное значение, которое выдает модель u в момент времени i.

4.3 Реализация генетического алгоритма для выбора управляющего воздействия

Имеются три управляющих воздействия:

- Не применять воздействие;
- Лечение с мексидолом;
- Лечение с сарбифером.

Закодируем воздействия следующим образом:

$u_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$ – без воздействия;

$u_1 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$ – с мексидолом;

$u_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ – с сарбифером;

Тогда хромосома (решение) представляется в виде:

$$Chromosome = \begin{array}{|c|c|c|c|c|c|} \hline u_{i1} & u_{i2} & u_{i3} & u_{i4} & u_{i5} & u_{i6} \\ \hline \end{array}$$

Где первый нижний индекс i у u_{ij} представляет собой порядковый номер воздействия, а второй индекс j – момент времени.

Функция приспособляемости (фитнес функция) хромосомы вычисляется следующим образом:

$$Fitness(Chromosome) = \frac{1}{6}\sum_{i=t_0}^{t_6}\left(I_{\text{адапт}_i} - I_{\text{кр}_i}\right)^2$$

Самый частый и трудоемкий процесс в работе генетического алгоритма – это вычисление функции приспособляемости $Fitness(Chromosome)$. Поэтому процесс вычисления этой функции распараллеливается. Тем самым, генетический алгоритм занимает гораздо меньше временя для получения требуемого результата.

При рассмотрении результатов работы ГА варьируются способы селекции и скрещивания особей:

1. Селекция – наилучший + случайно выбранный; Скрещивание – одноточечное.
2. Селекция – наилучший + случайно выбранный; Скрещивание – двухточечное.
3. Селекция – наилучший + случайно выбранный; Скрещивание – $u^l æ_k u^r, \quad k = \frac{F(u^l)}{F(u^l) + F(u^r)}$
4. Селекция – рулеточный отбор; Скрещивание – одноточечное.
5. Селекция – рулеточный отбор; Скрещивание – двухточечное.
6. Селекция – рулеточный отбор; Скрещивание – $u^l æ_k u^r, \quad k = \frac{F(u^l)}{F(u^l) + F(u^r)}$

Для нахождения оптимального решения поставленной задачи рассматривается алгоритм, который был приведен в главе выше, хотя есть множество других вариантов генетического алгоритма. Например, вместо оператора инверсии применить оператор локального улучшения, поменять порядок следования оператора инверсии и оператора мутации и т.д.

В качестве критерия остановки задается определенное число итераций. Однако, поскольку перед запуском ГА для исходного примера известно его оптимальное решение (за счет полного перебора), можно задать условие: «продолжать до тех пор, пока рекордное значение популяции не достигнет оптимального решения», но не факт, что алгоритм точно сойдется и процесс не застрянет в цикле $while()$.

На рисунке ниже представлены усредненные по 100 запускам графики зависимостей значений функции приспособляемости от итерации:

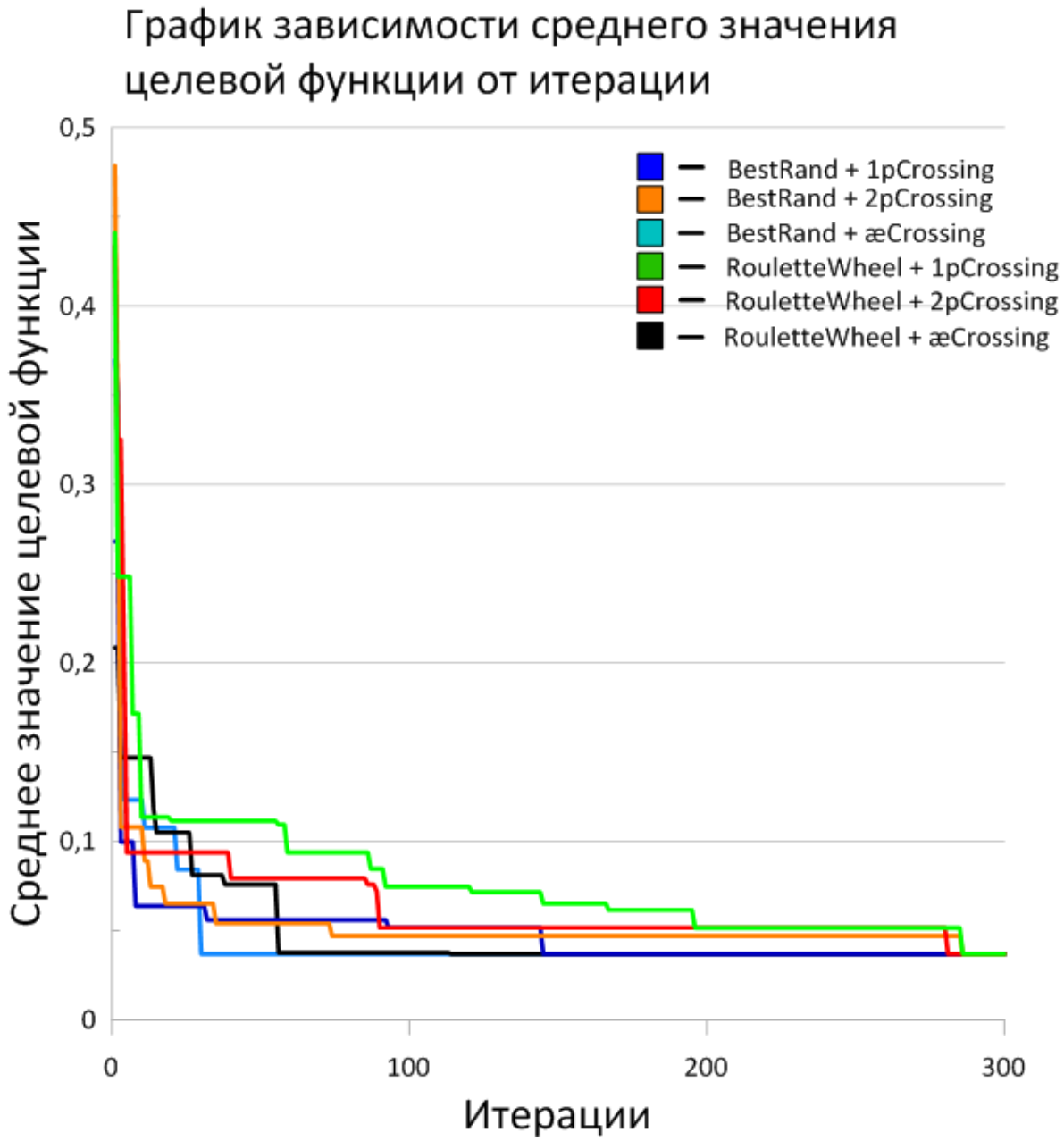


Рис. 15. Усредненный график зависимости целевой функции от итерации.

Используются следующие обозначения:

1. $BestRand + 1pCrossing$ – Селекция – наилучший + случайно выбранный; Скрещивание – одноточечное;
2. $BestRand + 2pCrossing$ – Селекция – наилучший + случайно выбранный; Скрещивание – двухточечное.
3. $BestRand + æCrossing$ – Селекция – наилучший + случайно выбранный; Скрещивание –

$$u^l æ_k u^r, \qquad k = \frac{F(u^l)}{F(u^l) + F(u^r)}$$

4. $Roulette + 1pCrossing -$ Селекция – рулеточный отбор; Скрещивание – одноточечное
5. $Roulette + 2pCrossing -$ Селекция – рулеточный отбор; Скрещивание – двухточечное.
6. $Roulette + æCrossing -$ Селекция – рулеточный отбор; Скрещивание–

$$u^l æ_k u^r, \qquad k = \frac{F(u^l)}{F(u^l) + F(u^r)}$$

Из рисунка выше можно заключить, что все рассматриваемые способы реализации ГА сходятся к оптимальному решению задачи. Также можно сделать вывод, что комбинация: 3) Селекция − наилучший + случайно выбранный; Скрещивание −

$$u^l æ_k u^r, \qquad k = \frac{F(u^l)}{F(u^l) + F(u^r)}$$

сходится быстрее других рассматриваемых комбинаций «селекция-скрещивание».

Ниже приведена таблица результатов работы генетического алгоритма для ста независимых запусков.

Таблица 2 – результаты работы генетического алгоритма для ста независимых запусков

Способ селекции и скрещивания	Достижение оптимума (да/нет)	Средне время работы алгоритма (сек.)	Среднее количество итераций	Размер популяции
BestRand+1pCrossing	да	0.240	146	30
BestRand+2pCrossing	да	0.301	290	25
BestRand+æCrossing	да	0.172	30	20
Roulette+1pCrossing	да	0.280	290	25
Roulette+2pCrossing	да	0.289	284	30

Roulette+æCrossing	да	0.196	55	30

Из таблицы видно, что реализация генетического алгоритма с комбинацией «$BestRand + æCrossing$» требует меньшее число итераций для достижения оптимума и занимает меньше времени. Таким образом, способ реализации ГА с парой «$BestRand + æCrossing$» является самым предпочтительным среди рассматриваемых подходов для решения данной задачи генетическим алгоритмом.

4.4 Реализация алгоритма симуляции отжига (по Коши, Больцману и Быстрый)

Больцмановский отжиг характеризуется следующим законом изменения температуры:

$$T(k) = \frac{T_{max}}{1 + \ln(k)}$$

и способом выбора нового решения: [10]

$$g_s(u) = u + t_k * N(0;1)$$

где $N(0;1) -$ стандартное нормальное распределение;

$t_k -$ значение температуры на $k -$ й итерации.

Нормальное распределение моделировалось следующим образом:

Реализовывались равномерно распределенные случайные величины γ_i на отрезке $[0;1]$. Как известно, для равномерно распределенной случайной величины γ_i математическое ожидание равно нулю, а дисперсия $-\frac{1}{12}$. Далее, воспользовавшись центральной предельной теоремой, можно получить следующее приближение: [11]

$$\frac{\sum_{i=1}^{n} \gamma_i}{\sqrt{\frac{n}{12}}} \xrightarrow[n\to\infty]{} N(0;1)$$

Для $n = 24$ это приближение получается достаточно хорошим.

Ниже приведены результаты моделирования Больцмановского отжига для начальной температуры $t_{max} = 100$ и конечной $t_{min} = 0{,}1$.

График зависимости температуры Больцмановского отжига от итерации:

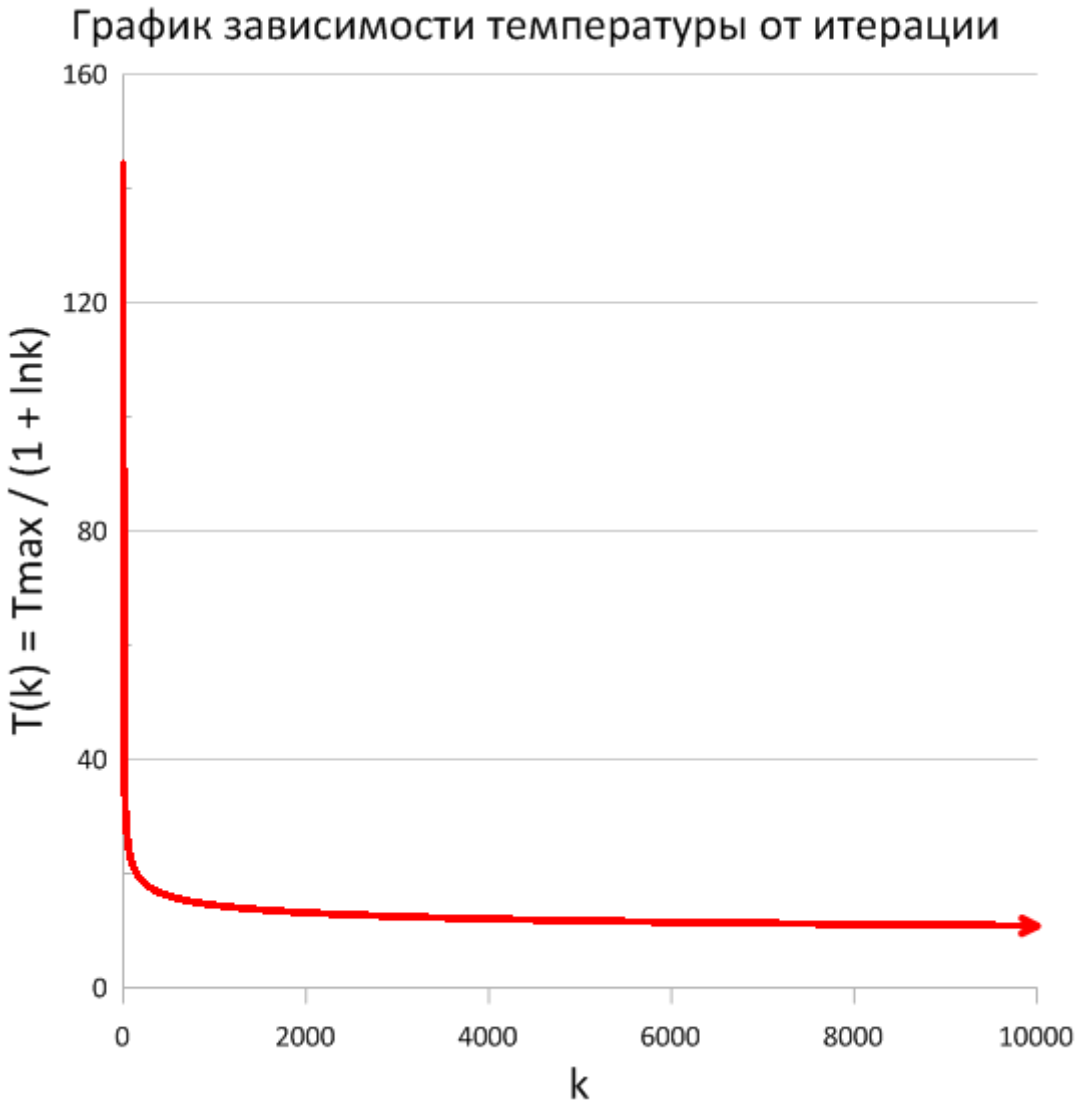


Рисунок 17 – График изменения температуры Больцмановского отжига.

Как видно, $T(k)$ является убывающей функцией. Однако температура убывает очень медленно и не достигает нуля. Это чревато тем, что алгоритм на каждой итерации будет иметь высокую вероятность принятия худшего решения и, как следствие, постоянно возвращаться назад на плохие решения. Таким образом, при такой скорости убывания температуры, алгоритм может не

сойтись к оптимальному решению задачи за определенное число итераций или потребовать очень много итераций для успешного завершения работы.

График зависимости вероятности принятия худшего решения Больцмановского отжига от итерации:

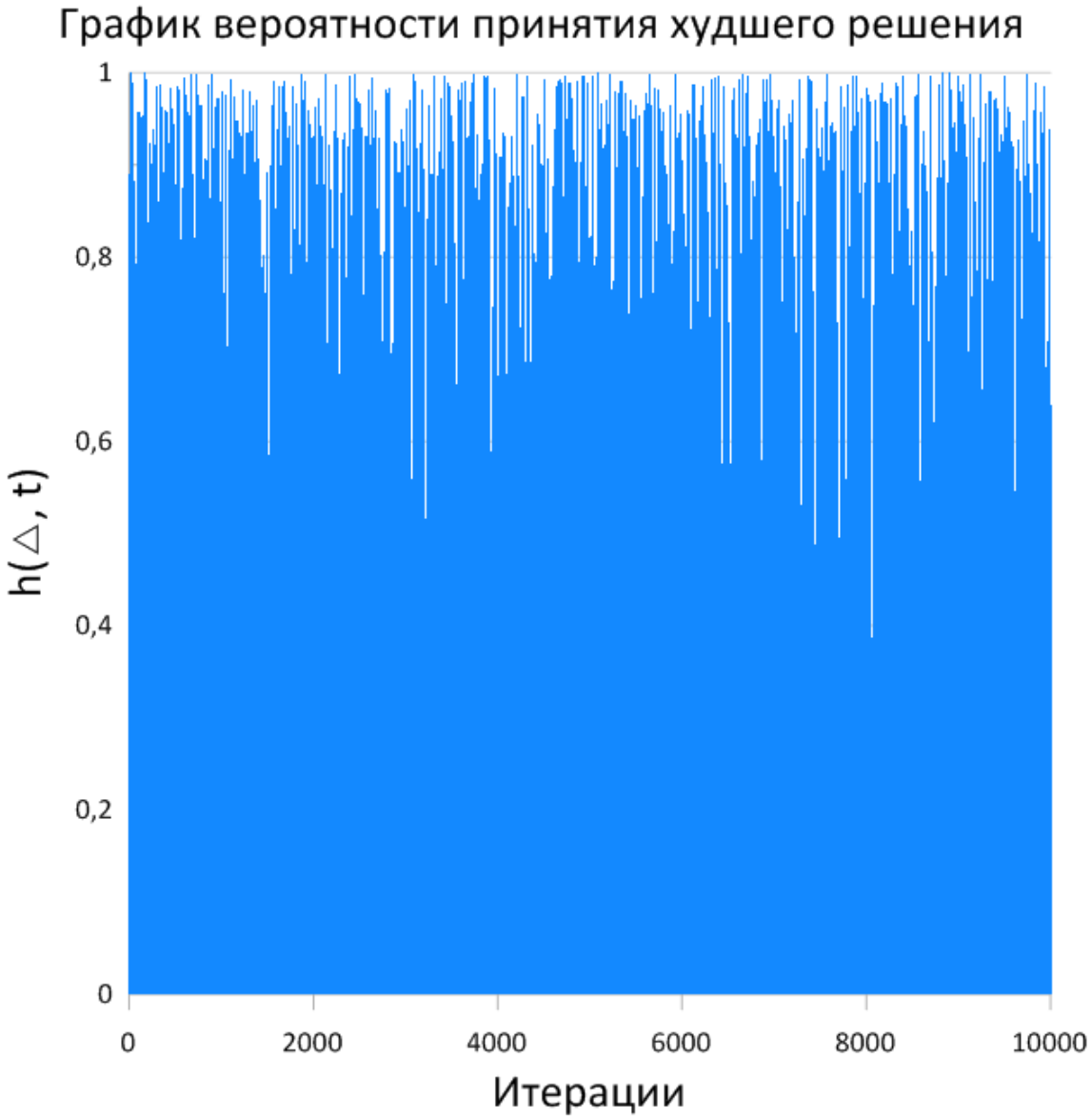


Рисунок 18 – График вероятности принятия худшего решения для Больцмановского отжига

График зависимости целевой функции Больцмановского отжига от итерации:

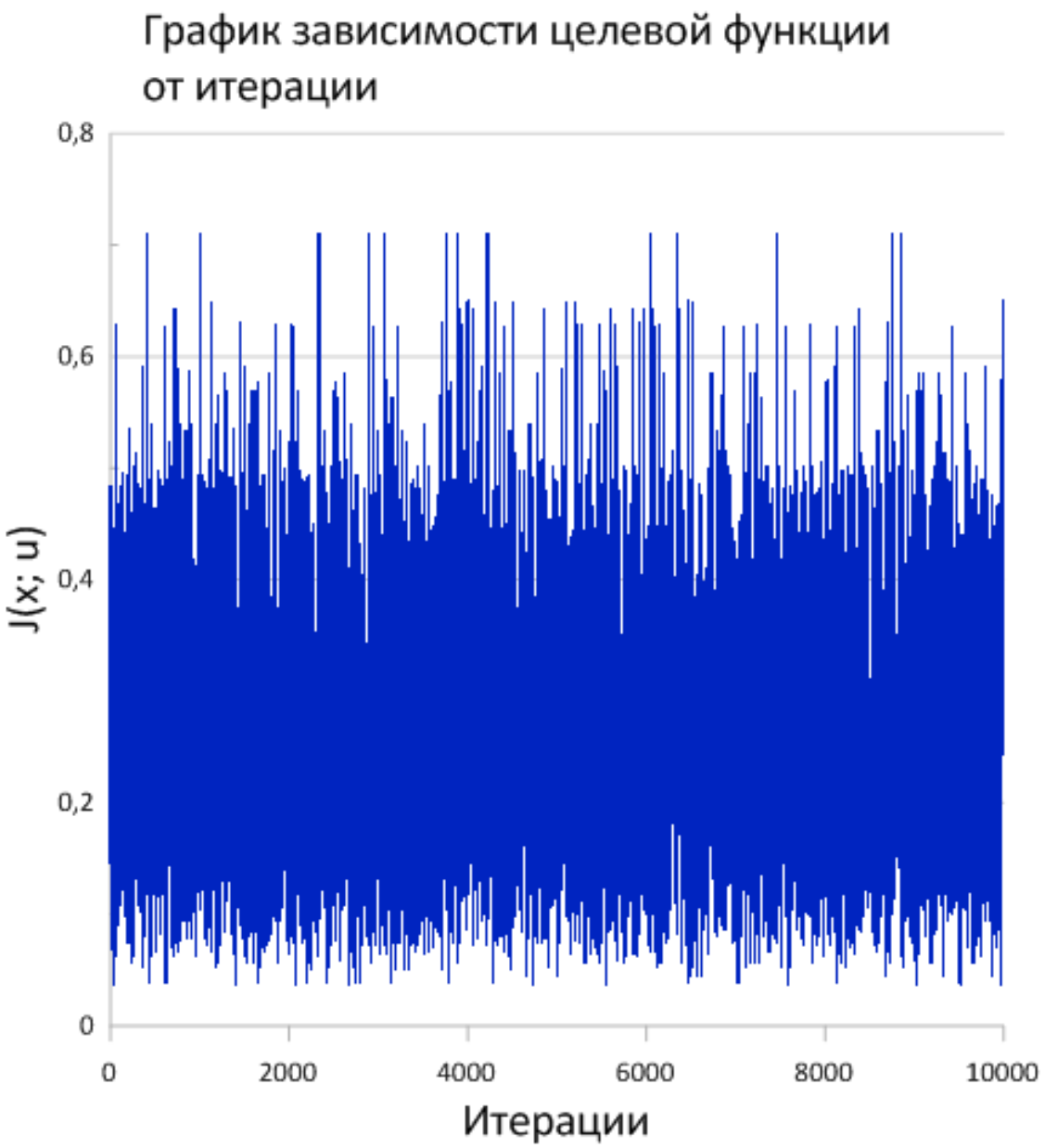


Рисунок 19 – График зависимости целевой функции от итерации для Больцмановского отжига

Как видно из графика, алгоритм Больцмановского отжига с начальной температурой $t_{max} = 100$ и конечной температурой $t_{min} = 0{,}1$ не сошелся к решению задачи из-за низкой скорости убывания температуры.

Быстрый$(fast - annealing)$ отжиг характеризуется следующим законом изменения температуры:

$$T(k) = T_{max} e^{-Ck}$$

где константа $C > 0$.

Для быстрого отжига функция $g_s(u)$ моделировалась следующим образом:

Случайным образом выбирался параметр $s = \{0,1,2,3,4,5\}$, принимающий значения от 1 до 5. Далее компонента u_s решения $u = [u_0, \dots, u_5]$ заменялась на другую, отличную от нее. Таким образом, $g_s(u)$ осуществляет переход от решения u на близкое к нему решение $\tilde{u}$:

$$g_s(u) = \tilde{u}$$

Ниже приведены результаты моделирования быстрого отжига с начальной температурой $t_{max} = 10000$ и конечной температурой $t_{min} = 0{,}001$.

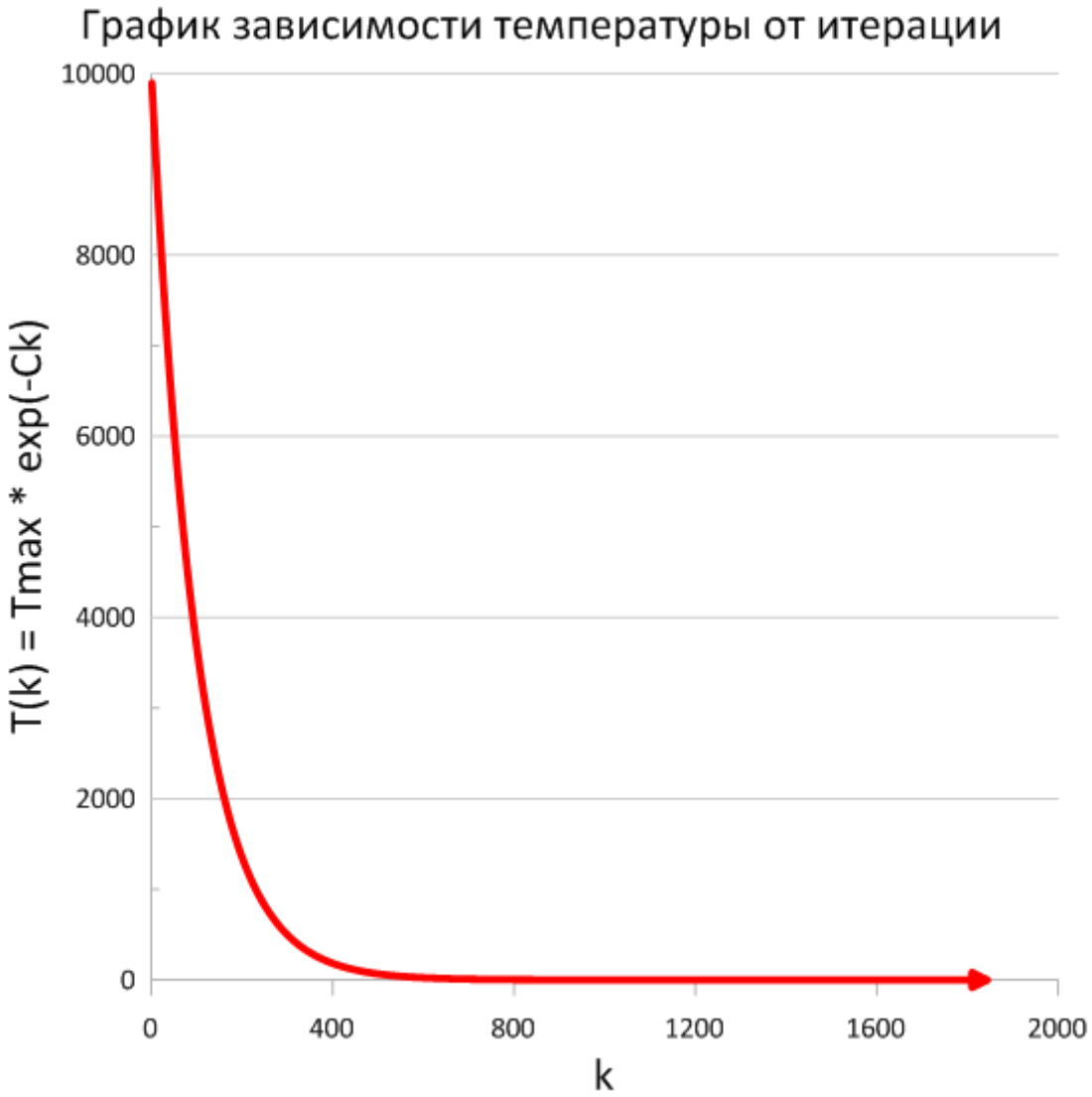


Рисунок 20 – График изменения температуры экспоненциального отжига.

$T(k)$ является убывающей функцией. В отличае от температуры Больцмановского отжига, температура быстрого отжига достигает нуля уже после 400 – й итерации. Также у алгоритма есть возможность «побегать» по решениям из множества допустимых решений.

График зависимости вероятности принятия худшего решения быстрого отжига от итерации:

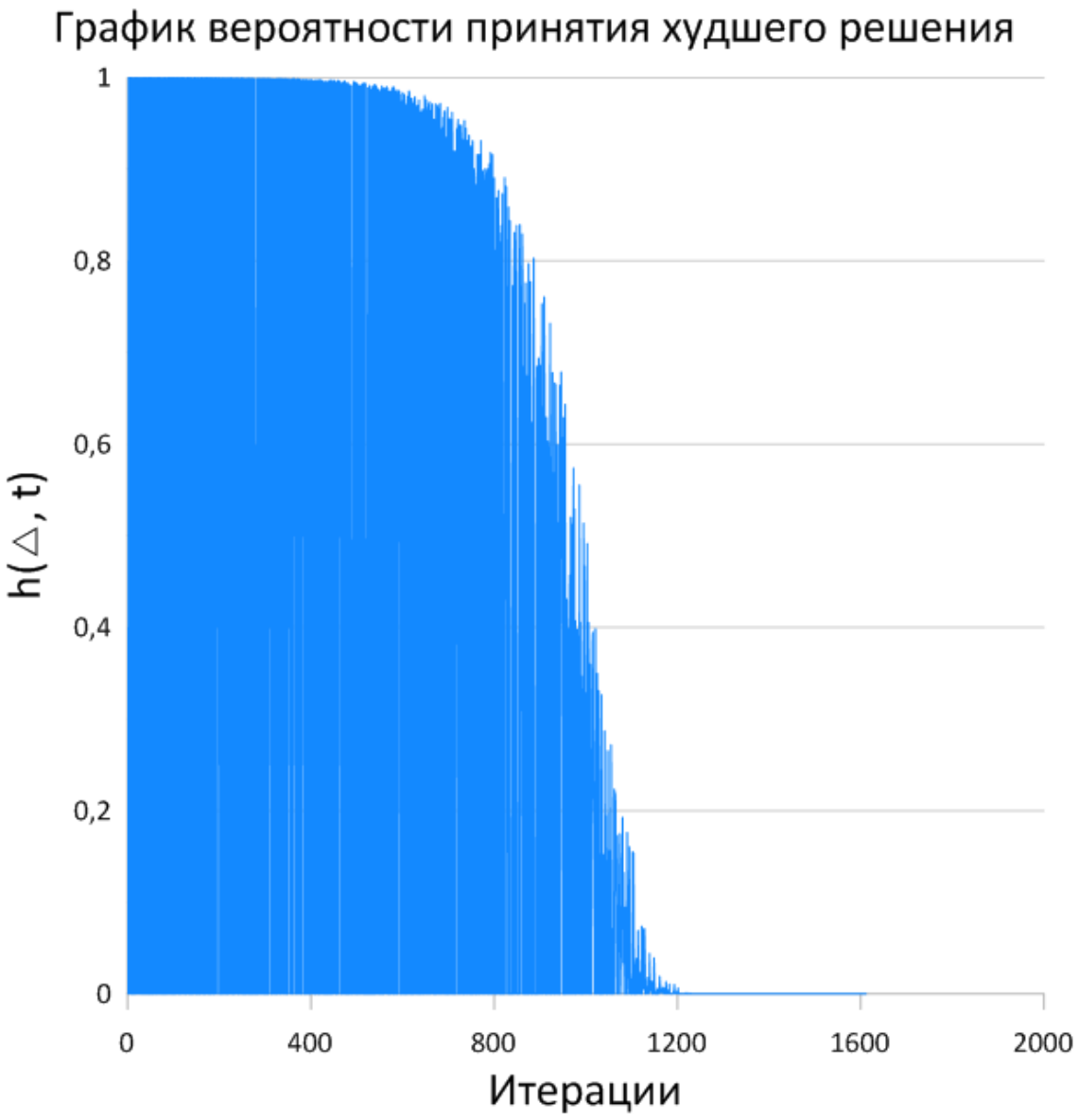


Рисунок 21 – График вероятности принятия худшего решения для экспоненциального отжига

Как видно из графика выше, алгоритм после 1000 – й итерации будет двигаться только в сторону лучшего решения.

График зависимости температуры Быстрого отжига от итерации:

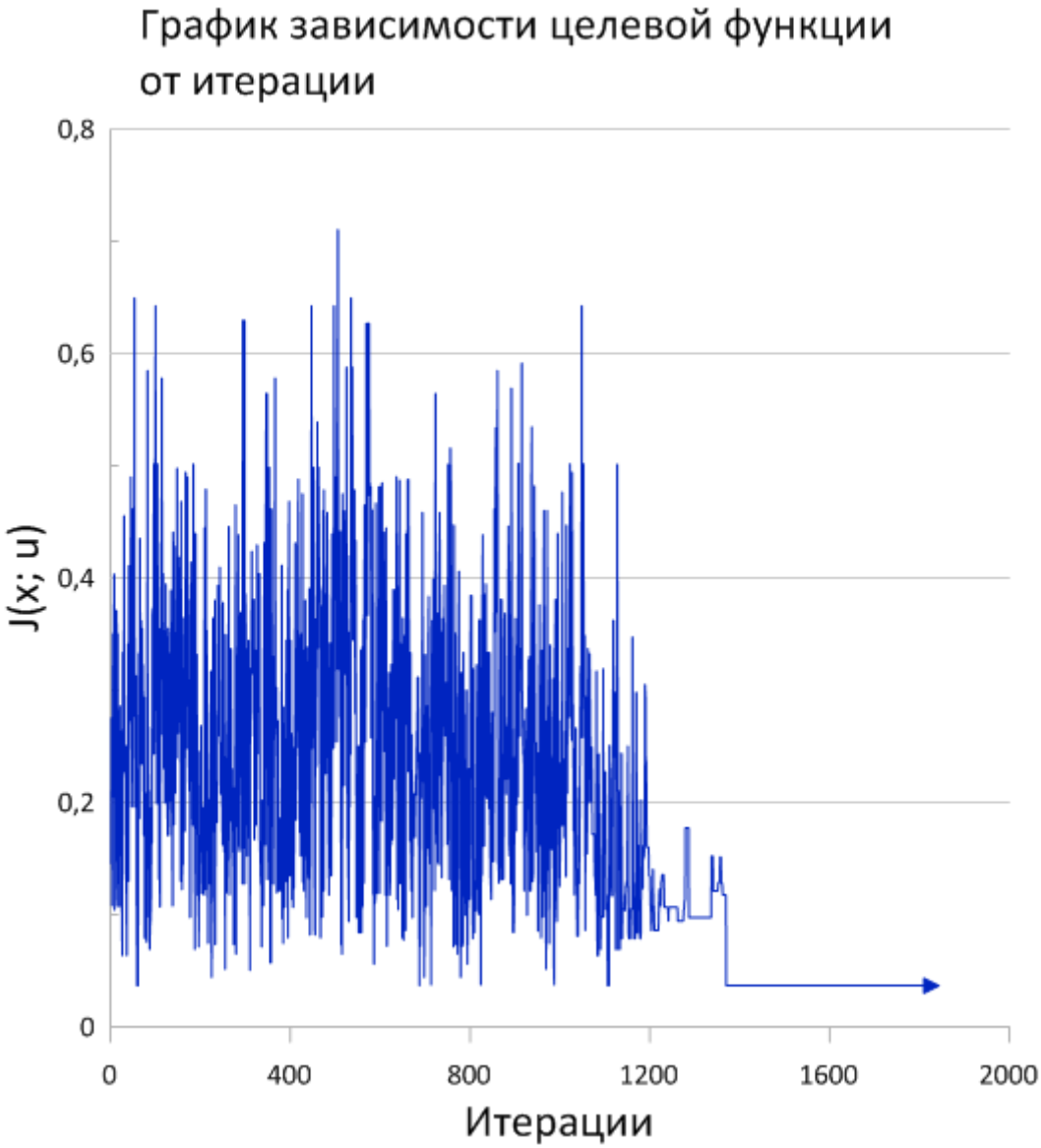


Рисунок 22 – График зависимости целевой функции от итерации для экспоненциального отжига

Видно, что алгоритм быстрого отжига с начальной температурой $t_{max} = 10000$ и конечной температурой $t_{min} = 0{,}001$ сошелся к оптимальному решению задачи.

Отжиг Коши характеризуется следующим законом изменения температуры:

$$T(k) = \frac{T_{max}}{k}$$

Для отжига Коши функция формирования нового решения выглядит следующим образом: [12]

$$g_s(u) = u + t_k * C(0;1)$$

Где C – распределение Коши.

Распределение C моделировалась следующим образом:

Бралась равномерно распределенная случайная величина γ_i на отрезке $[0; 1]$.

Для того чтобы получить требуемое распределение, необходимо вычислить следующую величину: [13]

$$tg\left(\pi\gamma_i - \frac{\pi}{2}\right)$$

Ниже приведены результаты моделирования отжига Коши с начальной температурой $t_{max} = 1000$ и конечной температурой $t_{min} = 0{,}1$.

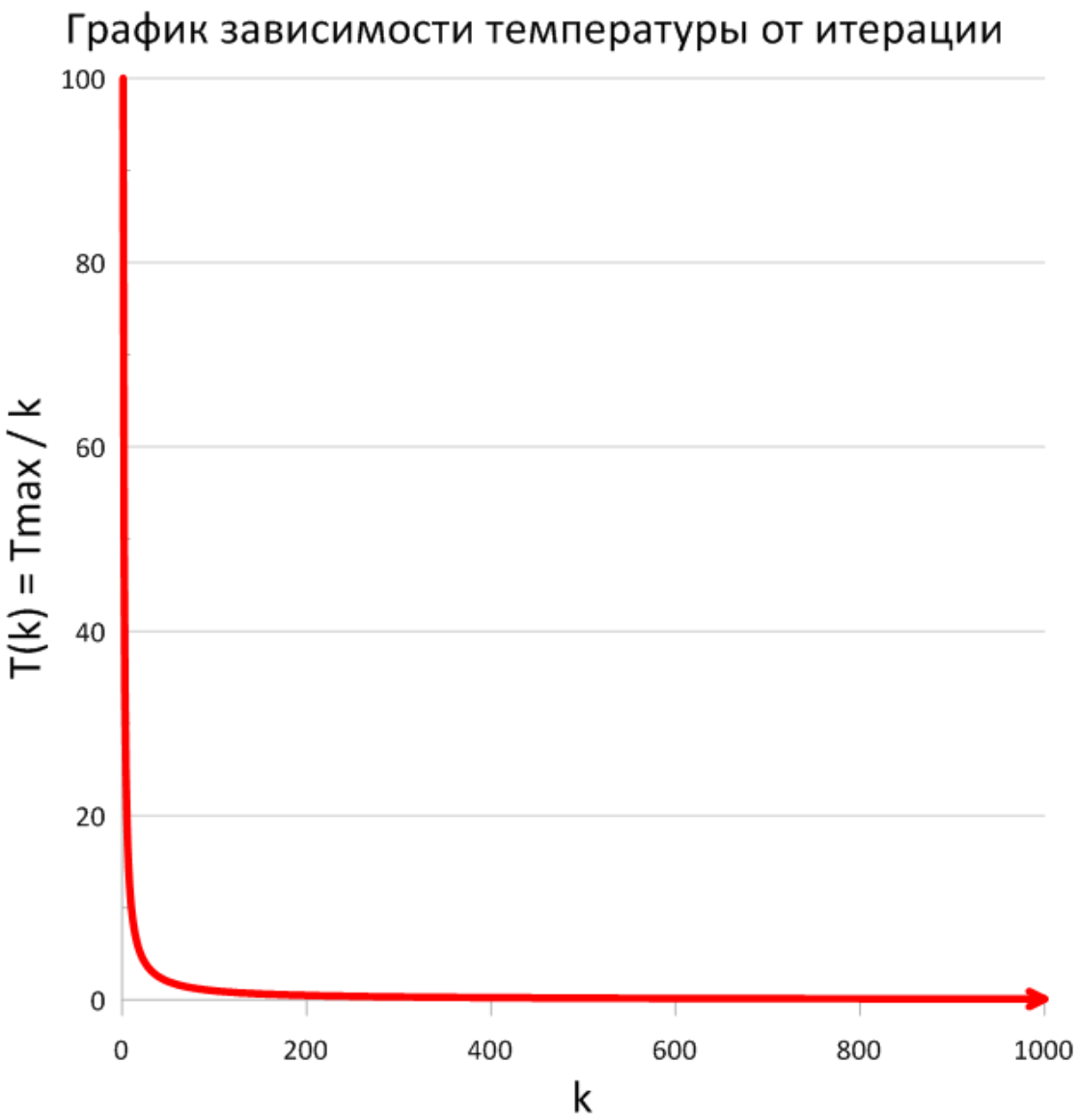


Рисунок 23 – График изменения температуры отжига Коши.

$T(k)$ является убывающей функцией. Как видно, температура убывает быстрее, чем функции температур в предыдущих рассматриваемых случаях и достигает нуля.

График зависимости вероятности принятия худшего решения отжига Коши от итерации:

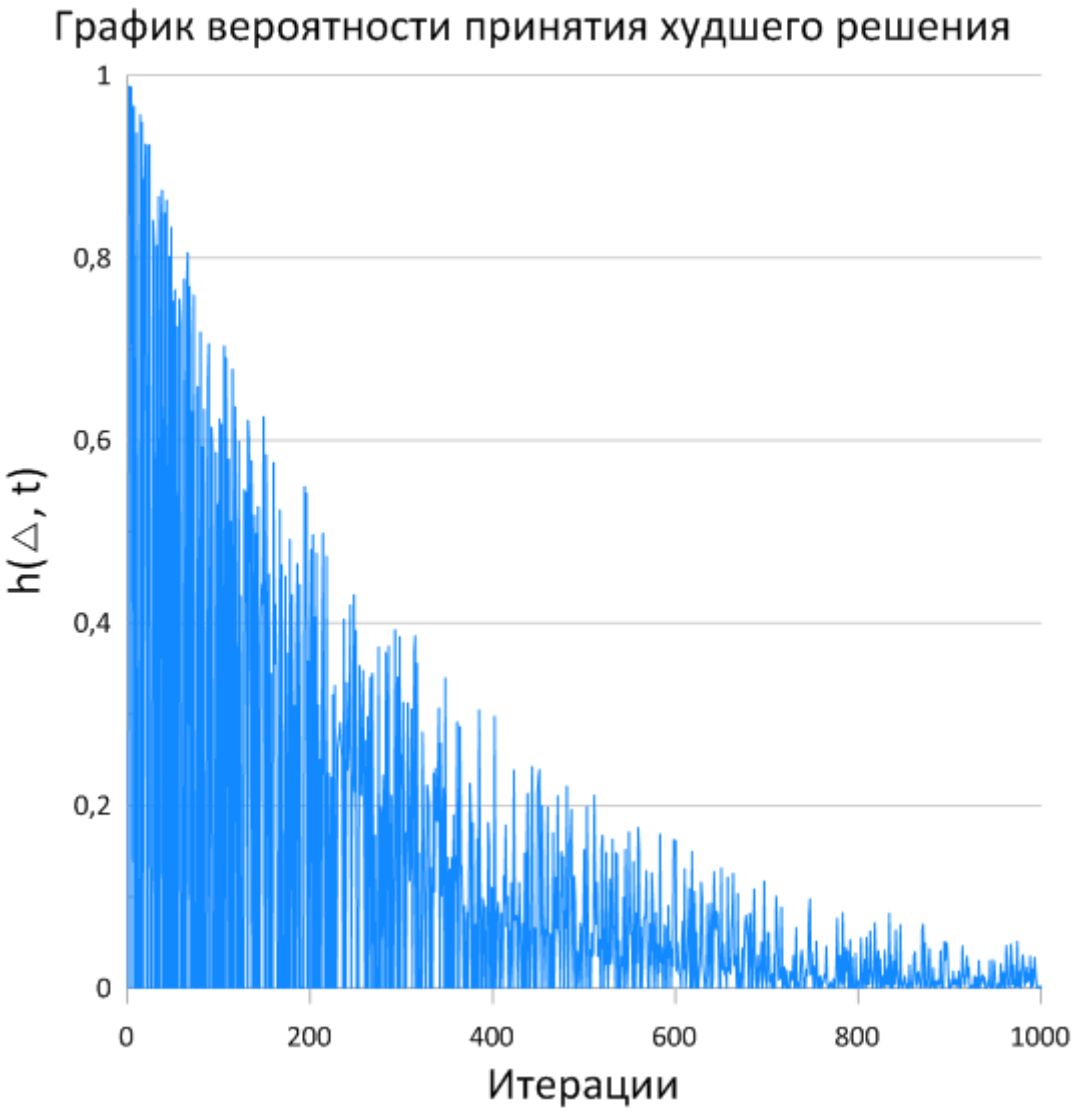


Рисунок 24 – График вероятности принятия худшего решения для отжига Коши

График зависимости целефой функции отжига Коши от итерации:

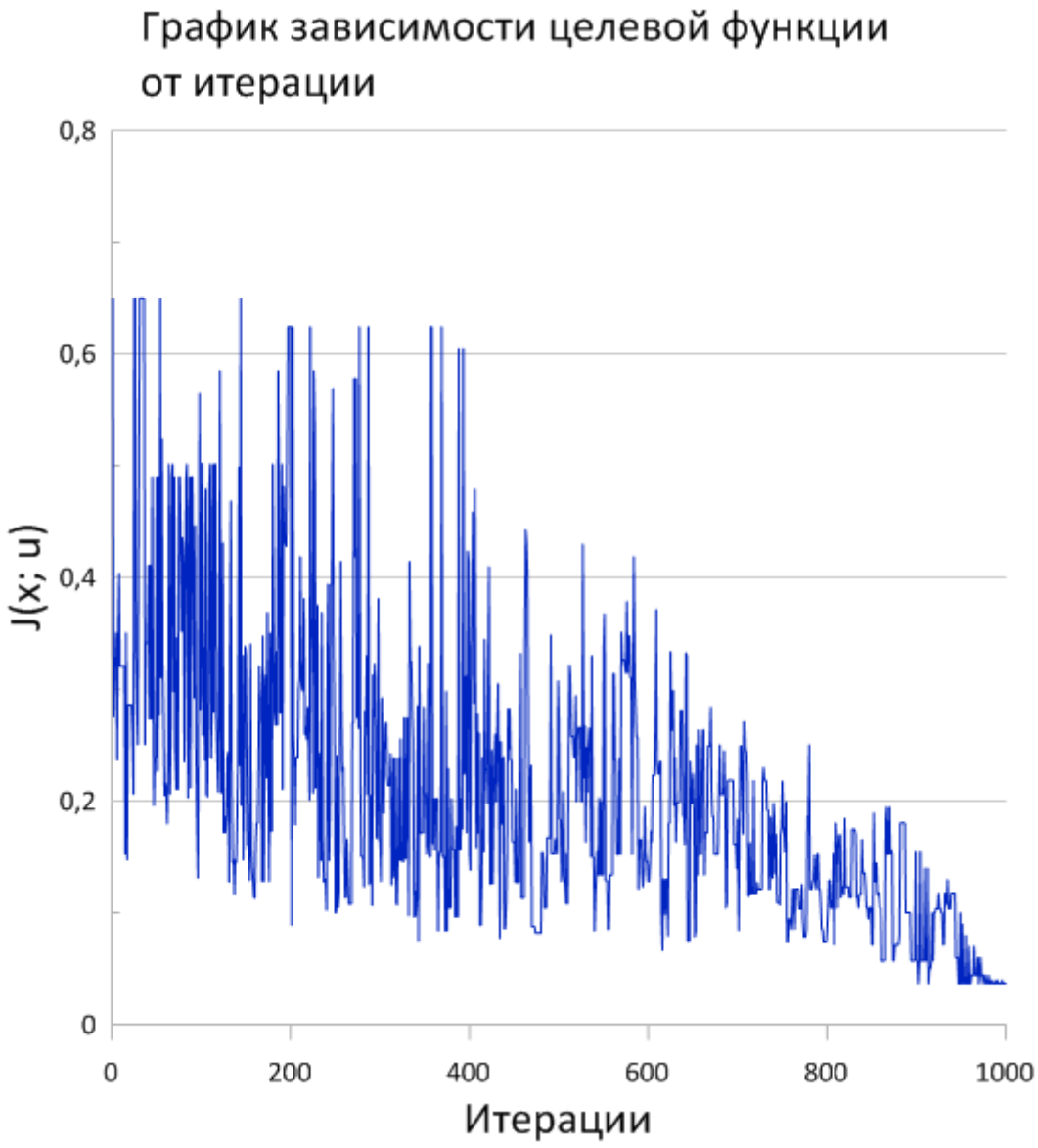


Рисунок 25 – График зависимости целевой функции от итерации для отжига Коши

Видно, отжиг Коши с начальной температурой $t_{max} = 1000$ и конечной температурой $t_{min} = 0{,}1$ сошелся к оптимальному решению задачи быстрее быстрый отжиг. Таким образом, отжиг Коши показал себя лучше чем другие рассматриваемые виды отжигов применительно к решаемой задачи.

5 Статистический анализ

Для статистической обработки данных используется язык R.[19] В качестве исследуемых выборок рассматриваются показатели крови и интегральные критерии.

5.1 Подбор закона распределения

Перед нами стоит следующая задача: мы имеем некоторые наблюдения количественного показателя $x_1, \dots, x_n$ и хотим проверить, принадлежит ли выборка некой теоретической генеральной совокупности с функцией плотности вероятности $f(x, \vec{\theta})$, где $\vec{\theta} -$ вектор параметров, оцениваемых по имеющимся данным.

Подбор закона распределения можно разбить на несколько этапов:

1) предполагаем, что наша выборка принадлежит некоторому известному распределению;
2) оцениваем параметры теоретического распределения;
3) графически оцениваем качество приближения;
4) применяем статистический тест для проверки согласия.

Выясняются законы распределения показателей крови по генеральной совокупности. Из восьми показателей крови выбираются наиболее информативные, а именно:

Лейкоциты – 7;
Трансферин – 8;
Гемоглобин – 9;
Ретикулоциты – 17.

Цифрами обозначены влияния факторов на интегральный показатель. Таким образом, ретикулоциты – самый информативный показатель.

Ниже приведены рисунки сравнения функций и плотностей распределения для исходной выборки и теоретического распределения:

Подбор закона распределения для «Трансферина» до лечения

Оценка параметров для логнормального распределения:

$$Meanlog: 5.97090158\ ;\ \ sdlog: 0.21780312;$$

Критерий Колмогорова-Смирнова: $D = 0.093795, p - value = 0.3932$

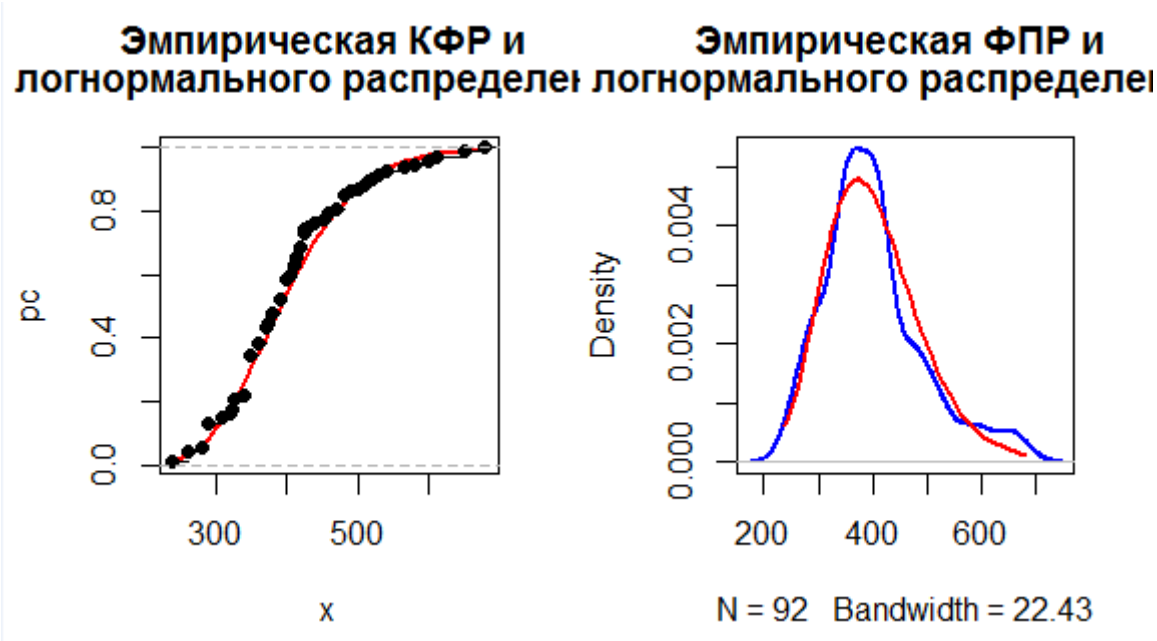


Рисунок 26 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «Трансферина» после лечения

Оценка параметров для нормального распределения:

$$Mean: 296.706522\ ;\ \ sd: 56.752234\ ;$$

Критерий Колмогорова-Смирнова: $D = 0.11225, p - value = 0.1967$

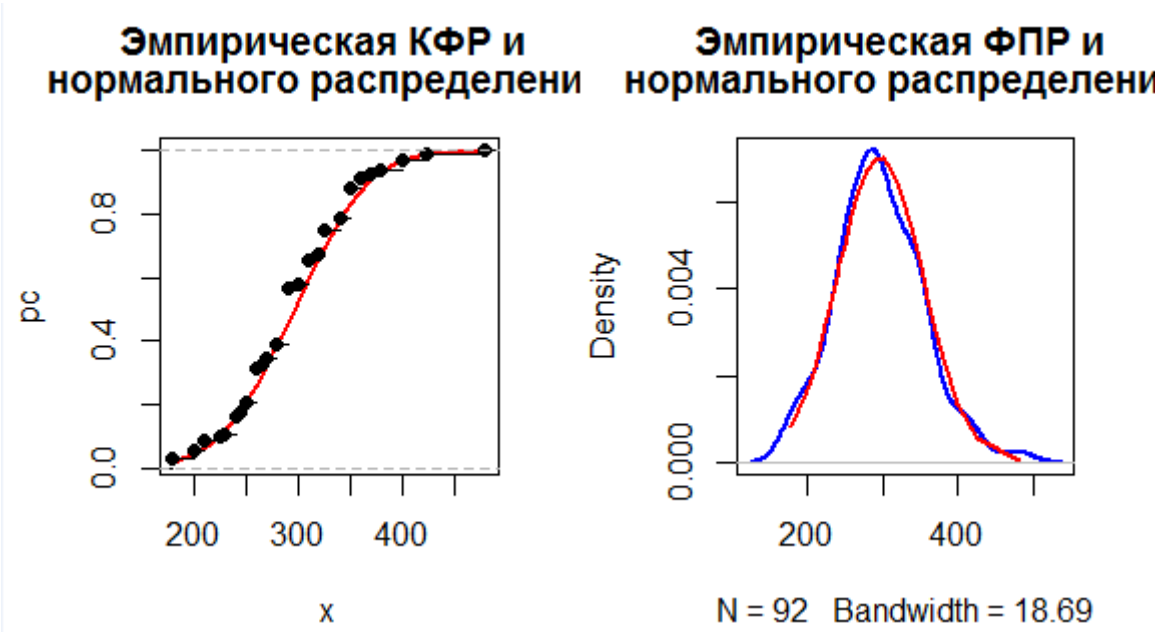


Рисунок 27 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «Трансферина» ч/з месяц лечения

Оценка параметров для нормального распределения:

$$Mean: 289.054348;\ \ sd: 67.045955\ ;$$

Критерий Колмогорова-Смирнова: $D = 0.12481, p - value = 0.1138$

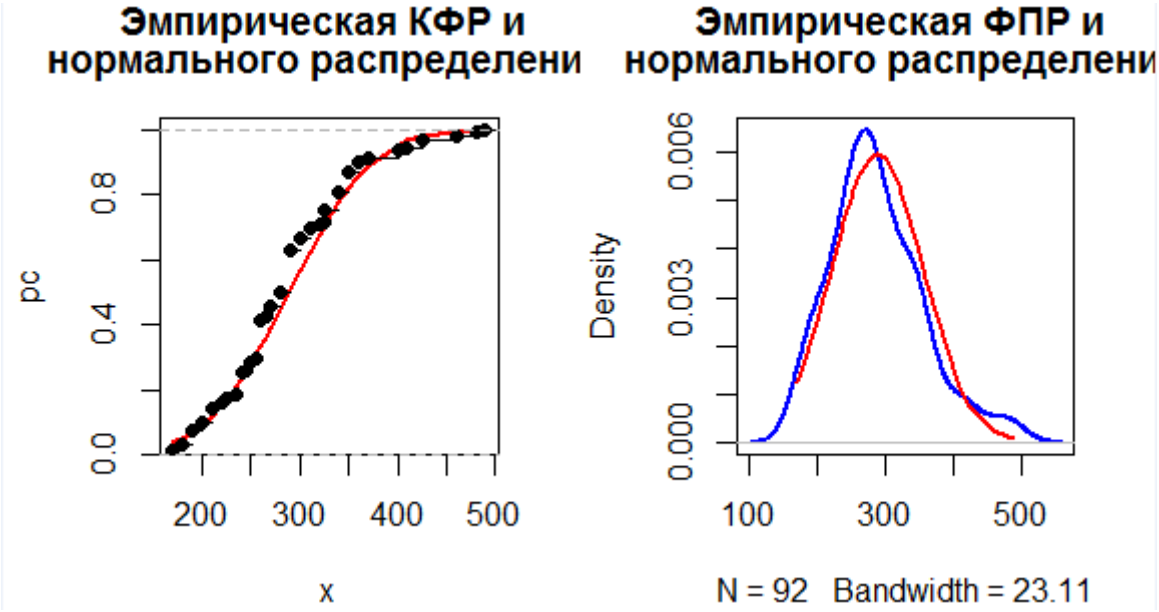


Рисунок 28 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «Лейкоцитов» до лечения

Оценка параметров для нормального распределения:

$$Mean: 9.6613043\ ;\ \ sd: 3.2572061;$$

Критерий Колмогорова-Смирнова: $D = 0.12663, p - value = 0.1046$

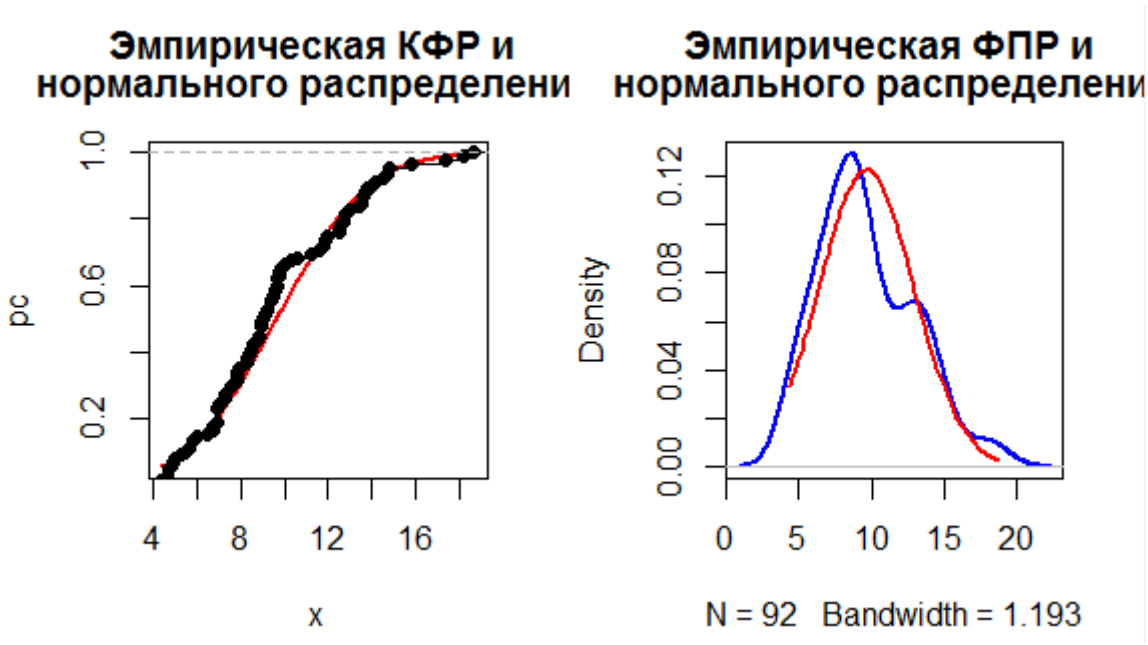


Рисунок 29 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «Лейкоцитов» после лечения

Оценка параметров для нормального распределения:

$Mean$: 9.6613043 ; sd: 3.2572061;

Критерий Колмогорова-Смирнова: $D = 0.12663, p - value = 0.1046$

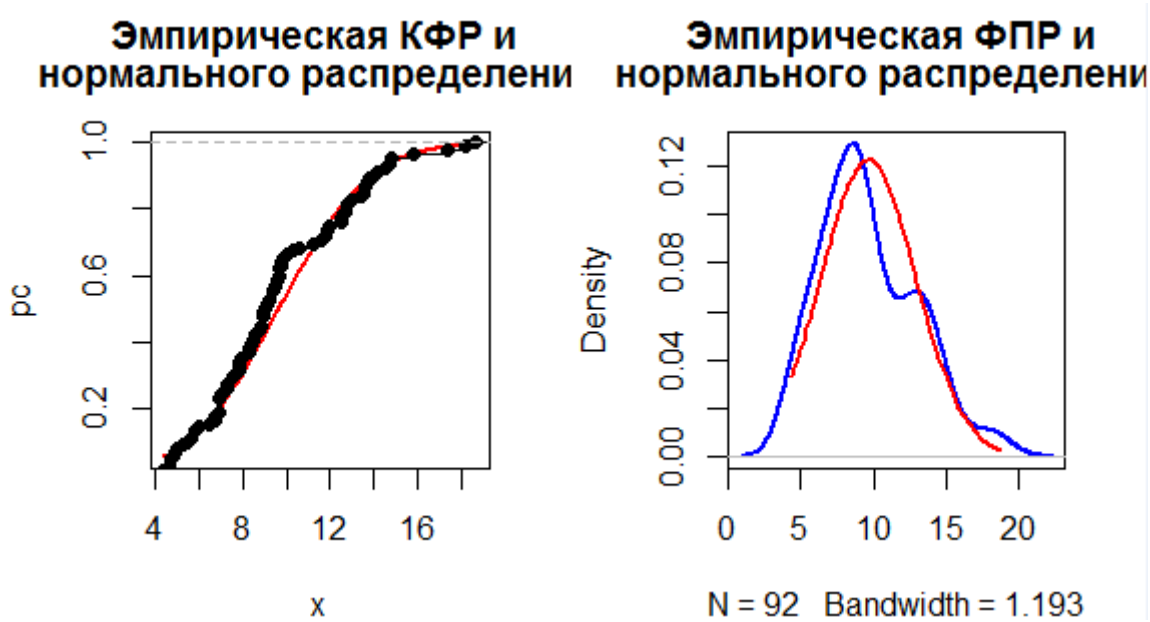


Рисунок 30 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «Лейкоцитов» ч/з месяц лечения

Оценка параметров для нормального распределения:

$Mean$: 11.5121739; sd: 3.3929725;

Критерий Колмогорова-Смирнова: $D = 0.11312, p - value = 0.1897$

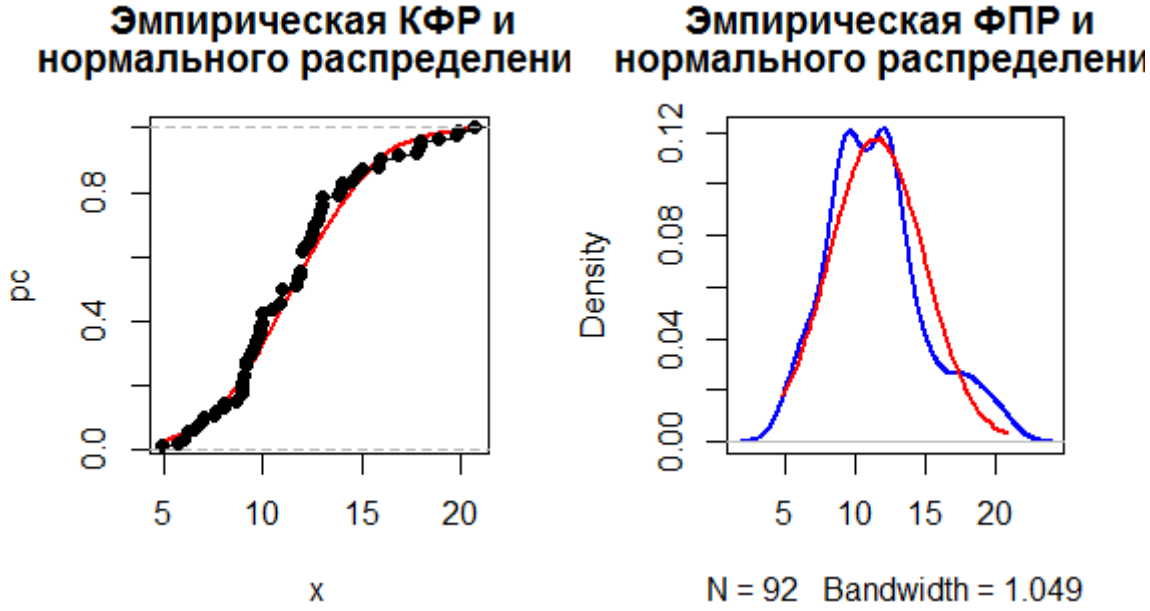


Рисунок 31 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «Ретикулоцитов» до лечения

Оценка параметров для нормального распределения:

$Mean$: 7.4021739; sd: 3.4296019;

Критерий Колмогорова-Смирнова: $D = 0.11414, p - value = 0.1818$

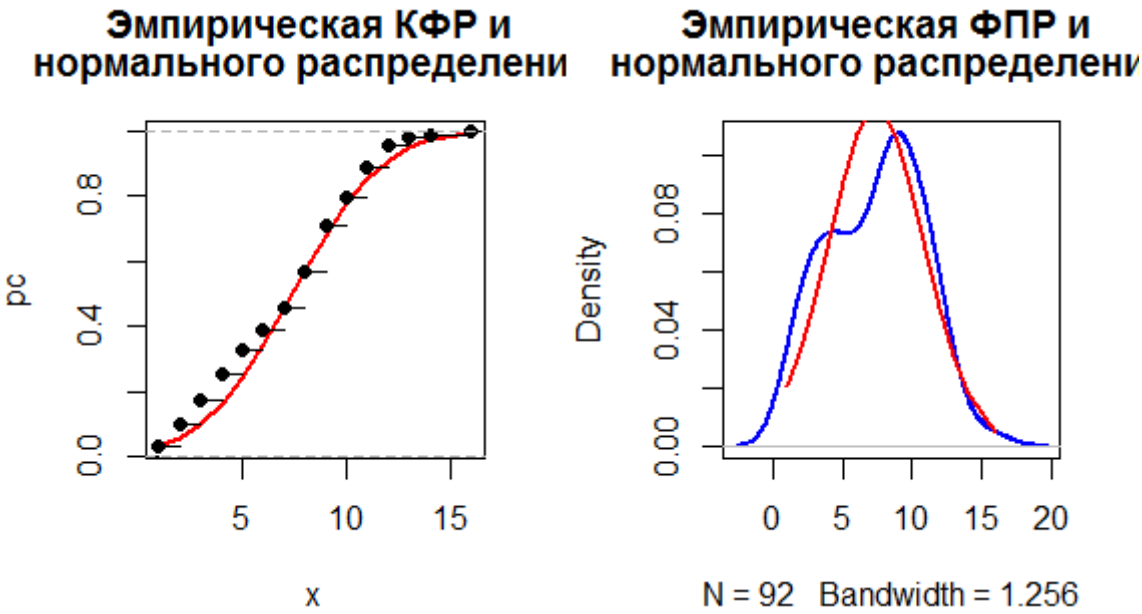


Рисунок 32 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «Ретикулоцитов» после лечения

Оценка параметров для нормального распределения:

$Mean$: 10.1086957; sd: 2.9944025;

Критерий Колмогорова-Смирнова: $D = 0.11647, p - value = 0.1648$

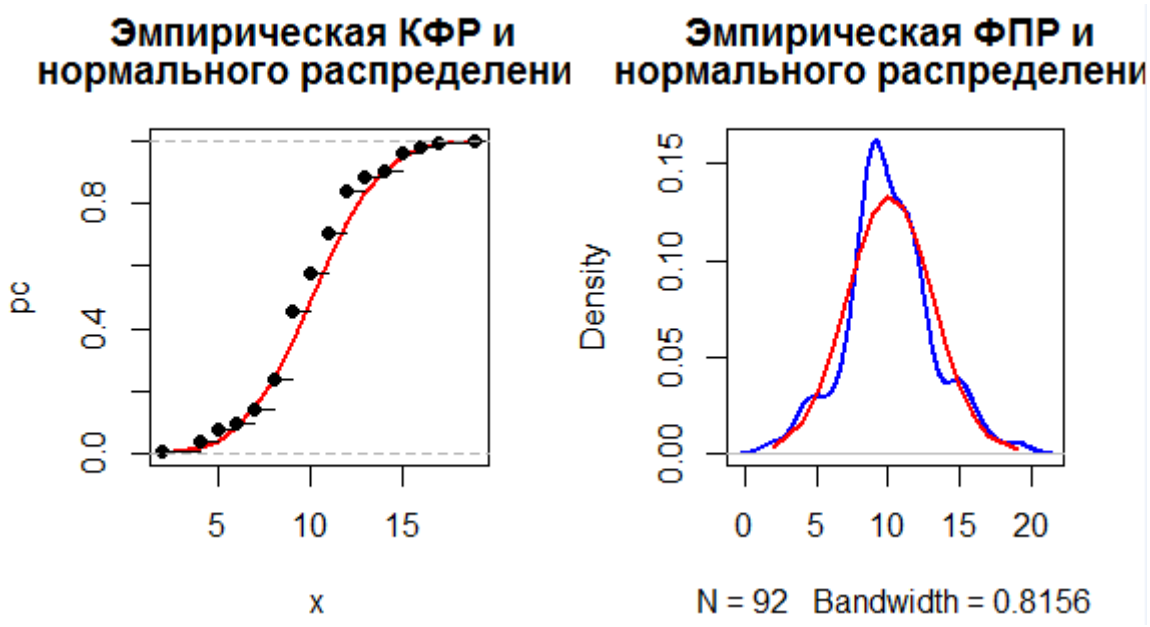


Рисунок 33 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «Ретикулоцитов» ч/з месяц лечения

Оценка параметров для нормального распределения:

$$Mean: 8.8695652; \quad sd: 2.9200111;$$

Критерий Колмогорова-Смирнова: $D = 0.11647, p - value = 0.1648$

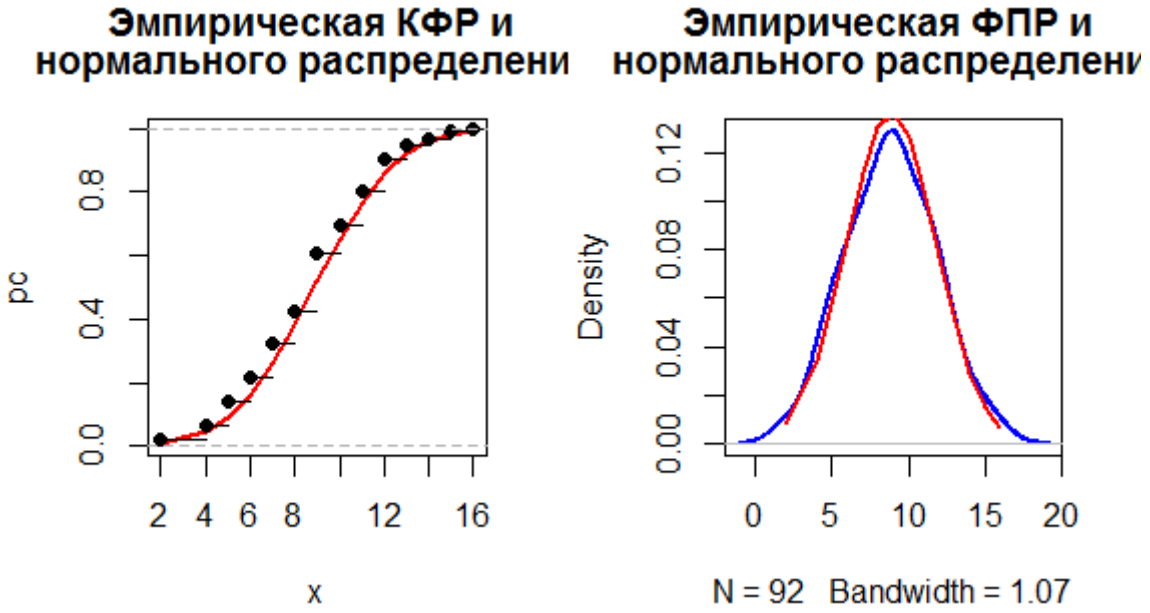


Рисунок 34 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «Гемоглобина» до лечения

Оценка параметров для логнормального распределения:

$Meanlog$: 4.670409191; $sdlog$: 0.073292109;

Критерий Колмогорова-Смирнова: $D = 0.13901, p - value = 0.05714$

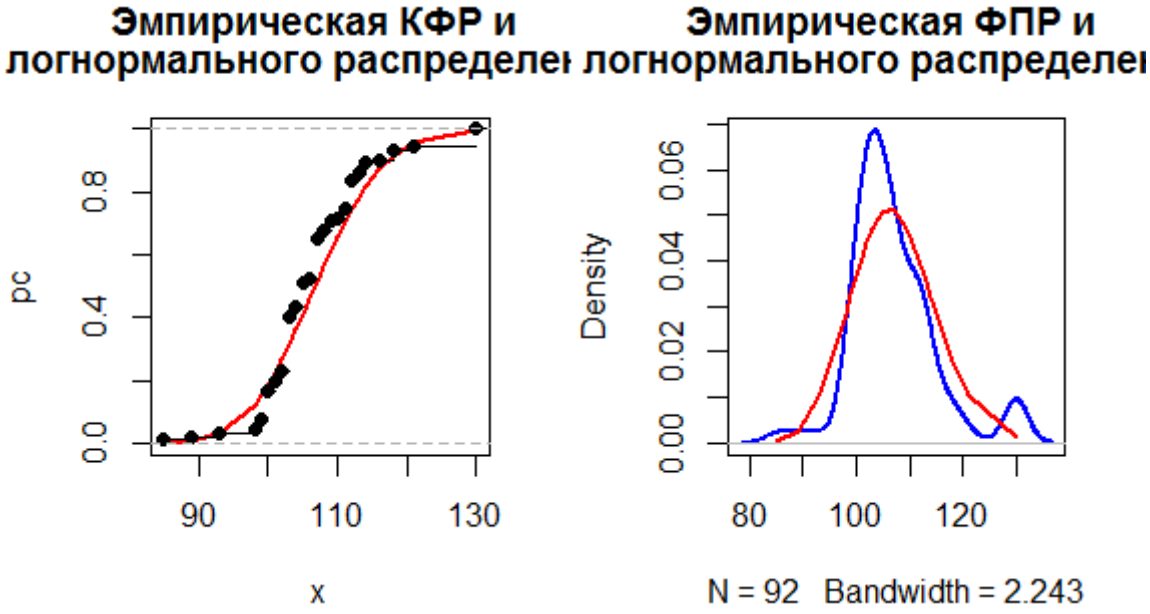


Рисунок 35 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «Гемоглобина» после лечения

Оценка параметров для нормального распределения:

$Mean$: 112.4130435; sd: 6.9878033;

Критерий Колмогорова-Смирнова: $D = 0.094743, p - value = 0.3808$

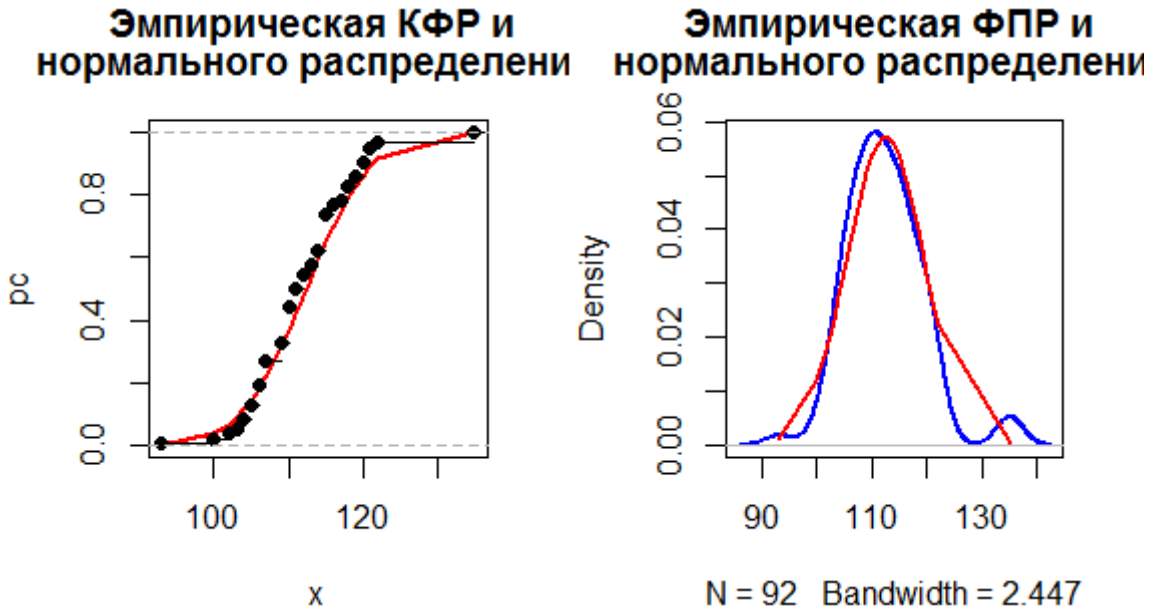


Рисунок 36 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «Гемоглобина» ч/з месяц лечения

Оценка параметров для нормального распределения:

$Mean$: 113.0434783; sd: 7.2019216;

Критерий Колмогорова-Смирнова: $D = 0.087617, p - value = 0.48$

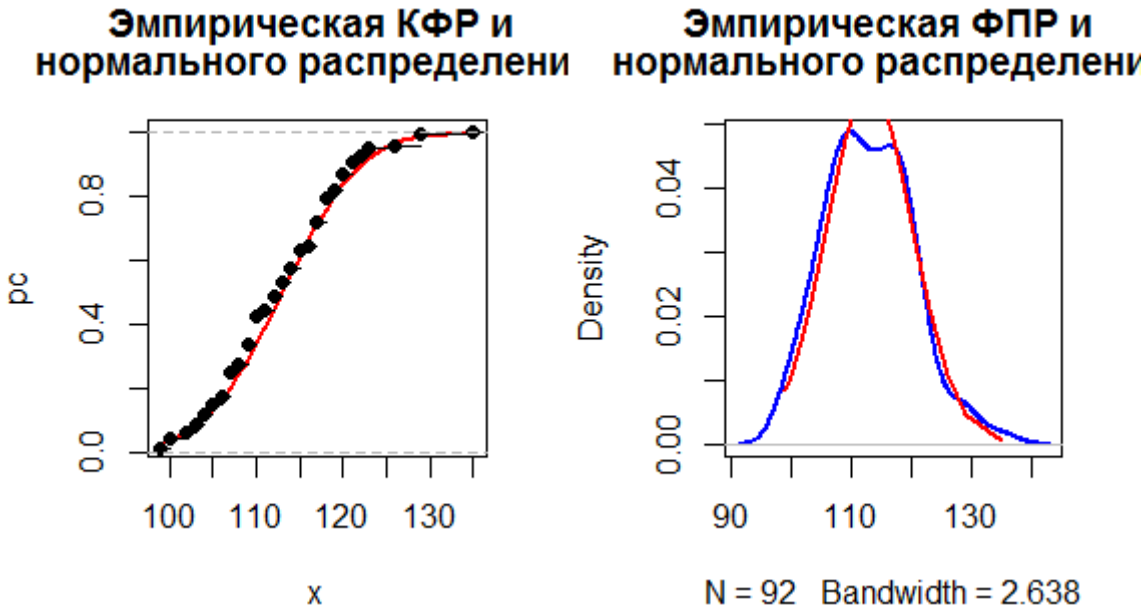


Рисунок 37 – Эмпирические и теоритические функции распределения и плотности

Ниже приведена таблица распределений для четырех показателей крови в исследуемые моменты времени:

Таблица 3 – Распределения наиболее информативных показателей крови

	до лечения	после лечения	ч/з месяц лечения
трансферин	$LN(\mu, \sigma^2)$	$N(\mu, \sigma^2)$	$N(\mu, \sigma^2)$
лейкоциты	$N(\mu, \sigma^2)$	$N(\mu, \sigma^2)$	$N(\mu, \sigma^2)$
ретикулоциты	$N(\mu, \sigma^2)$	$N(\mu, \sigma^2)$	$N(\mu, \sigma^2)$
гемоглобин	$LN(\mu, \sigma^2)$	$N(\mu, \sigma^2)$	$N(\mu, \sigma^2)$

Как видно из таблицы, значения показателей крови имеют нормальное распределение за исключением «трансферина» и «гемоглобина», которые в момент времени «до лечения» распределены логнормально.

Рассмотрим распределение значений интегрального показателя на основе информационной меры кульбака для второй группы (группа пациентов, проходивших лечение с мексидолом):

Подбор закона распределения для «$I_{кр}$» до лечения

Оценка параметров для нормального распределения:

$Mean$: 0.493; sd: 0.31584;

Критерий Колмогорова-Смирнова: $D = 0.19927, p - value = 0.1845$

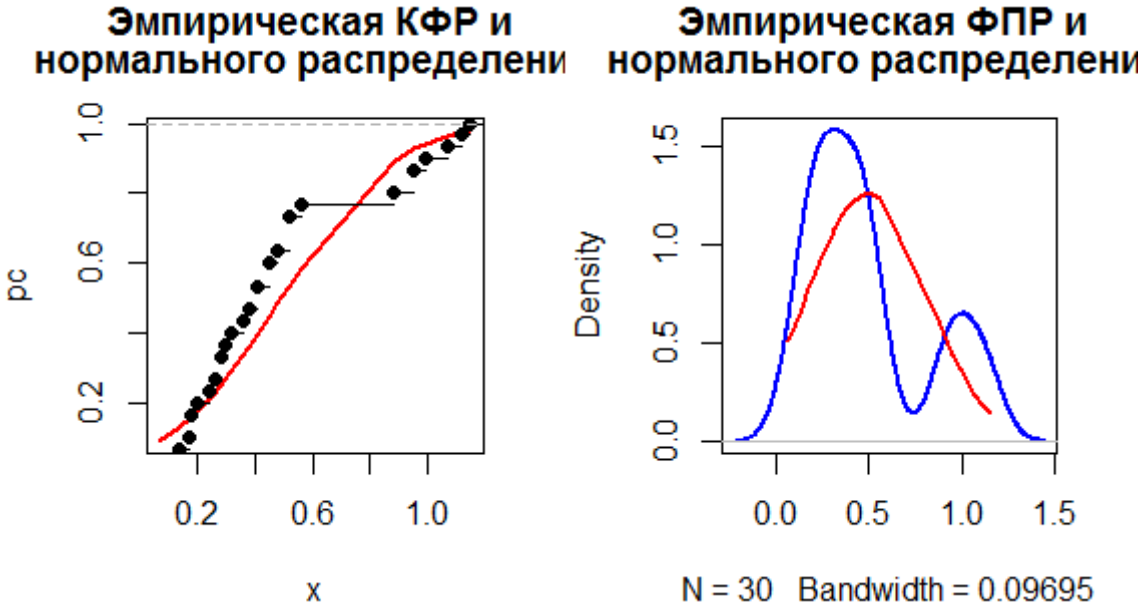


Рисунок 38 – Эмпирические и теоритические функции распределения и плотности

Оценка параметров для логнормального распределения:

$Meanlog$: -0.922; $sdlog$: 0.6816;

Критерий Колмогорова-Смирнова: $D = 0.11139, p - value = 0.8506$

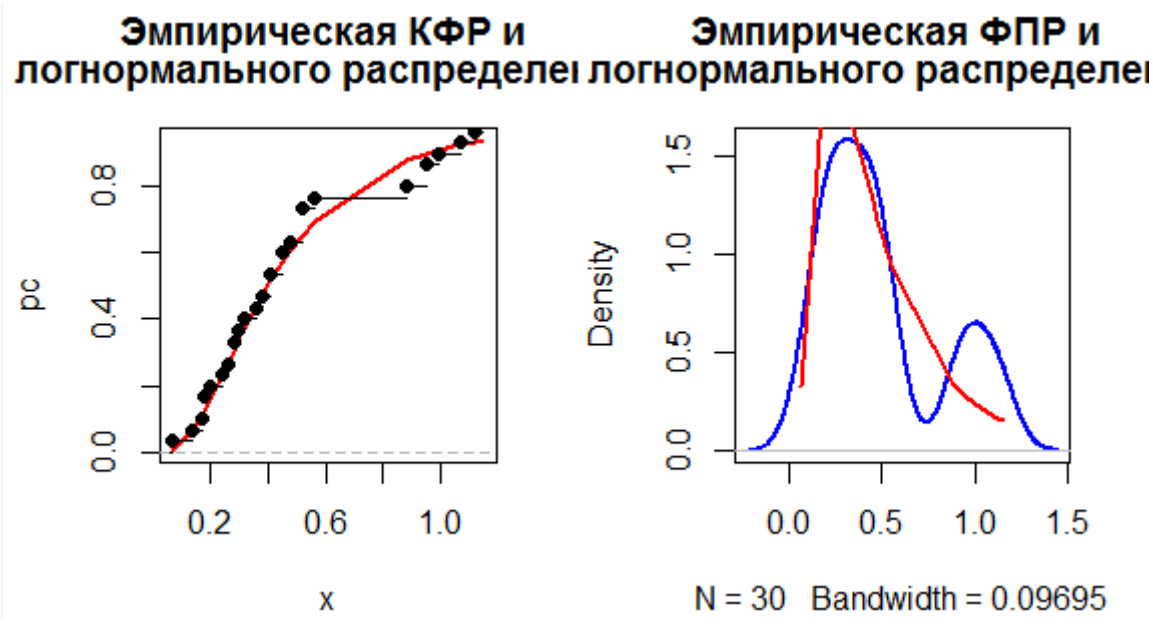


Рисунок 39 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «$I_{кр}$» после лечения

Оценка параметров для нормального распределения:

$$Mean: 0.9163;\ \ sd: 0.392169;$$

Критерий Колмогорова-Смирнова: $D = 0.13246, p - value = 0.6684$

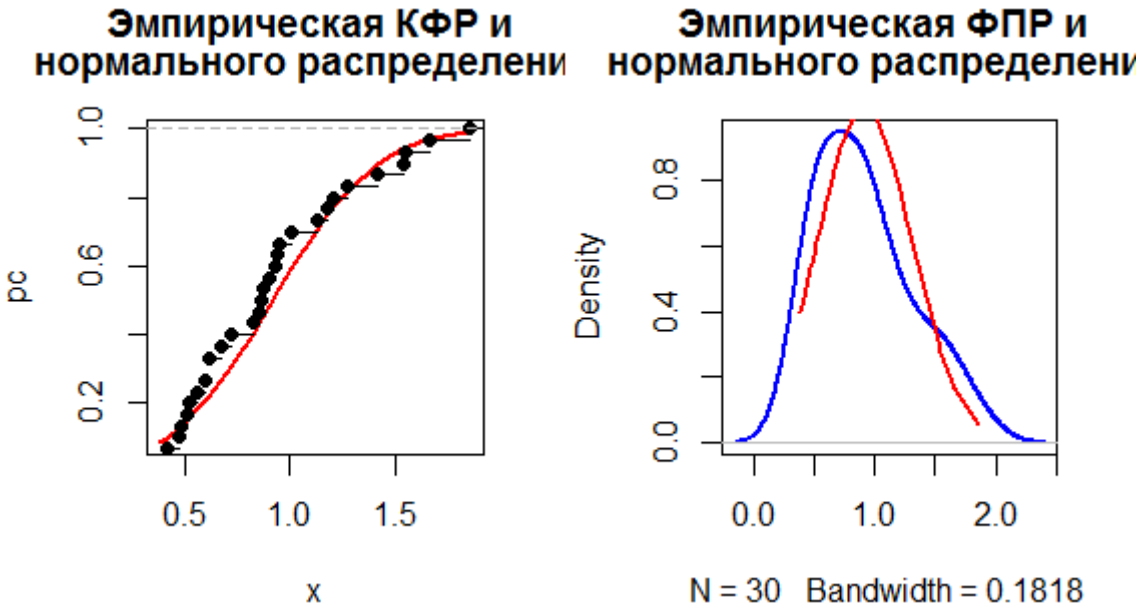


Рисунок 40 – Эмпирические и теоритические функции распределения и плотности

Подбор закона распределения для «$I_{кр}$» ч/з месяц лечения

Оценка параметров для нормального распределения:

$$Mean: 0.8566;\ \ sd: 0.32964;$$

Критерий Колмогорова-Смирнова: $D = 0.091952,\ p - value = 0.9616$

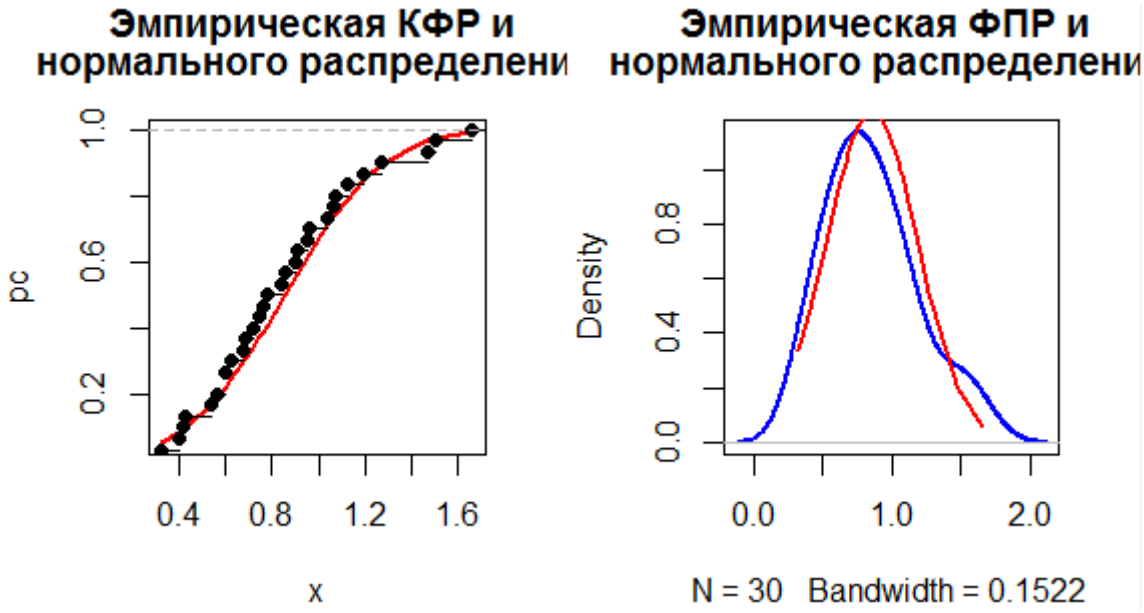


Рисунок 41 – Эмпирические и теоритические функции распределения и плотности

Для значений интегрального показателя на основе информационной меры кульбака $-I_{\text{кр}}$ в момент времени «до лечения» выдвигалось две гипотезы:

1) $I_{\text{кр}}$ распределен нормально
2) $I_{\text{кр}}$ распределен логнормально

Как видно из рисунков для подбора закона распределения для $I_{\text{кр}}$ до лечения, значения $I_{\text{кр}}$ больше соответствуют логнормальному распределению. Такой вывод также можно сделать исходя из результата критерия Колмогорова-Смирнова: $p - value = 0.8506$, который больше чем результат того-же теста для проверки нормального распределения. Однако, значение теста для нормального распределения $p - value = 0.1845$ больше чем 0.05, поэтому можно говорить, что значения интегрального показателя до лечения имеет нормальное распределение.

4.2 Проверка соответствия нормальному распределению интегрального показателя

Как было видно из главы 5.1. в момент времени «до лечения» при проверке на нормальность распределения значения величины $p - value$ для теста Колмогорова-Смирнова самое маленькое среди остальных моментов времени. Такое можно объяснить тем, что как раз в этот момент времени

рассматриваемые показатели крови также дали наименьший показатель $p - value$ при проверке на нормальность. Поэтому в этой главе будут исследоваться на нормальность интегральные критерии именно в момент времени «до лечения».

Воспользуемся самым примитивным способом проверки распределения данных – построением гистограммы. Построим гистограммы распределений для интегральных показателей в момент времени «до лечения»:

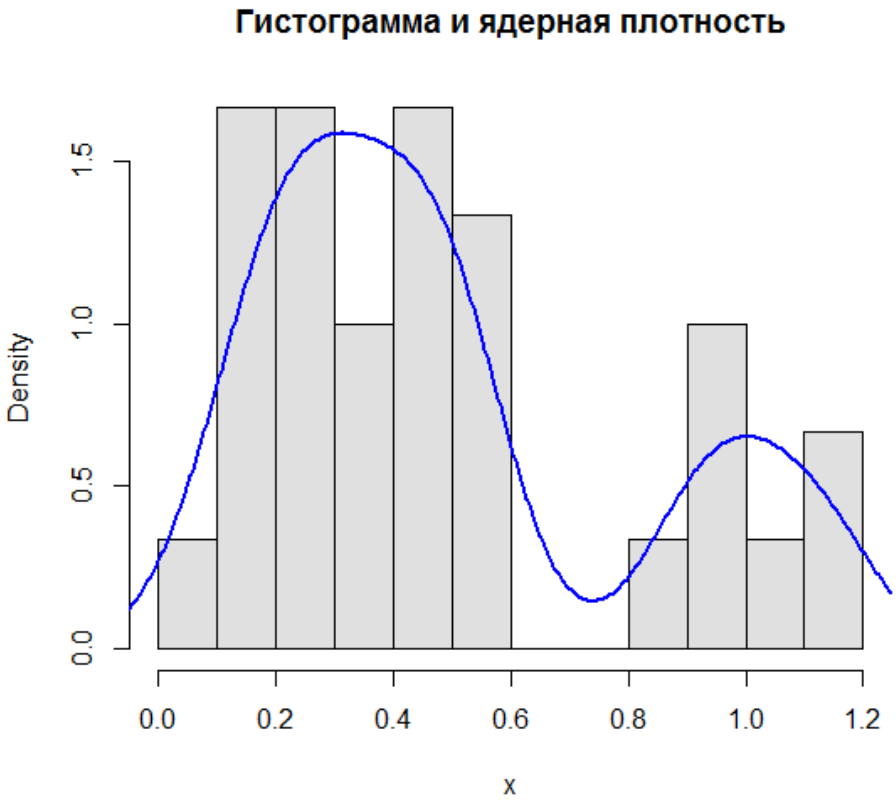


Рисунок 42 – Гистограмма и плотность первого интегрального показателя

Несмотря на то, что графики гистограмм для интегральных показателей имеют не симметричный вид, все же можно найти сходство с плотностью нормального распределения. Но нужно отметить, что график гистограммы зависит от величины выборки и шага, поэтому следует рассмотреть другой способ проверки на нормальность распределения.

Для проверки на нормальность распределения построим графики эмпирического и теоритического квантилей. Если выборка действительно

соответствует нормальному распределению, то ее значения должны выстраиваться в прямую линию под углом $\frac{\pi}{2}$. В конкретном случае такой подход очень полезен ввиду маленького размера выборки (30 значений в рассматриваемой группе).

Построим график квантилей для значений интегрального показателя на основе меры кульбака в момент времени «до лечения»:[19]

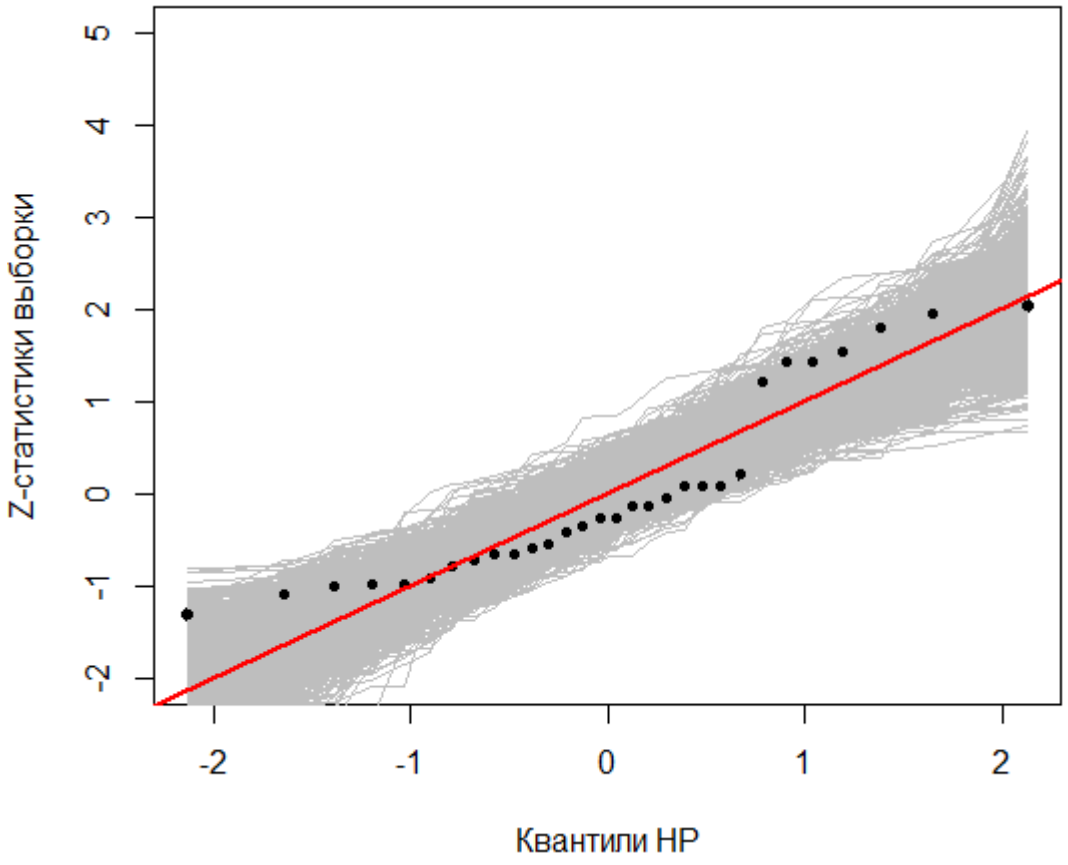


Рисунок 43 – Квантили для первого интегрального показателя

На основе тестов Колмогорова-Смирнова, графиков функций распределения, плотностей распределения и квантилей интегральных показателей можно сделать вывод, что $I_{кр}$ второй группы во все рассматриваемые моменты времени имеют нормальное распределение.

5.2 Доверительные интервалы для интегральных показателей

Строятся доверительные интервалы для интегральных показателей второй группы. Поскольку выборка значений очень мала, то к ней применяется

$bootstrap-$ распределение и строится доверительный интервал уже для новой большой выборки данных:

Таблица 4 – Доверительные интервалы для интегрального критерия с использованием меры кульбака для группы – 2.

	до	после	ч/з месяц
Нормальное распределение	$I\in$(0.37 ,0.61)	$I\in$(0.77 ,1.06)	$I\in$(0.73 ,0.98)
Bootstrap – распределение	$I\in$(0.37 ,0.59)	$I\in$(0.79 ,1.07)	$I\in$(0.73 ,0.97)

Таблица 5 – Доверительные интервалы для интегрального критерия с использованием меры кульбака для группы – 1.

	до	после	ч/з месяц
Нормальное распределение	$I\in$(0.29 ,0.48)	$I\in$(0.60 ,0.86)	$I\in$(0.49 ,0.73)
Bootstrap – распределение	$I\in$(0.29 ,0.48)	$I\in$(0.60 ,0.85)	$I\in$(0.48 ,0.71)

Таблица 6 – Доверительные интервалы для интегрального критерия с использованием меры кульбака для группы – 0.

	до	после	ч/з месяц
Нормальное распределение	$I\in$(0.88 ,1.13)	$I\in$(0.88 ,1.17)	$I\in$(0.90 ,1.18)
Bootstrap – распределение	$I\in$(0.89 ,1.13)	$I\in$(0.87 ,1.16)	$I\in$(0.91 ,1.18)

Как видно из таблиц для сравнения доверительных интервалов, разница между исходным распределением и $bootstrap-$ распределением почти незаметна.

5.3 Проверка адекватности модели нейронной сети

Проверим, насколько построенная нами модель нейронной сети выдает «адекватные» значения интегральных показателей в последний момент времени. Для этого рассмотрим обученную НС для второй группы лечения. Возьмем первую группу лечения, которая не участвовала в обучении НС. Значения интегральных показателей первой группы нам известны. Далее подставим значения $I_{кр}$ в момент времени «до лечения» первой группы в обученную НС и получим на выходе прогнозные значения интегрального показателя, которые следует сравнить с уже известными значениями. Таким образом, сравниваются теоретические и практические значения интегральных показателей.

Ниже приведены столбцы значений, которые будут сравниваться:

Таблица 7 – Теоритические и практические значения интегральных показателей

№ показателя	теоретическое $I_{кр}$	практическое $I_{кр}$
1	0.46	-0.36
2	0.32	1.56
3	0.62	0.88
4	0.38	0.64
5	1.49	-0.03
6	0.44	0.76
7	1.10	1.45
8	0.38	0.83
9	0.40	0.47
10	1.19	0.37
11	0.32	0.76
12	0.45	0.83

13	0.43	1.06
14	0.25	0.84
15	0.39	0.76
16	0.25	0.74
17	0.32	0.15
18	0.38	0.15
19	1.08	1.04
20	0.42	1.05
21	0.67	0.76
22	0.77	0.68
23	0.40	0.35
24	0.90	-0.04
25	0.48	0.76
26	0.46	0.71
27	0.62	0.76
28	0.96	0.76
29	0.88	1.47
30	1.02	1.28

Теоретическое $I_{\text{кр}}$ − это известные значения интегрального показателя первой группы в прогнозируемый момент времени;
практическое $I_{\text{кр}}$ − это значений, полученный в результате работы НС для второй группы, на вход которой подавались значений интегрального критерий первой группы в самый первый момент времени.

Проверим гипотезу о равенстве дисперсий двух выборок (теоретическое и практическое $I_{\text{кр}}$). Для этого воспользуемся F −критерием для нормально распределенных величин.
Результат F − критерия:

$$F = 0.52395, num\ df = 29, denom\ df = 29, p - value = 0.0871$$

Доверительный интервал: $0.2493807 \ldots 1.1008123$

Соотношение отклонений: 0.5239479

Как видно, значение $p - value$ превышает $5\% -$ й уровень значимости. Поэтому нулевая гипотеза о равенстве дисперсий принимается и, как следствие, можно судить об адекватности модели нейронной сети для второй группы.

6 Программный модуль

Все алгоритмы поиска оптимального решения, описанные в этой работе, а так же вычисление интегрального показателя и сплайн-функции, реализовывались на языке C++. Для ускорения вычислительного процесса была распараллелена самая трудоемкая часть – вычисление функционала (5). Для распараллеливания использовалась библиотека языка C++ – OpenMp. Для построения нейронных сетей использовалась библиотека, написанная на языке Python, – Keras.

В **приложении A** находится функция, реализующая *построение коэффициентов сплайна*, функция, которая возвращает *значение сплайна в точке*, а так же *функция отрисовки сплайна*.

В **приложении B** находится функция, которая строит *значение интегрального показателя* по входным данным результатов крови.

Заголовочный файл и файл *реализации методов симуляции отжига* находятся в **приложении C**.

Заголовочный файл и файл *реализации методов генетического алгоритма* находятся в **приложении D**.

Заголовочный файл и файл *реализации алгоритма поиска в ширину* находятся в **приложении E**.

7 ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ

ЗАДАНИЕ ДЛЯ РАЗДЕЛА «ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8БМ61	Кисатову Марату Александровичу

Школа	Инженерная школа информационных технологий и робототехники	**Отделение**	Информационных технологий
Уровень образования	магистратура	**Направление/специальность**	01.04.02 Прикладная математика и информатика

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. *Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих*	*Оклады участников проекта, нормы рабочего времени, ставки налоговых отчислений во внебюджетные фонды, районный коэффициент по г. Томску*
2. *Нормы и нормативы расходования ресурсов*	
3. *Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования*	
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Оценка коммерческого и инновационного потенциала НТИ*	– *потенциальные потребители результатов исследования;* – *SWOT – анализ.*
2. *Разработка устава научно-технического проекта*	– *определение цели и результатов проекта;* – *определение участники проекта.*
3. *Планирование процесса управления НТИ: структура и график проведения, бюджет, риски и организация закупок*	– *структура работ в рамках научного исследования;* – *определение трудоемкости выполнения работ и разработка графика проведения научного исследования;* – *бюджет научно - технического исследования.*
4. *Определение ресурсной, финансовой, экономической эффективности*	*оценка сравнительной эффективности исследования.*
Перечень графического материала*(с точным указанием обязательных чертежей):*	
1. Матрица SWOT 2. График проведения и бюджет НТИ	

Дата выдачи задания для раздела по линейному графику	07.03.2018 г

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Старший преподаватель ШИП	Шаповалова Наталья Владимировна			

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8БМ61	Кисатов Марат Александрович		07.03.2018

ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ

7.1 Предпроектный анализ

При осуществлении любой проектной и научно-исследовательской деятельности большую роль играет экономическое обоснование работ. Понятие «экономическое обоснование работ» включает в себя: определение потенциальных потребителей и сегмента рынка, сравнительный анализ предлагаемого решения по отношению к конкурентам, определение себестоимости разработки. Себестоимость проекта содержит в себе следующие статьи затрат: основная и дополнительная заработная плата участников проекта, затраты на необходимое оборудование и его амортизацию и прочие накладные расходы. Таким образом, происходит формирование бюджета научно - исследовательского проекта, в соответствии с календарным.

Потенциальными потребителями результатов исследования является гор. Больница №4. (г. Томск). Вообще потребителями могут быть любые лечебные учреждения, которые заинтересованы этими исследованиями. В будущем данная система поддержки принятия решений может быть полезна другим учреждениям (если поменять начальные данные).

7.1.1 SWOT-анализ

SWOT – Strengths (сильные стороны), Weaknesses (слабые стороны), Opportunities (возможности) и Threats (угрозы) – представляет собой комплексный анализ научно-исследовательского проекта. SWOT- анализ применяют для исследования внешней и внутренней среды проекта.

В таблице 1 представлена матрица SWOT-анализа работы.

Таблица 1 – Матрица SWOT-анализа

Сильные стороны научно-исследовательского проекта: С1: Работа реализована в математических пакетах Matlab, phyton, c++ С2: Нулевые денежные затраты на разработку. С3: Высокая точность и достоверность результатов С4: Разработка эффективных методов нахождения оптимального решения (оптимального управляющего воздействия на организм человека).	**Слабые стороны научно-исследовательского проекта:** Сл1: Небольшой объем исходных данных. Сл2: Сильная зависимость алгоритмов от базы данных (при решении задачи с другим набором данных необходимо вносить изменения в работу алгоритмов) Сл3: Необходимость наличия большой оперативной памяти у компьютера, на котором производятся расчеты.
Возможности: В1. Доработка в связи с пожеланиями. В2. Возможность продать разработку. В3. Возможность интегрировать с другими системами.	**Угрозы:** У1. Отсутствие спроса на доработку. У2. Развитая конкуренция технологий производства.

Разрабатываемый продукт включает в себя эффективные методы поиска оптимального управляющего воздействия для лечения пациента.

7.2 **Инициация проекта**

Инициация проекта состоит из процессов, которые выполняются для нового проекта или новой стадии проекта. Для этого определяются начальные цели, содержание, фиксируются ресурсы. Также определяются внутренние и внешние заинтересованные стороны проекта.

Заинтересованные стороны проекта отображены в таблице 4.

Таблица 4 – Заинтересованные стороны проекта

Заинтересованные стороны проекта	Ожидания заинтересованных сторон
Гор. Больница №3 (г. Томск)	Реализация информационной системы интеллектуальной поддержки принятия решения для выбора управляющего воздействия

В таблице 5 представлена информация о целях проекта, критериях достижения целей, а также требования к результатам проекта.

Таблица 5 – Цели и результаты проекта

Цели проекта	Создание информационной системы поддержки принятия решений.
Ожидаемые результаты проекта	Готовая информационная система поддержки принятия решения для оптимального процесса лечения пациентов.
Критерии приемки результата проекта	Технически реализованная информационная система с сопутствующей технической документацией

Требования к результату проекта	Функционирующая информационная система. Простой графический интерфейс. Достоверность получаемых результатов.

Рабочая группа проекта отображена в таблице 6.

Таблица 6 – Рабочая группа проекта

ФИО, основное место работы, должность	Роль в проекте	Функции	Трудозатраты, ч.
Гергет О.М., ТПУ, кафедра ПИ, кандидат техн. наук, доцент	Научный руководитель	Консультирование, определение задач, контроль выполнения.	512
Кисатов М.А.., ТПУ, кафедра ПИ, студент	Студент (дипломник)	Анализ литературных источников, разработка алгоритмов, , программирование, эксперименты.	1392

7.3 Планирование управлением научно-технических проектом

7.3.1 Структура работ в рамках научного исследования

Для организации и систематизации работы выпускника необходимо сформировать план работ. Данный этап предназначен для обеспечения своевременного и эффективного выполнения задания ВКР.

В таблице 3 представлен перечень этапов, работ и распределение исполнителей. В качестве исполнителей были выбраны следующие участники процесса:

- студент (С);
- научный руководитель (НР).

Таблица 2 - Перечень этапов, работ и распределение исполнителей

Основные этапы	**№ раб**	**Содержание работ**	**Должность исполнителя**
Подготовительный этап	1	Выбор темы ВКР	студент научный руководитель
	2	Получение ТЗ	студент научный руководитель
	3	Подбор материала, его анализ и обобщение	студент частично научный рук.
	4	Выбор методов выполнения работы	студент
	5	Календарное планирование работ по теме	научный руководитель студент
Основной этап	6	Построение интегрального показателя для состояния здоровья человека	студент
	7	Разработка алгоритмов выбора управляющего воздействия для организма человека	студент
	8	Применение алгоритмов на реальных данных и сравнительный анализ	студент
Заключительный этап	9	Составление отчета о проделанной работе и оценка эффективности полученных результатов	научный руководитель студент
	10	Защита дипломного проекта	студент

В результате анализа работы получилось 10 этапов разработки и 2 исполнителя: научный руководитель и студент.

7.3.2 Определение трудоемкости выполнения работ

Трудовые затраты в большинстве случаях образуют основную часть стоимости разработки, поэтому важным моментом является определение трудоемкости работ каждого из участников научного исследования.

Трудоемкость выполнения научного исследования оценивается экспертным путем в человеко-днях и носит вероятностный характер, т.к. зависит от множества трудно учитываемых факторов. Для определения ожидаемого (среднего) значения трудоемкости используется следующая формула:

$$t_{ожi} = \frac{3t_{\min i} + 2t_{\max i}}{5}, \quad (11)$$

где $t_{ожi}$ – ожидаемая трудоемкость выполнения *i*-ой работы чел.-дн.;

$t_{\min i}$ –минимально возможная трудоемкость выполнения заданной *i*-ой работы (оптимистическая оценка: в предположении наиболее благоприятного стечения обстоятельств), чел.-дн.;

$t_{\max i}$ – максимально возможная трудоемкость выполнения заданной *i*-ой работы (пессимистическая оценка: в предположении наиболее неблагоприятного стечения обстоятельств), чел.-дн.

Для построения линейного графика необходимо рассчитать длительность этапов в рабочих днях, а затем перевести ее в календарные дни. Для этого необходимо воспользоваться следующей формулой:

$$T_{кi} = T_{рi} \cdot k_{кал}, \quad (12)$$

где $T_{кi}$ – продолжительность выполнения i-й работы в календарных днях;

$T_{рi}$ – продолжительность выполнения i-й работы в рабочих днях;

$k_{кал}$ – коэффициент календарности.

Коэффициент календарности определяется по следующей формуле:

$$k_{кал} = \frac{T_{кал}}{T_{кал} - T_{вых} - T_{пр}}, \quad (13)$$

где $T_{кал}$ – количество календарных дней в году (365);

$T_{вых}$ – количество выходных дней в году (52);

$T_{пр}$ – количество праздничных дней в году (10).

$$k_{КАЛ} = \frac{365}{365 - 52 - 10} = 1,2.$$

В таблице 4 приведен расчет определения продолжительности этапов работ и их трудоемкости по исполнителям, занятым на каждом этапе. По показанию полученных величины трудоемкости этапов по исполнителям $T_{кі}$ построен линейный график осуществления проекта, который представлен в таблице 5.

Таблица 3 – Определение временных затрат на проект

№ работы	Продолжительность работ, дни			Исполнители	Трудоемкость работ по исполнителям чел.- дн.			
	tmin,чел-дни	tmax, чел-дни	$T_{ожі}$ чел-дни		$T_{РД}$		$T_{КД}$	
					НР	И	НР	И
1	2	6	3,6	С, НР	1,8	1,8	2	2
2	4	6	4,8	С, НР	2,4	2,4	3	3
3	12	20	15,2	С	0	15,2	0	18
4	12	19	14,8	С	0	14,8	0	18
5	5	8	6,2	НР, С	3,1	3,1	4	4
6	11	18	13,8	С	0	13,8	0	17
7	29	33	30,6	С	0	30,6	0	37
8	10	12	10,8	С	0	10,8	0	13
9	12	15	13,2	НР, С	6,6	6,6	8	8
10	1	1	1	С	0	1	0	1
Итого			114		13,9	100,1	17	120

На основе полученных данных строится Диаграмма Ганта, которая наглядно отображает оптимальные сроки начала и окончания выполнения работ. Результат представлен в таблице 5.

Таблица 4 – Календарный план-график

№	Вид работ	С	НР	Продолжительность выполнения работ											
				февраль			март			апрель			май		
				1	2	3	1	2	3	1	2	3	1	2	3
1	Выбор темы ВКР	2	2												
2	Получение ТЗ	3	3												
3	Подбор материала, его анализ и обобщение	0	18												
4	Выбор метода выполнения работы	0	18												
5	Календарное планирование работ по теме	4	4												
6	Построение интегрального критерия для состояния здоровья человека	0	17												
7	Программная реализация алгоритмов поиска оптимального решения, среди которых два эвристических, основываются на вероятностном поиске и один – реализует полный перебор.	0	37												
8	Тестирование алгоритмов	0	13												
9	Составление отчета о проделанной работе и оценка эффективности полученных результатов	8	8												
10	Защита дипломного проекта	0	1												
	-научный руководитель	-студент													

7.3.3 Бюджет научно-технического исследования

Формирование бюджета для выполнения научно-технического исследования складывается из следующих статей:

- материальные затраты;
- оборудование для научного исследования;
- основная заработная плата;
- дополнительная заработная плата;
- отчисления во внебюджетные фонды (страховые отчисления);
- накладные расходы.

Затраты на материалы

В эту статью включаются затраты на приобретение всех видов материалов, комплектующих изделий и полуфабрикатов, необходимых для выполнения работ по данной теме.

Расчет стоимости материальных затрат производится по действующим прейскурантам или договорным ценам. В стоимость материальных затрат включают транспортно-заготовительные расходы (3 – 5 % от цены). В эту же статью включаются затраты на оформление документации (канцелярские принадлежности, тиражирование материалов).

Расчет затрат на материалы приведён в таблице 6.

Таблица 5 – Материальные затраты

Наименование	Единица измерения	Количество	Цена за ед., руб.	Затраты на материалы, руб.
Бумага	Лист	110	1,9	209
Картридж для принтера	Шт.	1	1000	1000
Итого		1209	1140	2300

Специальное оборудование для научных (экспериментальных) работ

В данную статью включают все затраты, связанные с приобретением специального оборудования (приборов, контрольно-измерительной аппаратуры, стендов, устройств и механизмов), необходимого для проведения работ по конкретной теме. Расчет затрат на спецоборудование для научных работ представлен в таблице 7.

Таблица 6 – Расчет затрат по статье «Спецоборудование для научных работ»

Наименование	Единица измерения	Количество	Цена за ед., руб.	Затраты на материалы, руб.
Компьютер	шт.	2	41300	82600
МФУ		1	8550	8550
Монитор		4	7 690	30760
Итого				121910

Основная заработная плата

В настоящую статью включается основная заработная плата научных и инженерно-технических работников, рабочих макетных мастерских и опытных производств, непосредственно участвующих в выполнении работ по данной теме. Величина расходов по заработной плате определяется исходя из трудоемкости выполняемых работ и действующей системы оплаты труда.

В состав основной заработной платы включается премия, выплачиваемая ежемесячно из фонда заработной платы (размер определяется Положением об оплате труда):

$$C_{зп} = З_{осн} + З_{доп}, \quad (14)$$

где $З_{осн}$ – основная заработная плата;

$З_{доп}$ – дополнительная заработная плата (12-20 % от $З_{осн}$).

Основная заработная плата ($З_{осн}$) руководителя (лаборанта, инженера) от предприятия (при наличии руководителя от предприятия) рассчитывается по следующей формуле:

$$З_{осн} = З_{дн} \cdot T_{раб}, \quad (15)$$

где $З_{осн}$ – основная заработная плата одного работника;

$T_{раб}$ – продолжительность работ, выполняемых научно-техническим работником, раб. дн.;

$З_{дн}$ – среднедневная заработная плата работника, руб.

Среднедневная заработная плата рассчитывается по формуле:

$$З_{дн} = \frac{З_{м} \cdot M}{F_{д}}, \quad (16)$$

где $З_{м}$ – месячный должностной оклад работника, руб.;

M – количество месяцев работы без отпуска в течение года:

при отпуске в 24 раб. дня M =11,2 месяца, 5-дневная неделя;

при отпуске в 48 раб. дней M =10,4 месяца, 6-дневная неделя;

$F_{д}$ – действительный годовой фонд рабочего времени научно- технического персонала, раб. дн. (298 дней).

Руководителем научной работы в университете является доцент, к.т.н., заработная плата взята из системы оплаты труда в ТПУ. Заработная плата магистранта соответствует должности учебно-вспомогательного персонала ТПУ. В таблице 8 представлен расчет основной заработной платы руководителя и студента.

Таблица 7 – Расчет основной заработной платы

Исполнитель	Оклад, руб.	Районный коэффициент	Средняя заработная плата, руб./дн.	Трудоемкость, раб. дн.	Основная заработная плата, руб.
Руководитель	33664	1,3	1644,79	13,9	22862,58
Студент	9489		463,62	100,1	46408,36
				ИТОГО	**69270,94**

Дополнительная заработная плата исполнителей темы

Расчет дополнительной заработной платы ведется по следующей формуле:

$$З_{доп} = З_{осн} \cdot k_{доп}, \quad (17)$$

где $k_{доп}$ – коэффициент дополнительной заработной платы (на стадии проектирования принимается равным 0,12 – 0,15).

Расчет дополнительной заработной платы представлен в таблице 9.

Таблица 8 – Расчет дополнительной заработной платы

Исполнитель	Основная заработная плата, руб.	Коэффициент дополнительной заработной платы	Дополнительная заработная плата, руб.
Руководитель	39146,04	0,15	5871,91
Студент	46176,91		6926,54
		ИТОГО	12798,44

Отчисления во внебюджетные фонды (страховые отчисления)

В данной статье расходов отражаются обязательные отчисления по установленным законодательством Российской Федерации нормам органам государственного социального страхования (ФСС), пенсионного фонда (ПФ) и медицинского страхования (ФФОМС) от затрат на оплату труда работников.

Величина отчислений во внебюджетные фонды определяется исходя из следующей формулы:

$$З_{внеб} = k_{внеб} \cdot (З_{осн} + З_{доп}), \quad (18)$$

где $k_{внеб}$ – коэффициент отчислений на уплату во внебюджетные фонды (пенсионный фонд, фонд обязательного медицинского страхования и пр.).

В таблице 10 представлен результат расчета отчислений во внебюджетные фонды.

Таблица 9 – Отчисления во внебюджетные фонды

Исполнитель	Основная заработная плата, руб.	Дополнительная заработная плата, руб.
Руководитель	39146,04	5871,91
Студент	46176,91	6926,54
Коэффициент отчислений во внебюджетные фонды	0,30	
Итого	**29436,42**	

Накладные расходы

Накладные расходы учитывают прочие затраты организации, не попавшие в предыдущие статьи расходов: печать и ксерокопирование материалов

исследования, оплата услуг связи, электроэнергии, почтовые и телеграфные расходы, размножение материалов и т.д. Их величина определяется по следующей формуле:

$$З_{накл} = (сумма\ статей\ 1 \div 7) \cdot k_{нр}, \quad (19)$$

где $k_{нр}$ – коэффициент, учитывающий накладные расходы.

$З_{накл}$=(2300+121910+85322,95+12798,44+29436,42)*16% = 40282,85 руб.

Формирование бюджета затрат научно-исследовательского проекта

Рассчитанная величина затрат научно-исследовательской работы (темы) является основой для формирования бюджета затрат проекта, который при формировании договора с заказчиком защищается научной организацией в качестве нижнего предела затрат на разработку научно-технической продукции.

Определение бюджета затрат на научно-исследовательский проект по каждому варианту исполнения приведен в таблице 11.

Таблица 10 – Расчет бюджета затрат НТИ

Статьи расходов	рублей
Материальные затраты	2300
Спецоборудование для научных работ	121910
Основная заработная плата	85322,95
Дополнительная заработная плата	12798,44
Отчисления во внебюджетные фонды	29436,42
Накладные расходы	40282,85
Бюджет затрат НТИ	292050,66

Из данной таблицы можно сделать вывод, что затраты на полную реализацию проекта составляют 292050,66 рублей.

7.4 Оценка экономической эффективности проекта

В зависимости от того, в какой сфере и форме проявляется эффективность проекта различают следующие его виды: бюджетный, народнохозяйственный, коммерческий.

Разрабатываемый проект представляет собой интеллектуальную систему поддержки принятия решения. Основная цель системы минимизировать риск перехода в неблагоприятное функциональное состояние организма человека. Чтобы понять сложность решаемой задачи, достаточно сравнить интеллектуальные возможности компьютера и человека. По большинству показателей, например, по вычислительной возможности, по скорости реакции, работоспособности искусственный интеллект лидирует с большим отрывом.

Основными проблемами при принятии медицинских решений являются недостаточность знаний, ограниченность временных ресурсов, неполнота информации о состоянии пациента. Медицинские экспертные системы позволяют врачу не только проверять собственные диагностические предположения, но и дают возможность обращаться к системе за консультацией в сложных клинических случаях.

Минимизация рисков, оптимизация трудозатрат и временных ресурсов – вектор развития сферы информационных технологий в современных предприятиях любых отраслей. Соответственно, реализуемый проект предполагает получение коммерческого эффекта.

7.5 Выводы по разделу «Финансовый менеджмент, ресурсоэффективность и ресурсобережение»

В ходе выполнения раздела «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение» проведен SWOT-анализа. Выявлено, что разрабатываемый продукт включает в себя эффективные алгоритмы нахождения оптимального управляющего воздействия для лечения пациента, что будет всегда востребовано в медицинских учреждениях. Оценка готовности проекта к коммерциализации говорит о том, что перспективность данной работы средняя.

Выполнено планирование научно-технических решений, построена диаграмма Ганта, которая отображает последовательность работ, их длительность и занятость исполнителей в то или иное время.

Рассчитан бюджет НТИ и он составил 292050,66рублей. В него входят: затраты на материалы, заработные платы исполнителям, отчисления в фонды и прочие расходы.

Оценка научно-технического уровня НИР показала, что реализуемый проект имеет средний уровень научно-технического эффекта.

8 СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ

ЗАДАНИЕ ДЛЯ РАЗДЕЛА «СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
8БМ61	Кисатов Марат Александрович

Школа	Инженерная школа информационных технологий и робототехники	**Отделение**	Информационных технологий
Уровень образования	магистратура	**Направление/специальность**	01.04.02 Прикладная математика и информатика

Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования (набор показателей крови, алгоритм, методика) и области его применения	*Объектом исследования являются разработка системы интеллектуальной поддержки принятия решения в медицине.* *Разработкой интеллектуальной системы занимается инженер-программист.; рабочее место – компьютерный стол с персональным компьютером.* *Разрабатываемая информационная система применяется в медицине.*
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	

1. Производственная безопасность 1.1. Анализ выявленных вредных факторов при разработке и эксплуатации проектируемого решения в следующей последовательности: – физико-химическая природа вредности, её связь с разрабатываемой темой; – действие фактора на организм человека; – приведение допустимых норм с необходимой размерностью (со ссылкой на соответствующий нормативно-технический документ); – предлагаемые средства защиты; – (сначала коллективной защиты, затем – индивидуальные защитные средства). 1.2. Анализ выявленных опасных факторов при разработке и эксплуатации проектируемого решения в следующей последовательности: – механические опасности (источники, средства защиты; – термические опасности (источники, средства защиты); – электробезопасность (в т.ч. статическое электричество, молниезащита – источники, средства защиты); – пожаровзрывобезопасность (причины, профилактические мероприятия, первичные средства пожаротушения).	*Анализ выявленных вредных факторов при разработке и эксплуатации информационной системы поддержки принятия решения включает:* - *повышенный уровень электромагнитных излучений;* - *повышенная или пониженная влажность воздуха;* - *недостаточная освещенность рабочей зоны;* - *отсутствие или недостаток естественного света;* - *повышенный уровень шума на рабочем месте;* - *повышенная или пониженная температура воздуха рабочей зоны;* - *статические физические перегрузки;* - *нервно-психические перегрузки такие, как умственное перенапряжение, монотонность труда;* *Анализ выявленных опасных факторов:* - *статическое электричество;* - *короткое замыкание;* - *электрический ток.*
2. Экологическая безопасность: – защита селитебной зоны – анализ воздействия объекта на атмосферу (выбросы); – анализ воздействия объекта на гидросферу (сбросы); – анализ воздействия объекта на литосферу (отходы); – разработать решения по обеспечению экологической безопасности со ссылками на НТД по охране окружающей среды.	*Разрабатываемый объект не несёт негативного влияния на атмосферу и гидросферу. Анализ негативного воздействия на литосферу: утилизация люминесцентных ламп, непригодных для работы компьютеров и другой оргтехники.*
3. Безопасность в чрезвычайных ситуациях: – перечень возможных ЧС при разработке и эксплуатации проектируемого решения; – выбор наиболее типичной ЧС; – разработка превентивных мер по предупреждению ЧС; – разработка действий в результате возникшей ЧС и мер по ликвидации её последствий.	*Пожар. Обоснование мероприятий по предотвращению пожара и разработка порядка действия в случае его возникновения ЧС.*
4. Правовые и организационные вопросы обеспечения безопасности: – специальные (характерные при эксплуатации объекта исследования, проектируемой рабочей зоны) правовые нормы трудового законодательства;	*- Рабочее место при выполнении работ сидя регулируется ГОСТом 12.2.032 – 78;* *- Организация рабочих мест с электронно-вычислительными машинами регулируется СанПиНом 2.2.2/2.4.1340 – 03;*

– организационные мероприятия при компоновке рабочей зоны.	*- "Трудовой кодекс Российской Федерации" от 30.12.2001 N 197-ФЗ (ред. от 05.02.2018).*

Дата выдачи задания для раздела по линейному графику	01.03.2018

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
ассистент	Авдеева Ирина Ивановна			01.03.2018

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8БМ61	Кисатов Марат Александрович		01.03.2018

Социальная ответственность

Введение

В данном разделе проанализированы проблемы, связанные с организацией рабочего места инженера – программиста в соответствии с нормами производственной санитарии, техники безопасности, охраны труда и окружающей среды. Все вышеупомянутые аспекты регламентируются рядом соответствующих документов.

Рабочая зона программиста – офисное помещение, рабочее место – компьютерный стол с персональным компьютером.

Инженеру - программисту необходимо реализовать различные варианты решения задач на поиск оптимального управления для организма человека. Оптимальное управление представляет собой набор воздействий, применяемых к пациенту в различные моделируемые моменты времени. Алгоритмы решения оптимизационных задач различают по скорости работы, требовательности к памяти компьютера и размеру начальных данных. Разработанная система интеллектуальной поддержки принятия решения может быть полезна для сотрудников медицинских учреждений и исследовательских институтов. В данном разделе представлены вредные и опасные факторы, которые оказывают негативное влияние на организм программиста при выполнении работы за персональным компьютером. Так же описаны ЧС, которые могли случиться на рабочем месте и действия, которые необходимо выполнить в случае их возникновения.

8.1 Производственная безопасность

8.1.1 Анализ вредных и опасных факторов, которые могут возникнуть на рабочем месте при проведении исследований

Перечень опасных и вредных факторов, характерных для проектируемой производственной среды представлен в таблице 1.

Таблица 11 – Опасные и вредные факторы на рабочем месте инженера-программиста

Источник фактора, наименование видов работ	Факторы (по ГОСТ 12.0.003 - 74) [1]		Нормативные документы
	Вредные	Опасные	
Работа за персональным компьютером в офисном помещении	1) Повышенный уровень шума на рабочем месте; 2) повышенная или пониженная влажность воздуха; 3) отсутствие или недостаток естественного света; 4) повышенный уровень электромагнитных излучений; 5) повышенная или пониженная температура воздуха рабочей зоны; 6) недостаточная освещенность рабочей зоны; 7) статические физические перегрузки;	1) Повышенный уровень статического электричество; 2) короткое замыкание; 3) опасность поражения эл.током	1) Шум. Общие требования безопасности устанавливаются ГОСТ 12.1.003–83 ССБТ [2]. 2) Показатели микроклимата устанавливаются СанПиН 2.2.2.548-96 [3]. 3) Нормы освещения устанавливаются СанПиН 2.2.1/2.1.1.1278–03 [4]. 4) Допустимые уровни напряженности электростатических полей устанавливается

	8) умственное перенапряжение 9) монотонность труда.		ГОСТ 12.1.045–84 ССБТ [5]. 5) ГОСТ 12.1.004-91 ССБТ. Пожарная безопасность. Общие требования [7]. 6) Электробезопасность устанавливается по ГОСТ 12.1.038–82 ССБТ [6]. 7) ГОСТ 12.0.003-74 8) ГОСТ 12.0.003-74 9) ГОСТ 12.0.003-74

8.2 Обоснование мероприятий по защите исследователя от действия опасных и вредных факторов

Повышенный уровень шума на рабочем месте

На рабочем месте инженера-программиста в офисном помещении, исходя из ГОСТ 12.1.003–83[2], действует постоянный шум. Шум возникает в помещении кондиционерами и вентиляторами при охлаждения нагревающихся частей ЭВМ и т.д. Основной характеристикой шума является уровень звукового давления в активной полосе частот.

Шум создает значительную нагрузку на нервную систему человека, оказывая на него психологическое воздействие. Работающие в условиях длительного шумового воздействия испытывают раздражительность, головные боли, головокружение, снижение памяти, повышенную утомляемость и т. д. При выполнении основной работы на ПЭВМ уровень шума на рабочем месте не должен превышать 50 дБ [2]. Допустимый уровень шума для работы программиста представлен в таблице 2.

Таблица 12 – Предельно допустимые уровни звукового давления по ГОСТ 12.1.003–83 ССБТ

Вид трудовой деятельности/ Частоты	Уровни звукового давления, дБ, в октавных полосах со среднегеометрическими частотами, Гц								
	31,5	63	125	250	500	1000	2000	4000	8000
Научная деятельность, конструирование и проектирование, программирование, программистов вычислительных машин и т.д.	86	71	61	54	49	45	42	40	38

Если уровень шума в помещении выше допустимого, то необходимо принимать меры по снижению его уровня. Например, для снижения шума систем вентиляции и кондиционирования воздуха можно использовать различные глушители, ограничение скорости воздуха на воздухораспределительных устройствах. Средства индивидуальной защиты органов слуха работающих установлены ГОСТ 12.4.011-89 ССБТ [12] - это наушники, заглушки, вкладыши. Однако они должны использоваться лишь как дополнение к коллективным средствам защиты, когда последние не могут решить проблему борьбы с шумом.

Защита от шумов – заключение вентиляторов в защитный кожух и установление их внутри корпуса ЭВМ. Для снижения уровня шума стены и потолок помещений, где установлены компьютеры, могут быть облицованы звукопоглощающими материалами с максимальными коэффициентами звукопоглощения в области частот 63 - 8000 Гц [12].

Вывод: на рабочем месте уровень шума не должен превышать 50 Дб. В помещении, в котором работает инженер-программист, основным источником шума является кулер системного блока компьютера. Общий уровень шума компьютера не превышает 30 Дб, что не превышает допустимой нормы.

Отклонение показателей микроклимата

Выполняемые работы программистом, по степени физической тяжести, относятся к категории легких работ [3]. Оптимальные и допустимые значения показателей температуры, влажности воздуха и скорости движения воздуха в производственном помещении с ВДТ и ПЭВМ представлены в таблицах 3 и 4 соответственно, согласно СанПиН 2.2.4.548-96 [3].

Таблица 13 – Оптимальные величины показателей микроклимата на рабочих местах производственных помещений по СанПиН 2.2.4.548-96

Период года	Категория работ по уровню энергозатрат, Вт	Температура воздуха, °С	Температура поверхностей, °С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с
Холодный	Ia (до 139)	22 - 24	21 - 25	60 - 40	0,1
Теплый	Ia (до 139)	23 - 25	22 - 26	60 - 40	0,1

Таблица 14 – Допустимые величины показателей микроклимата по СанПиН 2.2.4.548-96

Период года	Категория работ по уровню энергозатрат, Вт	Температура воздуха, ^{0}С		Температура поверхностей, ^{0}С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с	
		диапазон ниже оптимальных величин	диапазон выше оптимальных величин			для диапазона температур воздуха ниже оптимальных величин, не более	для диапазона температур воздуха выше оптимальных величин, не более**
Холодный	Ia (до 139)	20,0-21,9	24,1-25,0	19,0-26,0	15-75*	0,1	0,1
Теплый	Ia (до 139)	21,0-22,9	25,1-28,0	20,0-29,0	15-75*	0,1	0,2

В помещении предусмотрена система отопления, функционирующая в зимнее время. Она обеспечивает достаточное, постоянное и равномерное нагревание воздуха. В аудитории установлена вентиляция, которая поможет регулировать температуру в помещении в летнее время. Также имеются окна, которые способствуют проветриванию помещения.

Вывод: в помещении предусмотрена система отопления, функционирующая в зимнее время. Система обеспечивает достаточное,

постоянное и равномерное нагревание воздуха. В аудитории установлена вентиляция, которая поможет регулировать температуру в помещении в летнее время. Имеются окна, которые способствуют проветриванию помещения. Согласно СанПиН 2.2.2/2.4.1340-03, в рабочих помещениях с ПЭВМ необходимо ежедневно проводить влажную уборку и каждый час проветривать помещение. Система вентиляции и отопления соответствует нормативным рекомендациям.

Недостаточная освещённость рабочей зоны

При плохой освещенности у работников ощущается усталость глаз и переутомление, что приводит к снижению работоспособности. Часто причинами являются слишком низкие уровни освещенности, слепящее действие источников света и соотношение яркостей, которое недостаточно хорошо сбалансировано на рабочих местах. Головные боли могут быть вызваны пульсацией освещения, что является результатом использования электромагнитных пускорегулирующих аппаратов (ПРА) для газоразрядных ламп, работающих на частоте 50 Гц [9].

Освещённость на рабочем месте должна соответствовать характеру зрительной работы; равномерное распределение яркости на рабочей поверхности и отсутствие резких теней; отсутствие пульсации светового потока; оптимальная направленность светового потока и оптимальный спектральный состав; все элементы осветительных установок должны быть долговечны, взрыво-, пожаро-, электробезопасны [9].

Работа за ПК относится к зрительным работам высокой точности для помещений жилых и общественных зданий. Согласно СанПиН 2.2.2/2.4.1340-03 [9], такие помещения должны удовлетворять требованиям, представленным в таблице 5.

Таблица 15 – Нормы освещенности по СанПиН 2.2.2/2.4.1340-03

Помещения	Рабочая поверхност	Естественное освещение	Совмещенное освещение	Искусственное освещение

	ь и плоскость нормирования КЕО и освещенности (Г - горизонтальная, В - вертикальная) и высота плоскости над полом, м	КЕО $e_н$, %		КЕО $e_н$, %						
		при верхнем или комбинированном освещении	при боковом освещении	при верхнем или комбинированном освещении	при боковом освещении	Освещенность, лк			Объединенный показатель дискомфорта, UGR, не более	Коэффициент пульсации освещенности, $К_п$, %, не более
						при комбинированном освещении		при общем освещении		
						всего	от общего			
1	2	3	4	5	6	7	8	9	10	11
Административные здания (министерства, ведомства, комитеты, префектуры, муниципалитеты управления, конструкторские и проектные организации, научно-исследовательские учреждения и тому подобное)										
Кабинеты, рабочие комнаты, офисы, представительства	Г-0,8	3,0	1,0	1,8	0,6	400	200	300	21	15

Освещенность рабочего помещения

Рассмотрим офисное помещение, в котором производились работы, с размерами: длина А = 5 м, ширина В = 7 м, высота *H* = 4 м. Всего имеется шесть светильников, по 4 лампы в каждом. Фактическая освещенность рассчитывается по следующей формуле:

$$E_{ф} = \frac{N \cdot n \cdot \Phi_{ст} \cdot y}{S \cdot K \cdot z}, \qquad (20)$$

где N – число светильников, шт; n – число ламп в светильнике, шт; $\Phi_{ст}$ – световой поток люминесцентной лампы, Лм (при мощности 11Вт – 750лм); y – коэффициент использования светового потока (для исследуемого помещения – 0.8); S – площадь помещения, $м^2$; k – коэффициент запаса (помещения с малым выделением пыли -

1,5); z – коэффициент неравномерного освещения (для люминесцентных ламп - 1,1).

Получаем

$$E_{ф} = \frac{6 \cdot 4 \cdot 750 \cdot 0.8}{35 \cdot 1.5 \cdot 1.1} = 249 \text{ (Лк)}.$$

Отличие от нормированного уровня

$$\Delta E = \frac{E_{ф} - E_{норм}}{E_{норм}} \cdot 100\%,$$

$$\Delta E = \frac{249 - 300}{300} \cdot 100\% = 17\%.$$

Вывод: Полученный промежуток -10% ≤ 17 % ≤ +20% попадает в интервал, который рекомендованный межотраслевыми нормативами, что свидетельствует о соблюдении норм освещенности в рабочем помещении.

Повышенный уровень электромагнитных излучений

В данной работе источником электромагнитного излучения является персональный компьютер.

Степень и характер воздействия ЭМП на организм человека зависят: от интенсивности излучения; частоты колебаний; поверхности тела, облучаемого и т.д. [11].

Для соблюдения нормативов следует руководствоваться правилами [29]:

- выбирать монитор с жидкокристаллическим экраном;
- по возможности располагать монитор в углу помещения. Так стены будут поглощать электромагнитное излучение, испускаемые боковыми и задними стенками;
- выключать монитор, даже если отходите ненадолго;
- монитор должен стоять на расстоянии вытянутой руки от кресла;
- по возможности системный блок расположить как можно дальше от вас;

– выключать компьютер, если больше не собираетесь им пользоваться;

– по возможности сокращать время, проводимое за компьютером.

Деятельность программиста проходит перед монитором, поэтому необходимо чаще делать перерыв. Для этого можно просто пройтись. Помимо опасности от электромагнитных волн излучение от монитора может нести опасность возникновения ряда глазных заболеваний, таких как близорукость или сухость глаз [9]. Согласно СанПиНу 2.2.2/2.4.1340-03 [9] приемлемые временные допустимые уровни ЭМП, создаваемых ПЭВМ на рабочих местах, указаны в таблице 6.

Таблица 16 – Временные допустимые уровни ЭМП, создаваемых ПЭВМ по СанПиНу 2.2.2/2.4.1340-03

Наименование параметров		ВДУ ЭМП
Напряженность электрического поля	в диапазоне частот 5 Гц - 2 кГц	25 В/м
	в диапазоне частот 2 кГц - 400 кГц	2,5 В/м
Плотность магнитного потока	в диапазоне частот 5 Гц - 2 кГц	250 нТл
	в диапазоне частот 2 кГц - 400 кГц	25 нТл
Электростатический потенциал экрана видеомонитора		500 В

Статические физические перегрузки

Статические перегрузки вызываются длительным пребыванием человека в вынужденной рабочей позе или длительным статическим напряжениям отдельных групп мышц при выполнении работ. Например, сидя или стоя с наклоненной головой (шейный и плечевой пояс); сидя или стоя с наклоненным туловищем (пояснично-крестцовый отдел); лежа (шейно-плечевая область); на коленках (коленные суставы); на корточках (коленные и голеностопные суставы, сдавливание нервов); с упором на локоть (давление на локтевой сустав). Основным видом статической нагрузки при исследовательской работе в лаборатории, является нахождение в неподвижном состоянии, часто в неудобной позе при работе на ЭВМ.

При этом возникает локальная динамическая перегрузка пальцев и кистей рук. Статическим перенапряжениям мышц способствуют неподходящие эргономические параметры рабочего места и его компонентов (отсутствие подлокотников, пюпитра, подставки для ног), отсутствие возможности регулировки параметров рабочего стула, высоты рабочей поверхности стола, неудобное расположение клавиатуры и дисплея.

В данном случае, в качестве нормативов обеспечения безопасного рабочего процесса, следует принимать допустимые параметры рабочей зоны пользователя ЭВМ, в соответствии с СанПиНом 2.2.2/2.4.1340-03.

Таблица 7 – Требования к организации и оборудованию рабочих мест с персональной ЭВМ (по СанПиН 2.2.2/2.4.1340-03)

Объект	Характеристика	Численное значение
Рабочий стол	Высота (при наличии возможности регулирования)	680-800мм
	Высота (при отсутствии возможности регулирования)	780 мм
	Размер рабочей поверхности – ширина (при высоте 780 мм)	800, 1000, 1200, 1400 мм
	Размер рабочей поверхности – глубина (при высоте 780 мм)	800, 1000 мм
	Пространство для ног: – высота – ширина – глубина на уровне колен – глубина на уровне вытянутых ног	– не менее 600 мм – не менее 500 мм – не менее 450 мм – не менее 650 мм
Рабочи	Ширина и глубина поверхности сиденья	не менее 400 мм
	Регулировка высоты сиденья	400-550 мм

й стул	Регулировка угла наклона сиденья	вперед до 15^{o} и назад до 5^{o}
	Высота опорной поверхности спинки	300 ± 20 мм
	Ширина опорной поверхности спинки	не менее 380 мм
	Радиус кривизны горизонтальной плоскости	400 мм
	Угол наклона спинки в вертикальной плоскости	± 30^{o}
	Регулировка расстояния спинки от переднего края сиденья	260-400 мм
	Стационарные или съемные подлокотники: – длина – ширина	– менее 250 мм □ 50-70 мм
	Регулировка подлокотников по высоте над сиденьем	230 ± 30 мм
	Регулировка внутреннего расстояния между подлокотниками	350-500 мм
Подставка под ноги	Ширина	не менее 300 мм
	Глубина	не менее 400 мм
	Регулировка по высоте	до 150 мм
	Угол наклона упорной поверхности	до 20^{o}
	Высота бортика по переднему краю	10 мм

В качестве средства профилактики заболеваний, связанных с сидячим образом жизни, предлагается использовать физкультминутки (ФМ). ФМ способствует снятию локального утомления. По содержанию ФМ различны и предназначаются для конкретного воздействия на ту или иную группу мышц или систему организма в зависимости от самочувствия и ощущения усталости.

Монотонность труда

При работе с ПЭВМ основным фактором, влияющим на нервную систему инженера-программиста, является огромное количество информации, которое он должен воспринимать. Это является сложной задачей, которая очень сильно влияет на сознание и психофизическое состояние из-за монотонности работы. Поэтому меры, позволяющие снизить воздействие этого вредного производственного фактора, которые регулируются СанПиН 2.2.2/2.4.1340-03, являются важными в работе разработчика. Они позволяют увеличить производительность труда и предотвратить появление профессиональных болезней.

Организация работы с ПЭВМ осуществляется в зависимости от вида и категории трудовой деятельности. Виды трудовой деятельности разделяются на 3 группы [9]:

- группа А – работа по считыванию информации с экрана с предварительным запросом;
- группа Б – работа по вводу информации;
- группа В – творческая работа в режиме диалога с ПЭВМ.

Работа инженера-программиста, разрабатывающего информационную систему поддержки принятия решений в данной работе относится к группам А и Б.

Категории трудовой деятельности, различаются по степени тяжести выполняемых работ. Для снижения воздействия рассматриваемого вредного фактора предусмотрены регламентированные перерывы для каждой группы работ [9] – таблица 8.

Таблица 8 – Суммарное время регламентированных перерывов в зависимости от продолжительности работы, вида категории трудовой деятельности с ПЭВМ

Категория работы с ПЭВМ	Уровень нагрузки за рабочую смену при видах работ с ПЭВМ			Суммарное время регламентированных перерывов, мин.	
	группа А,	группа Б,	группа В,	при 8-часовой смене	при 12-

	количество знаков	количество знаков	ч		часовой смене
А	до 20 000	до 15 000	до 2	50	80
Б	до 40 000	до 30 000	до 4	70	110
В	до 60 000	до 40 000	до 6	90	140

Повышенный уровень статическое электричества

Опасность возникновения статического электричества проявляется в возможности образования электрической искры и вредном воздействии его на человеческий организм, и не только в случае непосредственного контакта с зарядом, но и за счет действий электрического поля, которое возникает при заряде. При включенном питании компьютера на экране дисплея накапливается статическое электричество. Электрический ток искрового разряда статического электричества мал и не может вызвать поражение человека. Тем не менее, вблизи экрана электризуется пыль и оседает на нем. В результате чего искажается резкость восприятия информации на экране. Кроме того, пыль попадает на лицо работающего и в его дыхательные пути [5].

Основные способы защиты от статического электричества следующие: заземление оборудования, увлажнение окружающего воздуха. Также целесообразно применение полов из антистатического материала [5].

Короткое замыкание

Для защиты проводов от перегрева и предупреждения воспламенения окружающих предметов в цепь включаются аппараты защиты, например, плавкие предохранители [7].

Основной причиной возникновения коротких замыканий является нарушения изоляции электрооборудования.

Часто причиной повреждений в электрической части электроустановок являются неквалифицированные действия обслуживающего персонала.

Последствия коротких замыканий следующие:

1. механические и термические повреждения электрооборудования.

2. возгорания в электроустановках.

3. снижение уровня напряжения в сети.

4. Электромагнитное влияние на линии связи, коммуникации и т.п.

Меры по предотвращения короткого замыкания:

1. не использовать старые провода с несоответствующей изоляцией;

2. устанавливать защитные устройства отключения – автоматические выключатели, устройства защитного отключения, дифавтоматы;

3. регулярно следить за состоянием электрических точек – розеток и выключателей (при необходимости сразу же заменять);

4. не эксплуатировать поврежденные электроприборы, от которых летят искры [7].

Опасность поражения электрическим током

К опасным факторам относят поражение электрическим током согласно ГОСТ 12.0.003-74. Компьютеры, расположенные в помещении, питаются от сети 220В переменного тока с частотой 50Гц. Помещение с ПЭВМ, где проводились работы, относится к помещениям без повышенной опасности, согласно классификации помещений по опасности поражения людей электрическим током, так как отсутствуют следующие факторы:

- сырость;
- токопроводящая пыль;
- токопроводящие полы;
- высокая температура;
- возможность одновременного прикосновения человека к имеющим соединение с землёй металлоконструкциям зданий, технологическим аппаратам и механизмам и металлическим корпусам электрооборудования.

К мероприятиям по предотвращению возможности поражения электрическим током относятся:

1. обеспечение недоступности токоведущих частей путем использования изоляции в корпусах оборудования;
2. применение средств коллективной защиты от поражения электрическим током;
3. применение защитного заземления;

4. применение защитного зануления;

5. применение защитного отключения;

6. использование устройств бесперебойного питания.

7. запрет на работы на задней панели при включенном сетевом напряжении;

8. проведение работ по устранению неисправностей только квалифицированным персоналом;

9. слежение за исправностью электропроводки.

В соответствии с ГОСТ 12.0.004-2015 [54] обучение и инструктаж по безопасности труда носит непрерывный многоуровневый характер.

Профессиональная подготовка персонала, повышение его квалификации, проверка знаний и инструктажи проводятся в соответствии с требованиями государственных и отраслевых нормативных правовых актов по организации охраны труда и безопасной работы персонала.

Электротехнический персонал до допуска к самостоятельной работе должен быть обучен приемам освобождения пострадавшего от действия электрического тока, оказания первой помощи при несчастных случаях.

Персонал, обслуживающий электроустановки, должен пройти проверку знаний Правил безопасности и других нормативно-технических документов (правил и инструкций по технической эксплуатации, пожарной безопасности, пользованию защитными средствами, устройства электроустановок) в пределах требований, предъявляемых к соответствующей должности или профессии, и иметь соответствующую группу по электробезопасности.

Персонал обязан соблюдать требования Правил безопасности, инструкций по охране труда, указания, полученные при инструктаже.

Инструктаж по характеру и времени проведения подразделяют на следующие виды:

- вводный инструктаж;
- первичный и повторный инструктажи на рабочем месте;
- внеплановый инструктаж;
- целевой инструктаж.

Выводы по разделу «Социальная ответственность»

В данном разделе были рассмотрены опасные и вредные факторы на рабочем месте разработчика информационной системы принятия решений. Были актуализированы действующие нормативы, их значения были сравнены с фактическими показателями рабочей зоны инженера-программиста (при такой возможности).

Было выявлено, что разрабатываемый объект не влияет на окружающую среду на этапах его разработки, внедрения и эксплуатации. Единственная вероятная ЧС, пожар, не может быть вследствие действия ИС на любых этапах ее функционирования. Так как рабочее место может быть в поле действия возгорания, были проанализированы правовые и организационные вопросы обеспечения пожарной безопасности, показавшие отсутствие нарушений касаемо рабочей области разработчика.

Проанализировав помещение для работы, можно сделать вывод, что оно соответствует необходимым требованиям.

Заключение

Для нахождения оптимального управления были рассмотрены эвристические алгоритмы, среди них: различные реализации генетического алгоритма, алгоритмы имитации отжига, а так же алгоритм полного перебора – поиск в ширину ($breath\ first\ search$).

Для решения задачи было смоделировано шесть моментов времени. На каждом моменте времени было задано три управляющих воздействия (без воздействия, с мексидолом, с сарбифером).

Каждый алгоритм настраивался для решения конкретной задачи:

- Для генетического алгоритма были подобраны способы селекции и скрещивания и установлен оптимальный размер популяции.
- Для каждого из рассматриваемых алгоритмов отжига (Больцмановский отжиг, отжиг Коши и быстрый отжиг) подбирались начальная и конечная температуры
- Для алгоритма полного перебора построена структура дерева, которая соответствует постановке задачи.

Таблица 26 – Результаты работы алгоритмов задачи оптимизации

Название алгоритма	Достижение оптимума	Число итераций	Время работы
BestRand+1pCrossing	да	146	0.240
BestRand+2pCrossing	да	290	0.301
BestRand+æCrossing	да	30	0.172
Roulette+1pCrossing	да	290	0.280
Roulette+2pCrossing	да	284	0.289
Roulette+æCrossing	да	55	0.196
Fast annealing	да	1600	0.148
Koshi annealing	да	1000	0.053
BFS−algorithm	да	−	0.04

Из таблицы результатов работы алгоритмов можно заключить, что алгоритм поиска в ширину является самым быстрым среди других рассматриваемых алгоритмов применительно к данной задаче. Второй по скорости – это отжиг коши. Меньше всего итераций для достижения оптимального решения у генетического алгоритма, который в качестве селекции использует «наилучший + случайно выбранный», а в качестве оператора кроссовера – «æCrossing».

Как видно, почти все рассматриваемые алгоритмы сошлись к оптимальному решению, за исключением Больцмановского отжига, который затратил самое большое число итераций и времени.

Если сравнивать генетические алгоритмы и алгоритмы имитации отжига с алгоритмом полного перебора, то нужно обращать внимание на множество допустимых решений. Размер структуры дерева зависит от двух параметров – количество управляющих воздействий и количество моделируемых моментов времени. Таким образом, чем больше этих параметров, тем шире становится множество допустимых решений и соответственно структура дерева будет разрастаться и места в памяти компьютера для нее может не хватить.

Из сказанного выше можно сделать следующее заключение:

Если необходимо найти точное решение задачи и множество допустимых решений ограничено и относительно невелико, то рекомендуется применять алгоритм поиска в ширину для полного перебора, который займет мало времени для работы. С другой стороны, если задача более трудоемкая или $NP-$ трудная, то следует применить эвристические алгоритмы, среди которых в данной работе хорошо показали себя генетический алгоритм, который в качестве селекции использует «наилучший + случайно выбранный», а в качестве оператора кроссовера – «æCrossing» и отжиг Коши.

За счет рассмотренных методов для каждого пациента рассматриваемой грцппы было найдено оптимальное управляющее воздействие. В результате применения этого управляющего воздействия вероятность возникновения риска перехода в неблагоприятное состояние пациента снизился на 10%.

Список использованных источников

1. Всемирная организация здравоохранения. Международная классификация функционирования, ограничения жизнидеятельности и здоровья детей и подростков: доклад, - 2015 г.
2. Всемирная организация здравоохранения.Инструмент для оценки качества стационарной помощи матерям и новорожденным. Доклад, - 2009.
3. Бокучаева Н.В., Мамасахлисов Г.В. К вопросу применения методов термодинамики и информационной статистики в биологии// Сообщения АН ГССР. – 1985.
4. Фокин В.А. Критерий оценки состояния сложных иосистем// Известия Томского политехничесеого университета, - 2004.
5. Фокин В.А. технология интегральной оценки состояния биомедицинских систем// системы управления и информационные технологии. – 2008.
6. Ф.Харри. Теория графов. // Москва, 1973
7. Кочетов Ю.А. Вероятностные методы локального поиска для задач дискретной оптимизации // Дискретная математика и ее приложения: Сборник лекции молодежных научных школ по дискретной математике и ее приложениямю – М.: Изд-во центра прикладных исследований при механико-математическом факультете МГУ, 2001. С84-117.
8. Кочетов Ю.А. Введение в исследование операции: учебное пособие. Новосибирск: Новосибирский государственный университет, 2005.
9. А.Б. Барский Нейронные сети: распознавание, управление, принятие решений – М.: Финансы и статистика, 2004.
10. А.С. Лопатин Метод отжига – Санкт-Петербургский государственный университет, 2005.
11. Завьялов Ю.С., Квасов Б.И., Мирошенко В.Л. Методы сплайн – функции. – Мщсква: Наука, 1980.
12. С. Хайкин Нейронные сети: Полный курс – М.: Вильямс, 2006г.
13. Рассел, Норвиг. Искусственный интеллект. Совремменный подход. 2-е изд. 206 год.
14. Батищев, Д.И. Генетические алгоритмы решения экстремальных задач. – Нижний Новгород: 1995 год.
15. Турчак Л.И. Основы численных методов: Учебное пособие – М.: Наука. Гл. Ред. Физ-мат. Лит., 1987 год.

16. Вержбицеий В.М. Основы численных методов: Учебник для вузов – М.:Высш. Шк., 2002. – 730с.
17. Матренин П.В. Методы стохастической оптимизации: учебное пособие. – Новосибирск: Изд-во НГТУ, 2016.
18. Граничин О.Н. Введение в методы стохастической оптимизации и оценивания: учебное пособие. – Санкт-Петербургский Государственный Университет. 2003 год.
19. С.Э. Мастицкий, В.К. Шитиков: Статистичиский анализ и визуализация данных с помощью R. – 2004 год.
20. ГОСТ 12.0.003-74. ССБТ. Опасные и вредные производственные факторы. Классификация.
21. ГОСТ 12.1.003–83 ССБТ. Шум. Общие требования безопасности.
22. СанПиН 2.2.2.548-96. Гигиенические требования к микроклимату производственных помещений.
23. СанПиН 2.2.1/2.1.1.1278–03. Гигиенические требования к естественному, искусственному и совмещённому освещению жилых и общественных зданий.
24. ГОСТ 12.1.045–84 ССБТ. Электростатические поля. Допустимые уровни на рабочих местах и требования к проведению контроля.
25. ГОСТ 12.1.038–82 ССБТ. Электробезопасность. Предельно допустимые уровни напряжений прикосновения и токов.
26. ГОСТ 12.1.004-91 ССБТ. Пожарная безопасность. Общие требования.
27. ГОСТ 12.1.010-76 ССБТ. Взрывобезопасность. Общие требования.
28. СанПиН 2.2.2/2.4.1340–03. Санитарно-эпидемиологические правила и нормативы «Гигиенические требования к персональным электронно-вычислительным машинам и организации работы».
29. ГОСТ 12.2.032-78 ССБТ. Рабочее место при выполнении работ сидя. Общие эргономические требования.
30. Действия ЭМП на организм человека. Сборник лекций. [Электронный ресурс]. Режим доступа: http://studopedia.net/13_166967_deystviya-emp-na-organizm-cheloveka.html, свободный. Дата обращения: 19.03.2018 г.
31. ГОСТ 12.4.011-89 ССБТ. Средства защиты работающих. Общие требования и классификация
32. Технологический регламент обращения с ртутьсодержащими отходами. [Электронный ресурс]. Режим доступа: http://eco-profi.info/index.php/othod/instr/601-instr-3533010013011-3.html, свободный. Дата обращения: 10.03.2018 г.

Приложение А
(обязательное)

Основные методы при решении задачи (The main methods in solving the problem)
Английский язык

Студент:

Группа	ФИО	Подпись	Дата
8БМ61	Кисатов Марат Александрович		

Консультант проф. кафедры:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Гергет Ольга Михайловна	к.т.н		

Консультант – лингвист кафедры ИЯИК:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Комиссарова Ольга Валентиновна	к.ф.н.		

2.1 Integral indicator

One of the promising approaches to solve the problem of assessing the state of the bio-system (human body) is the use of entropy methods of modeling complex systems.

This approach to the analysis of the state of the bio-system allows to avoid a lot of such properties of the bio-system as:

- the non-linearity of biological systems;
- incomplete descriptions of the state of the bio-system;
- large variability and heterogeneous metric of bio-system parameters;
- the state of the bio-system cannot be described by the monotone function;
- Bio-systems are subject to a large number of conditions.

In this work, to assess the degree of deviation of the state of the bio-system from the level of normal functioning, two criteria are considered: the criterion based on the information Kulbak measure and the criterion using Mahalanobis metric.

2.1.1 The integrated indicator on the basis of information of the Kulback measure

The approach proposed by Bokuchaeva and Mamasakhlisov is considered, which is to consider the information measure of Kulbak as a measure of preference for the behavior of the bio-system:

$$I(t) = \int_t P_0(x) Ln \frac{P_0(x)}{P(x,t)} dx \,,$$

where $\boldsymbol{P_0(x)}$ – probability density of the current "equilibrium" state;

$\boldsymbol{P(x,t)}$ – density of probability of finding a biological system in the state x_t;

$\boldsymbol{x(t) = (x_1, \dots, x_n, t)}$ – time-dependent variables characterizing the state of the biological system.

This indicator allows to estimate the deviation (4) of the current state of the patient (1) from the “preferred” state (3). Taking as the "preferred" state of the bio-system the state in which the values of all variables of the bio-system state are given

to the mean square values of the physiological norm, the following expression can be used as a criterion for assessing the current state of the bio-system:

$$I = \frac{1}{n}\sum_{j=1}^{n} P_{0j} Ln \frac{P_{0j}}{P_j},$$

where $\boldsymbol{P_{0j}}$ – "preferred" probability of the condition;

$\boldsymbol{P_j}$ – the probability that the value of the state variable X corresponds to the “norm”.

n – the number of indicators characterizing the state of the bio-system.

The formula for the integral index assumes that the "preferred" probability of the state P_{0j} is one. Thus, the expression for I takes the form:

$$I = \frac{1}{n}\sum_{j=1}^{n} Ln \frac{1}{P_j},$$

where $\boldsymbol{P_j}$ is expressed in the following form:

$$P_j = P\left(\left|x_j - x_0\right| < \delta\right) = 2\Phi\left(\frac{\delta}{\sigma}\right) - 1,$$

Φ – Laplace function;

$\boldsymbol{\delta}$ – the value of deviation from the average value of this characteristic:

$$\delta = \left|x_j - x_{\text{норм}}\right|$$

$\boldsymbol{\sigma}$ – standard deviation of the characteristic:

2.2 Genetic algorithm

A genetic algorithm is a heuristic search algorithm. This algorithm is used to solve optimization and simulation problems by random selection, combination and variation of the required parameters using mechanisms similar to natural selection in nature. The genetic algorithm uses natural evolution techniques such as inheritance, mutation, selection, and crossover. A distinctive feature of the genetic algorithm is only the emphasis on the crossing operator, which performs the operation of recombination of candidate solutions, the role of which is similar to the role of crossing in nature.

Due to the fact that this algorithm is self-learning, the range of its application is very wide:

- Graph problems (traveling salesman problems, coloring, finding matching pairs)
- Build tasks
- Scheduling
- Game strategy
- Approximation of functions
- Bioinformatics
- Assignment tasks
- Function optimization
- Configuration and training of artificial neural networks

Genetic algorithms operate on a set of individuals (population). Each individual is a vector that encodes one of the solutions to the problem.

$$u = [u_1, u_2, \dots, u_n]$$

This vector u is called the chromosome, and its components $u_i -$ genes.

Each chromosome has a function of fitness. This function allows us to understand how good the solution is. Due to this function, individuals are divided into two types:

- The most adapted (the most suitable solutions). These individuals remain in the population and have the opportunity to participate in breeding and to give offspring.
- The least adapted (bad decisions), are gradually removed from the population (die) and do not give offspring.

Thus, there is a natural selection, as a result of which the adaptability of the new generation all the time on average becomes higher than the previous one.

It is assumed that some function is specified $f(u)$, where $u \in U$. The set U of discrete, consists of a finite number of elements:

$$U = \bigcup_{s \in S} u_s \ , S \subset R$$

It is necessary to find such an element u of the set U, which would minimize the original function:

$$\min_{u \in U} f(u)$$

Function f can be considered as a function of one variable $(f: R^1 \to R)$, or a function of several variables $(f: R^n \to R)$.

In the problem under consideration, the dimension of the function f is depend on the number of considered moments of time, i.e. each element of the set of admissible solutions U consists of six components.

Thus, as a function of f is considered a functional $J(x; u)$,
where $u = (u_1, \dots, u_6)$, $u_i -$ control action at time i.

3. Select the initial population P and remember the record $F^* = \min_{i=1,\dots,k} f(S_i)$
   - 3.1. Until the stop criteria is met, do the following:
   - 3.2. To Select "parents"S_{i_1}, S_{i_2} from the population.
   - 3.3. To apply to S_{i_1}, S_{i_2} the operator of the crossing and get a new solution S'.
   - 3.4. To apply to S' mutation operator and get the solution S''.
   - 3.5. To apply to S'' the operator of the local improvement and to get a new solution S'''.
   - 3.6. If $f(S''') < F^*$, then change the record $F^* := f(S''')$.
   - 3.7. To add S''' to the population and remove the worst solution

The formation of the initial population consists in the selection of solutions (individuals) from the set of the admissible solutions U. The size of the population is also determined. The result of the genetic algorithm depends on the choice of the initial population and its size. Elements from the set U are entered into the population arbitrarily.

The algorithm of formation of a population of n elements:

2. Repeat n times:
   1.4. Let $i = 1$.
   1.5. To repeat m times:
      1.5.1. To make a random uniformly distributed value $\xi(u_i) \in S$, $i = 1 \ldots m$.
      1.5.2. To select the corresponding element u_{i_ξ}, which is the $i-$ the gene of the solution corresponding to the implementation ξ.
      1.5.3. To increase the sequence number of a gene per unit: $i := i + 1$.
   1.6. To record the resulting chromosome $u = \left[u_{1_\xi}, \ldots, u_{n_\xi}\right]$ in the population.

$n-$ population size;

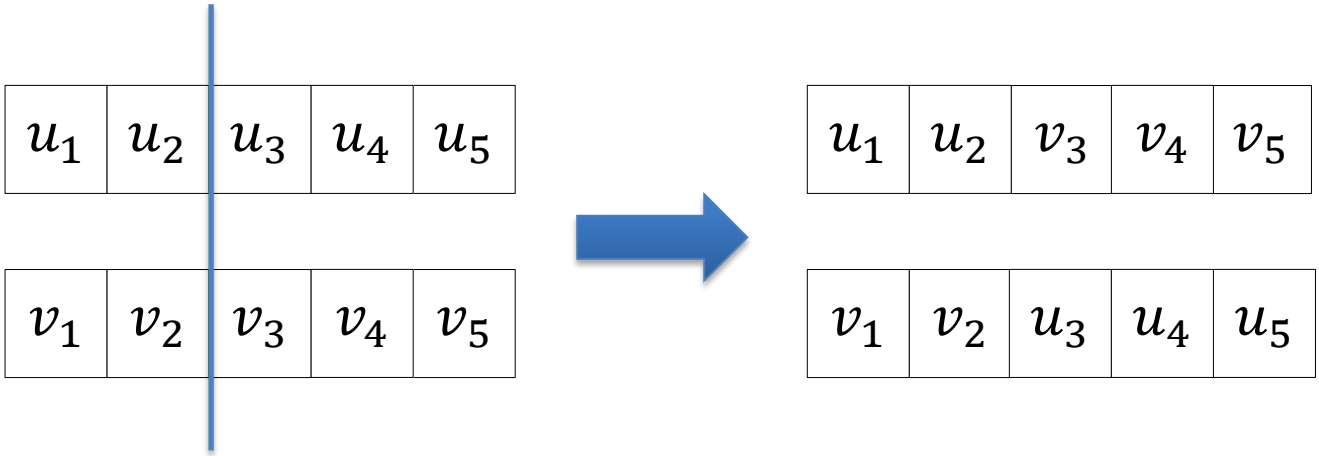
$m-$ the length of the chromosome;

The crossing operator is the crossing of two objects. There are two solutions $u^1, u^2 \in U$ and a procedure that allows an algebraic operation $u^1 æ_d u^2$, where d belongs to a certain family of indices ($d \in D \subset R$). In the genetic algorithm, the crossing operation is considered parametrically (parameter d regulates the method of crossing) in contrast to the reproduction of biological individuals, where half of the genes are taken from the father, and the other half from the mother. In this paper, the parameter $d-$ is a random number from 1 to m.

There are different variations of the crossing operator. The paper considers:

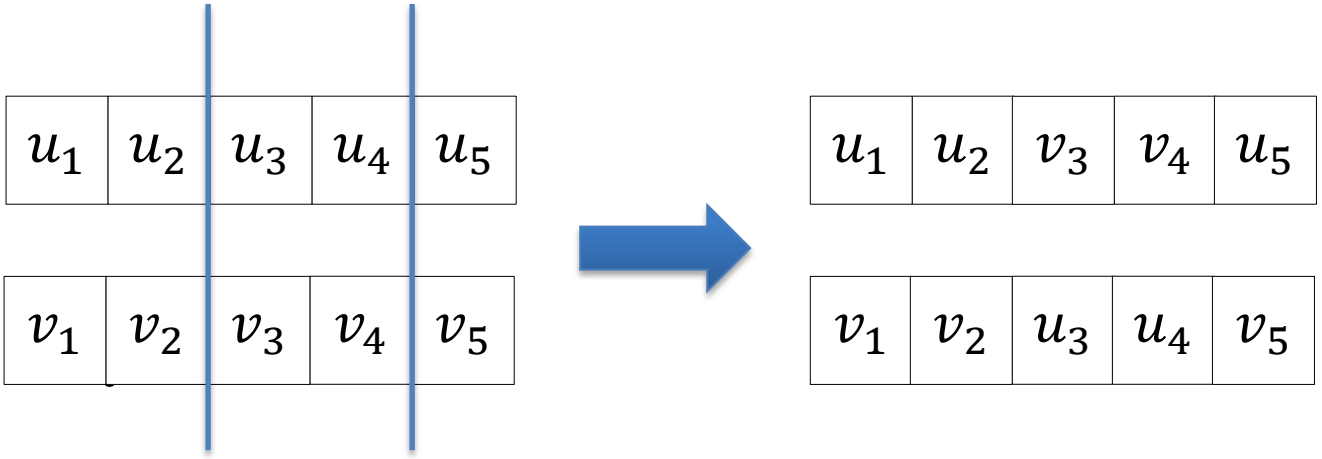
- **Single point crossing operator**
  Parameter d o of the crossing operator is chosen randomly. The result of the $æ_d$ operator is two children. One of these descendants appropriates part of the genes of the first parent, which is to the left of the separator d, and part of the genes of the second parent, which is the right of the delimiter d. Another descendant, on the contrary, the first part takes genes from the second parent,

the remaining part of the first. Thus, the formation of two offspring. Each of these descendants has the same chance of being selected ($p = 0.5$).



- **Two−point crossing operator**

  two separators are randomly selected. Just as in the case of the single-point crossing operator, the result $æ_d$ will be two individuals. One individual will take part of the genes from the first parent before the first separator and after the second separator, the other part – from the second parent between the separators. The second individual will take part of the genes from the second parent before the first separator and after the second separator, the remaining part – from the first parent between the separators. Next, one of the two descendants is chosen with equal probability.



- **The operator of crossing-dependent parents**

  Another variant of the crossing operator that is similar to the single-point operator is considered. However, unlike the single-point operator, the separator is not chosen randomly, but according to the following formula:

$$k = \frac{F(u^l)}{F(u^l) + F(u^r)} \in (0,1)\,,$$

where $u^l -$ first parent;

$u^r -$ second parent;

$F -$ the fitness function;

k belongs to the interval from 0 to 1.

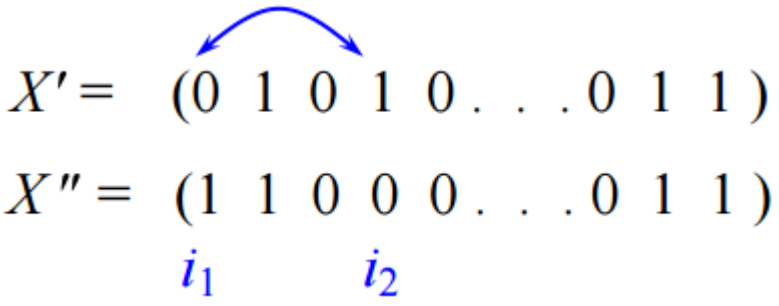
The value of the separator d is calculated as follows: the interval $(0,1)$ is divided into six equal parts. The parameter d is equal to the ordinal number of the part of the interval $(0,1)$, which contains the value k.

The mutation operator assumes the random change of genes in the solution. This operator is a mapping from a set of valid solutions to a set of valid solutions:

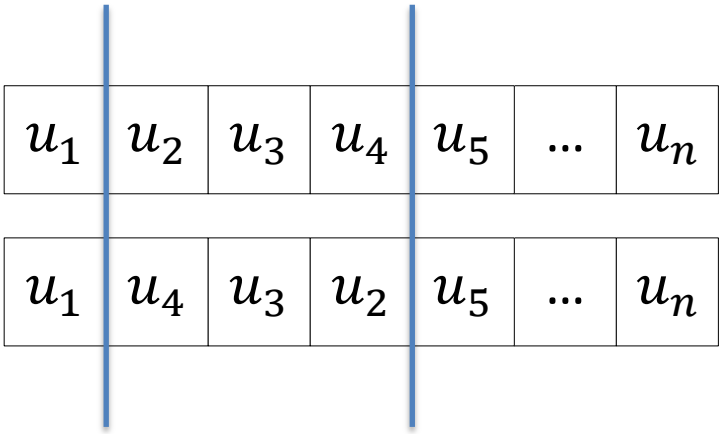
$$g_m : U \to U\,, m \in M$$

The mutation operator is probabilistic. With a certain probability, the initial solution, which is subjected to mutation, can change both for the better (increased adaptability) and for the worse (adaptability will decrease). Also, with the help of operator, the genetic algorithm is protected from failure to a local extreme (local minimum). The mutation operator can be implemented in the following ways:

- with a low probability $p < \frac{1}{n}$ in each coordinate, the value $u_i \in \{0,1\}$ is replaced by the opposite $1 - u_i$.
- If the solution requires saving $\sum_{i \in I} u_i = p$, then the coordinate i_1 is chosen randomly, that $u_{i_1} = 1$ and the coordinate i_2, such that $u_{i_2} = 0$ and replacement $u_{i_1} := 0,\ u_{i_2} := 1$ is made.



The inversion operator works with one chromosome. This operator changes the sequence of genes in the chromosome between two boundaries. In this paper, the inversion boundaries are chosen randomly:



Selection of chromosomes consists in the choice (by the calculated values of the fitness function) of those chromosomes that will participate in the creation of descendants for the next population, i.e. for the next generation. This choice is made according to the principle of natural selection, in which the greatest chances of participation in the creation of new individuals have chromosomes with the greatest importance of fitness function. The paper considers the following types of selection:

- **Roulette method**
  Chromosomes are selected by running a tape measure. Roulette is divided into sectors that correspond to individual chromosomes. Moreover, the value of the sector depends on the fitness function of the individual and in the case of solving the minimization problem can be calculated by the following formula:

$$P_{sel}(i) = \frac{1}{N-1}\left(1 - \frac{f(i)}{\sum_{i=1}^{n} f(i)}\right)$$

This formula is suitable for the minimization problem. The size of each sector is indicated as a percentage. The value of the whole roulette – 100%.

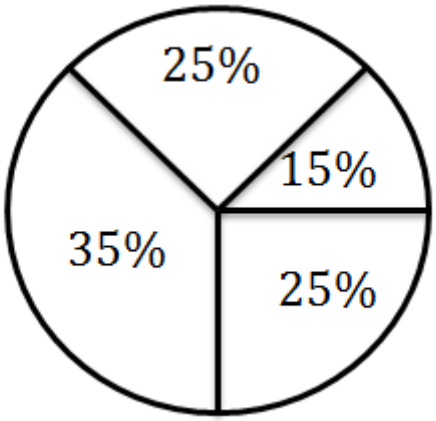
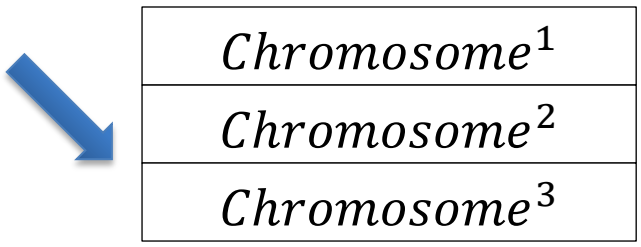


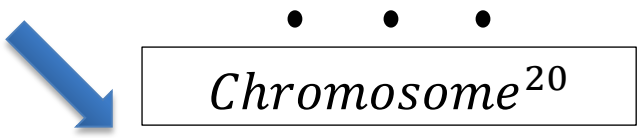
Figure A.1 –relative positions of objects in areas

With this selection, members of the population with higher adaptability have a larger sector and, accordingly, a greater chance of being selected than those chromosomes, the function of which is minimal.

- **Proportional selection**:
  Two parents are randomly selected from the population. For the solution S_i the probability of being chosen is inversely proportional to the value of the objective function $f(S_i)$. Parents are selected for crossing as long as they are not different.

- **Randomly selected + most remote from it:**
  The chromosome (first parent) is randomly selected from the population. Next, select the chromosome that takes the most remote position from the first.





- **Best + randomly selected**
  As the first parent selects the individual that has a record of the fitness function (minimum value of fitness-function). The second parent is chosen at random. To avoid degeneration (two identical parents), the second chromosome must be different from the first.

Let us consider the question related to the criterion of stopping the genetic algorithm. There are various options for stopping criteria: the algorithm achieves a certain number of iterations;

6. Expiration of the time allowed for the algorithm to work;
7. Achieving the optimal solution;
8. The convergence of the population.

You can set the number of iterations for the algorithm to work. Then there is a possibility that the algorithm will not be able to find the optimal solution simply for the number of generations given to it. On the other hand, the algorithm may need much less iteration to find the optimum and the rest of the iterations it will work idle. The same can be said about such a criterion as a stop time job. If the optimal solution of the problem is known in advance and it is necessary to check which of the variants of the genetic algorithm copes with the task faster, then you can use the stopping criterion at number 3. To solve the problems of finding an extreme optimal solution is unknown in advance. I would like to understand when to stop the genetic algorithm. This chapter is discussed the last version of the stopping criterion.

Under the convergence is understood to mean the state of the population when neither the operation of crossover or mutation operation do not alter the genetic diversity of the population within a few generations.

With this interpretation of the convergence of the population, the following problems arise:

3. It is unknown how many generations are sufficient to track the population's immutability;
4. There is no clear definition of the converged population.

Therefore, this approach is unacceptable for optimization problems. Therefore, it is proposed to use a metric that assesses the difference of individuals in the population as a stopping condition. Each individual in the population is an acceptable solution and is described by its genotype and phenotype. Genotype is the recording of a solution in coded form (using 0 and 1), and phenotype is the decoded solution. Based on this description, it is possible to determine two metrics with which to assess the difference between individuals, for the genotype and phenotype. For comparison of genotypes of individuals used Hemming distance, which is equal to the number of different positions in chromosomes. For comparison of phenotypes of individuals used Euclidean distance.

In this paper, we consider the Hemming distance, which is given by the formula:

$$\rho_H(X_i, X_j) = \sum_{l=1}^{m} |x_{il} - x_{jl}|$$

$\rho_H(X_i, X_j)$ equal to the number of mismatches of the values of the corresponding features in X_i и X_j.

A characteristic population metric is introduced that will assess the difference of all individuals in the population. This metric is calculated using the following formula:

$$\bar{e} = \frac{1}{N}\left(\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\sum_{l=1}^{m}|x_{il} - x_{jl}|\right)\right)$$

However, if you calculate the distance at each iteration, the algorithm will be quite demanding on computing resources, so it is proposed to use the absolute value of the difference between the average adaptability of the population:

$$\left|f_{\text{ср}}^{l+1} - f_{\text{ср}}^{l}\right|$$

If this value does not exceed the predetermined small ε or is equal to zero, then we proceed to the calculation of the distance between individuals of the population to calculate the characteristic metric of the population. Thus, the stopping criterion consists of two levels of checks.

Distance and characteristic metric calculations can be parallelized using, for example, the OpenMP library for C++.

2.3 The algorithm of simulation of annealing

The next method of control action selection (2), which is considered in this paper, is the simulation algorithm annealing (SA-method). This method is designed to find the optima of functions defined on different structures. Like the genetic algorithm, this method refers to heuristic methods of finding the optimum. Therefore, the annealing method may not always give a guaranteed result. On the other hand, it allows you to solve NP - complete problems in a fairly small number of steps.

The problem statement for the annealing simulation method is no different from the problem statement for the genetic algorithm and is as follows:

The set of admissible solutions U is given. It is assumed that this set is not empty. Also on this set U some function is defined $f(u), u \in U$ (that is, each element of the set U is mapped to a number). You want to find the minimum of this function $f(u)$, so it is assumed that

$$\inf_{u \in U} f(u) > -\infty$$

That is, the function $f(u)$ must be bounded from below (if $f(u)$ is not bounded from below, then the problem of finding the minimum becomes meaningless). Moreover, without limiting generality, it is assumed that the function f on the set U is non-negative: $f(u) \geq 0 \; \forall u \in U$.

You want to find such an element $u^* \in U$ that $f(u^*) = \min_{u \in U} f(u)$.

To design the method of simulated annealing, you will need to introduce some concepts:

$S -$ is a set of indices (the set is finite or countable). We consider the random variable ξ, which takes values from the set S. Also we consider the set of transformations of the set U in itself:

$$g_s : U \to U, \qquad s \in S$$

It is assumed that g_s is a small transformation:

$$\exists \, M > 0 : \; \forall s \in S \; \rho(g_s(u), u) \leq M$$

where $\rho -$ is some metric on the set$S$.

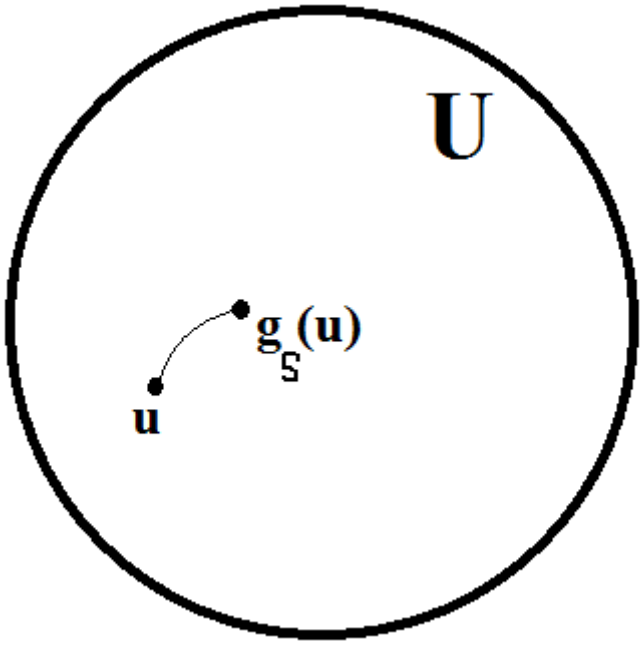


Figure A.2 – transformation of a function on a set of valid solutions

The SA−method uses a map, which is commonly called temperature:

$$T : S \to R$$

this function maps the iteration number to the corresponding real number. This function must be decreasing.

Another mapping is introduced, which is called the energy function:

$$E : U \to R$$

Energy function to each element of the set of feasible solutions U puts in line for some real numbers.

The algorithm of the SA−method is given below:

5. $k = 0$
6. Maximum and minimum temperatures are set.
7. The initial state is set: $u_0 \in U$
   7.1. Until the temperature reaches the minimum value must be made
   7.2. $\tilde{u} = g_{\xi}(u_k)$, где ξ − random variable

7.3. $\Delta = f(\tilde{u}) - f(u_k)$

7.4. If $\Delta \leq 0$ then $u_{k+1} = \tilde{u}$

7.5. If $\Delta > 0$ then $\eta \sim R(0,1)$ – random variable. If $\eta < \exp\left\{-\frac{\Delta}{t_k}\right\}$, then $u_{k+1} = \tilde{u}$ else $u_{k+1} = u_k$

7.6. $k = k + 1$

7.7. $t_{k+1} = T(k)$

t_{max} is chosen arbitrarily and must be commensurate with the function to be minimized. The initial state u_0 is also selected randomly (either a random value or a fixed value can be selected).

A distinctive feature of this algorithm from gradient algorithms is that it can't do optimal steps (make the worst decisions), while the last worst decisions are always discarded. Because the SA–algorithm takes the worst decisions with a certain probability, it is called stochastic (probabilistic). This feature is to move not only in the direction of improving the solution, but also to retreat a little back, protects the algorithm from failure to the local minimum.

As in the case of genetic algorithms, the SA–algorithm also has many different implementations, which differ from each other in the way of generating new States and the function of temperature change (decrease).

The probability transition to the worst solution is carried out according to the following principle: a random number is generated in the range from 0 to 1. If this number is in the interval $\left[0,\ \exp\left\{-\frac{\Delta}{T_k}\right\}\right]$, that decision is made (the algorithm steps back). Otherwise, the algorithm remains in place. The following conclusion can be made: the higher the temperature, the greater the exponent $\exp\left\{-\frac{\Delta}{T_k}\right\}$. However, the temperature decreases with time and it remains high only in the first iterations. Therefore, the algorithm has the ability to "walk" through the set of feasible solutions at the first iterations. Over time, the probability of making the worst decision decreases in accordance with the chosen law of temperature change and the algorithm is increasingly forced to make only the best decisions.

Приложение Б

```
void Spline (float* c, float * h, float * x, float * y, float S0, float Sn, int N)
```

```
	{

		float * P = new float[N+1];
		float * Q = new float[N+1];
		float * T = new float[N+1];
		float * F = new float[N+1];
		float * alf = new float[N+1];
		float * bet = new float[N+1];
		int i;


		c[N] = Sn;
		for (i = 1; i <=N; i++) {
			h[i] = x[i] - x[i-1];
		}



		for (i = 1; i <= (N-1); i++) {
			P[i] = h[i];
			Q[i] = 2*(h[i]+h[i+1]);
			T[i] = h[i+1];
			F[i] = 6*((y[i+1]-y[i])/h[i+1] - (y[i]-y[i-1])/h[i]);
		}

		alf[1] = 0;
		bet[1] = S0;

		for (i = 1; i <= (N-1); i++) {
			alf[i+1] = -T[i]/(P[i]*alf[i] + Q[i]);
			bet[i+1] = (F[i] - P[i]*bet[i])/ (P[i]*alf[i] + Q[i]);
		}

		for (i = (N-1); i >= 0; i--) {
			c[i] = alf[i+1]*c[i+1] + bet[i+1];
		}




	}

float ReturnSpline (float iks) {

	int number = 0;
	int i = 1;

	while ((number == 0) && (i <= N)) {
		if ((iks >= x[i-1]) && (iks <= x[i])) number = i;
		else
			i++;

	}




	float K, L, M, S;

	K = (y[i-1] * (x[i] - iks))/h[i];
	L = (y[i] * (iks - x[i-1]))/h[i];
```

```
	M = (c[i-1] * ((x[i] - iks)*(x[i] - iks) *(x[i] - iks) - h[i]*h[i]*(x[i] -
iks)))/(6.0*h[i]);
	S = (c[i] * ((iks - x[i-1])*(iks - x[i-1]) *(iks - x[i-1]) - h[i]*h[i]*(iks - x[i-
1])))/(6.0*h[i]);

		return K+L+M +S;
}




void DrowSpline  ( float * c, float * h, float * x, float * y, int N)
{
	float f;
	FILE* g = fopen("C:\\Users\\1\\Desktop\\rp.txt","w");

	for (float i = x[0]; i <= x[N]; i = i + 0.01)
	{
		f = ReturnSpline ( c, h, x, y, i, N);
		fprintf(g, "%f", i);
		fprintf(g, "%s", "    ");
		fprintf(g, "%f\n", f);
	}

fclose(g);
}
```

Приложение В

```
double lp (double x) {

	return (erf (x/sqrt(2.0)))/2.0;
}

double probability ( double delta, double sigma) {

    return 2.0 * lp (delta/sigma);
}


void alternativeIntegral (ifstream &fread, ofstream &fwrite, ifstream &fnormal) {

	double A[N+1][m+1];
	double integral[N+1];


	double del[N+1][m+1];
	double sigma[m+1];
	double xNormal[m+1];
	double disper[m+1];
	double sred[m+1];



	for (int i = 1; i <= N; i++)
	{
		integral[i] = 0;
	}

	for (int j = 1; j <= m; j++)
```

```
	{
		sigma[j] = 0;
		xNormal[j] = 0;
		disper[j] = 0;
		sred[j] = 0;
	}


	for (int i = 1; i <= N; i++)
	{
		for (int j = 1; j <= m; j++)
		{
			fread >> A[i][j];
		}
	}
	fread.close();

	for (int j = 1; j <= m; j++) { // считаем  xNorm признака j-ого
		fnormal >> xNormal[j];
	}
	fnormal.close();

	for (int i = 1; i <= N; i++) { // считаем  delta
		for (int j = 1; j <= m; j++)
		{
				del[i][j] = abs(xNormal[j] - A[i][j]);
		}
	}


	for (int j = 1; j <= m; j++) {
//=================================================================
			for (int i = 1; i <= N; i++)
			{
				sred[j] = sred[j] + A[i][j]/N;
			}
		}



	for (int j = 1; j <= m; j++) { // считаем  дисперсию признака j-ого
		for (int i = 1; i <= N; i++)
		{
			disper[j] = disper[j] + (  (sred[j] - A[i][j]) * (sred[j] - A[i][j])  )
/ N;
		}
	}

	for (int j = 1; j <= m; j++)
	{
		sigma[j] = sqrt(disper[j]);
	}

	for (int i = 1; i <= N; i++) { // считаем  delta
		for (int j = 1; j <= m; j++)
		{
			double arg = 1/probability(del[i][j], sigma[j]);
			integral[i] = integral[i] + log(arg )/m;
		}
	}

for (int i =1; i <= N; i++) {
```

```
		fwrite << integral[i] << endl;
}
//std:: cout << integral[1];
fwrite.close();

}
```

Приложение Г

```
class Annealing {
public:
	Annealing(double _tmax, double _tmin, double (*foo)(std::vector<int> solution), int
_dimension) : tmax(_tmax), tmin(_tmin), energy(foo), dimension(_dimension)  {}

protected:
	std::vector<int> setInit();
	virtual std::vector<int> generateSolution(std::vector<int> prevSolution);
	virtual std::vector<int> generateSolution(std::vector<int> prevSolution, double
temperature);
	virtual double temperature(int iteration) = 0;
	virtual bool isChange(double delta, double temperature) = 0;

public:
	virtual void start();

protected:
	double (*energy)(std::vector<int> solution);
	double tmax;
	double tmin;
	int dimension;
};


class BasicAnnealing : public Annealing {
public:
	BasicAnnealing(double _tmax, double _tmin, double (*foo)(std::vector<int> solution),
int _dimension) : Annealing(_tmax, _tmin, foo, _dimension)  {}

private:
	std::vector<int> generateSolution(std::vector<int> prevSolution) override;
	double temperature(int iteration) override;
	bool isChange(double delta, double temperature) override;
};


class BolcmanAnnealing : public Annealing {
public:
	BolcmanAnnealing(double _tmax, double _tmin, double (*foo)(std::vector<int>
solution), int _dimension) : Annealing(_tmax, _tmin, foo, _dimension)  {}
	void start() override;
private:
	std::vector<int> generateSolution(std::vector<int> prevSolution, double temperature)
override;
	double temperature(int iteration) override;
	bool isChange(double delta, double temperature) override;
};
```

```
class VeryFastAnnealing : public Annealing {
public:
	VeryFastAnnealing(double _tmax, double _tmin, double (*foo)(std::vector<int>
solution), int _dimension) : Annealing(_tmax, _tmin, foo, _dimension)  {}
	void start() override;
private:
	std::vector<int> generateSolution(std::vector<int> prevSolution, double temperature)
override;
	double temperature(int iteration) override;
	bool isChange(double delta, double temperature) override;
};
```

```
double normDistribution() {
	int n = 24;
	double sum = 0.0;
	for (int i = 1; i <= n; i++) {
		sum += 0 + double(rand()) / RAND_MAX * 1;
	}
	double s = sqrt( double(n/12) );
	return sum / s;
}

std::vector<int> BasicAnnealing::generateSolution(std::vector<int> prevSolution) {
	double prob = 0 + double(rand()) / RAND_MAX * 1;
	int pos;
	if (prob > 0.5)
		pos = 0 + rand() % 3;
	else
		pos = 3 + rand() % 3;
	int newComponent = 0;
	switch (prevSolution[pos]) {
		case 0:
			newComponent = 1 + rand() % 2;
			break;
		case 2:
			newComponent = 0 + rand() % 2;
			break;
		case 1:
			{
				if ( ( 0 + double(rand()) / RAND_MAX * 1) > 0.5)
					newComponent = 2;
				else
					newComponent = 0;
				break;
			}
	}

	prevSolution[pos] = newComponent;
	return prevSolution;
}

double BasicAnnealing::temperature(int iteration) {
	return tmax/iteration;
}

std::vector<int> Annealing::generateSolution(std::vector<int> prevSolution) {
	return prevSolution;
}
```

```
std::vector<int> Annealing::generateSolution(std::vector<int> prevSolution, double
temperature) {
	return prevSolution;
}

std::vector<int> Annealing::setInit() {
	std::vector<int> solution;
	for(int i = 0; i < dimension; i++) {
		solution.push_back(0 + rand() % 3);
	}

	return solution;
}

bool BasicAnnealing::isChange(double delta, double temperature) {
	double prob = ( 0 + double(rand()) / RAND_MAX * 1);
	double h = exp(-delta/temperature);
	return (prob <= h);
}




void Annealing::start() {
	StreamWriter * writer = new
StreamWriter("C:\\Users\\Maratell\\Desktop\\annealing.txt");
	std::vector<int> S2 = setInit();
	double t = tmax;
	int k = 0;

	while (t > tmin) {
		k++;
		std::vector<int> S1 = generateSolution(S2);
		double e1 = energy(S2);
		double delta = energy(S1) - energy(S2);
		if (delta <= 0)
			S2 = S1;
		else if (isChange(delta, t))
			S2 = S1;
		t = temperature(k);


		double h;
		if (delta <= 0)
			h = 0.0;
		else
		{
			double f = ( 0.3 + double(rand()) / RAND_MAX * 0.4);
			h = exp(-f/t);
		}
		writer->write( h );
		//std::cout<<energy(S2) << std::endl;
	}
}


double BolcmanAnnealing::temperature(int iteration) {
	return tmax/log(1 + double(iteration) );
}
```

```
bool BolcmanAnnealing::isChange(double delta, double temperature) {
	double prob = ( 0 + double(rand()) / RAND_MAX * 1);
	double h = exp(-delta/temperature);
	return (prob <= h);
}

std::vector<int> BolcmanAnnealing::generateSolution(std::vector<int> prevSolution, double
temperature) {
	double prob = 0 + double(rand()) / RAND_MAX * 1;
	int pos;
	if (prob > 0.5)
		pos = 0 + rand() % 3;
	else
		pos = 3 + rand() % 3;
	int newComponent = 0;
	switch (prevSolution[pos]) {
		case 0:
			newComponent = 1 + rand() % 2;
			break;
		case 2:
			newComponent = 0 + rand() % 2;
			break;
		case 1:
			{
				if ( ( 0 + double(rand()) / RAND_MAX * 1) > 0.5)
					newComponent = 2;
				else
					newComponent = 0;
				break;
			}
	}

	prevSolution[pos] = newComponent;
	return prevSolution;
}

void BolcmanAnnealing::start() {
	StreamWriter * writer = new
StreamWriter("C:\\Users\\Maratell\\Desktop\\annealing.txt");
	std::vector<int> S2 = setInit();
	double t = tmax;
	int k = 0;

	while (k < 10000) {
		k++;
		std::vector<int> S1 = generateSolution(S2, t);
		double e1 = energy(S2);
		double delta = energy(S1) - energy(S2);
		if (delta <= 0)
			S2 = S1;
		else if (isChange(delta, t))
			S2 = S1;
		t = temperature(k);


		double h;
		if (delta <= 0)
			h = 0.0;
		else
		{
			//double f = ( 0.1 + double(rand()) / RAND_MAX * 0.2);
			h = exp(-delta/t);
```

```
			}
			writer->write(  k );
		}
}

////////////////////


double VeryFastAnnealing::temperature(int iteration) {
		return tmax * exp( double(-0.01*iteration) );
}

bool VeryFastAnnealing::isChange(double delta, double temperature) {
		double prob = ( 0 + double(rand()) / RAND_MAX * 1);
		double h = exp(-delta/temperature);
		return (prob <= h);
}

void VeryFastAnnealing::start() {
		StreamWriter * writer = new
StreamWriter("C:\\Users\\Maratell\\Desktop\\annealing.txt");
		std::vector<int> S2 = setInit();
		double t = tmax;
		int k = 0;

		while (t > tmin) {
			k++;
			std::vector<int> S1 = generateSolution(S2, t);
			double e1 = energy(S2);
			double delta = energy(S1) - energy(S2);
			if (delta <= 0)
				S2 = S1;
			else if (isChange(delta, t))
				S2 = S1;
			t = temperature(k);


			double h;
			if (delta <= 0)
				h = 0.0;
			else
			{
				double f = ( 0.3 + double(rand()) / RAND_MAX * 0.4);
				h = exp(-f/t);
			}
			writer->write( h );
			//std::cout<<energy(S2) << std::endl;
		}
}

std::vector<int> VeryFastAnnealing::generateSolution(std::vector<int> prevSolution, double
temperature) {
		double prob = 0 + double(rand()) / RAND_MAX * 1;
		int pos;
		if (prob > 0.5)
			pos = 0 + rand() % 3;
		else
			pos = 3 + rand() % 3;
		int newComponent = 0;
		switch (prevSolution[pos]) {
			case 0:
				newComponent = 1 + rand() % 2;
```

```
                        break;
                case 2:
                        newComponent = 0 + rand() % 2;
                        break;
                case 1:
                        {
                                if ( ( 0 + double(rand()) / RAND_MAX * 1) > 0.5)
                                        newComponent = 2;
                                else
                                        newComponent = 0;
                                break;
                        }
        }

        prevSolution[pos] = newComponent;
        return prevSolution;
}
```

Приложение Д

```
class Selector {
protected:
        Chromosome firstParent;
        Chromosome secondParent;
        Population population;
public:
        Selector(const Population & pop) : population(pop) {}
        virtual std::pair<Chromosome, Chromosome> select() = 0;
};

class BestRandSelector : public Selector {


public:
        BestRandSelector(const Population & pop) : Selector(pop) {}
        std::pair<Chromosome, Chromosome> select() override;
};

class RandSelector : public Selector {


public:
        RandSelector(const Population & pop) : Selector(pop) {}
        std::pair<Chromosome, Chromosome> select() override;
};

class RouletteWheel : public Selector {


public:
        RouletteWheel(const Population & pop) : Selector(pop) {}
        std::pair<Chromosome, Chromosome> select() override;
};
```

```
template <class T>
class Genetic {
public:
	Genetic(const Population & pop) : population(pop) {}

	std::pair<Chromosome, Chromosome> select() {
		Selector * selector = new T(population);
		return selector->select();
	}

	Chromosome cross(std::pair<Chromosome, Chromosome> parents) {
		Point2Crossover * crossover = new Point2Crossover;
		return crossover->make(parents);
	}

	Chromosome mutation(Chromosome ch) {
		Mutator * mutator = new Mutator;
		return mutator->mutation(ch);
	}

	Chromosome invert(Chromosome ch) {
		Inverter * inverter = new Inverter;
		return inverter->invert(ch);
	}

	void setRecord() {
		population.setRecord();
	}

	void changePopulation(Chromosome ch) {
		population.changePopulation(ch);
	}

	double getRecord() {
		return population.getRecord();
	}

	void changeRecord(double newRecord) {
		population.changeRecord(newRecord);
	}

	Population population;
};

class Crossover {
public:
	Chromosome make(std::pair<Chromosome, Chromosome> parents);
};

class Point2Crossover {
public:
	Chromosome make(std::pair<Chromosome, Chromosome> parents);
};

class Mutator {
private:
	Impacts impacts;

public:
	Mutator() {
		impacts.add(new FirstImpact());
```

```
            impacts.add(new SecondImpact());
            impacts.add(new ThirdImpact());
        }

        Chromosome mutation(Chromosome ch);
};

class Inverter {
private:
        int cut1;
        int cut2;

public:
        Chromosome invert(Chromosome ch);
};




//////genetic.cpp

std::pair<Chromosome, Chromosome> BestRandSelector::select() {
        firstParent = population.getBestChromosome();
        do {
                secondParent = population.getRandomChromosome();
        } while(firstParent.fit == secondParent.fit);

        return std::make_pair(firstParent, secondParent);
}


std::pair<Chromosome, Chromosome> RandSelector::select() {
        firstParent = population.getRandomChromosome();
        do {
                secondParent = population.getRandomChromosome();
        } while(firstParent.fit == secondParent.fit);

        return std::make_pair(firstParent, secondParent);
}

std::pair<Chromosome, Chromosome> RouletteWheel::select() {
        std::vector<Chromosome> bufPopulation = population.getPopulation();
        std::vector<double> fitnesses;
        std::vector<double> wheel;
        double sum = 0.0;
        int pos1 = 0;
        int pos2 = 0;

        for (int i = 0; i < bufPopulation.size(); i++) {
                sum += bufPopulation[i].fit;
        }

        for (int i = 0; i < bufPopulation.size(); i++) {
                double val = ( 1 - (bufPopulation[i].fit / sum) ) / (bufPopulation.size() -
1);
                fitnesses.push_back(val);
        }

        wheel.push_back(fitnesses[0]);

        for (int i = 1; i < bufPopulation.size(); i++) {
```

```
			wheel.push_back(wheel[i-1] + fitnesses[i]);
		}

		while (pos1 == pos2) {
			double prob1 = 0 + double(rand()) / RAND_MAX * 1;
			double prob2 = 0 + double(rand()) / RAND_MAX * 1;

			if (prob1 < wheel[0]) pos1 = 0;
			if (prob2 < wheel[0]) pos2 = 0;

			for (int i = 1; i < bufPopulation.size(); i++) {
				if (prob1 > wheel[i-1] && prob1 <= wheel[i])
					pos1 = i;

				if (prob2 > wheel[i-1] && prob2 <= wheel[i])
					pos2 = i;
			}
		}


		firstParent = bufPopulation[pos1];
		secondParent = bufPopulation[pos2];

		return std::make_pair(firstParent, secondParent);
}



Chromosome Crossover::make(std::pair<Chromosome, Chromosome> parents) {
		Chromosome resault1;
		Chromosome resault2;
		int separator = 0 + rand() % (chromosome_lenth - 1);
		for (int i = 0; i <= separator; i++) {
			resault1.genes.push_back(parents.first.genes[i]);
			resault2.genes.push_back(parents.second.genes[i]);
		}

		for (int j = separator + 1; j <= chromosome_lenth - 1; j++) {
			resault1.genes.push_back(parents.second.genes[j]);
			resault2.genes.push_back(parents.first.genes[j]);
		}

		int child = 1 + rand() % 2;
		return (child % 2 == 0) ? resault2 : resault1;
}

Chromosome Point2Crossover::make(std::pair<Chromosome, Chromosome> parents) {
		int separator1 = 0 + rand() % (chromosome_lenth );
		int separator2 = 0 + rand() % (chromosome_lenth );
		while (separator1 == separator2) {
			separator2 = 0 + rand() % (chromosome_lenth - 1);
		}

		Chromosome resault1;
		Chromosome resault2;
		for (int i = 0; i <= separator1; i++) {
			resault1.genes.push_back(parents.first.genes[i]);
			resault2.genes.push_back(parents.second.genes[i]);
		}

		for (int j = separator1 + 1; j < separator2; j++) {
			resault1.genes.push_back(parents.second.genes[j]);
```

```
			resault2.genes.push_back(parents.first.genes[j]);
		}

		for (int k = separator2; k < chromosome_lenth; k++) {
			resault1.genes.push_back(parents.first.genes[k]);
			resault2.genes.push_back(parents.second.genes[k]);
		}

		double prob = 0 + double(rand()) / RAND_MAX * 1;

		return (prob > 0.5) ? resault2 : resault1;
}


Chromosome Mutator::mutation(Chromosome ch) {
		int pos = 0 + rand() % 6;
			int changePos = 0;

			if (*ch.genes[pos] == FirstImpact()) {
				changePos = 1 + rand() % 2;
			} else if(*ch.genes[pos] == ThirdImpact()) {
				changePos = 0 + rand() % 2;
			} else if(*ch.genes[pos] == SecondImpact()) {
				double prob = 0 + double(rand()) / RAND_MAX * 1;
				if (prob > 0.5)
					changePos = 2;
				else
					changePos = 0;
			}

			ch.genes[pos] = impacts[changePos];

			return ch;
}

Chromosome Inverter::invert(Chromosome ch) {
		cut1 = 0 + rand() % (ch.genes.size() - 1);

		do {
			cut2 = 0 + rand() % ch.genes.size();
		} while(cut2 <= cut1);

		std::reverse(ch.genes.begin() + cut1, ch.genes.end() - ch.genes.size() + cut2);

		return ch;
}


/////// Population.h



struct Impact {
public:
		bool operator==(const Impact & imp1) {
			return ((imp1.charset[1] == this->charset[1]) &&
				(imp1.charset[2] == this->charset[2]));
		}

		int charset[3];
};
```

```
struct FirstImpact : public Impact {
	FirstImpact() {
		charset[0] = 0;
		charset[1] = 0;
		charset[2] = 0;
	}
};

struct SecondImpact : public Impact {
	SecondImpact() {
		charset[0] = 0;
		charset[1] = 1;
		charset[2] = 0;
	}
};

struct ThirdImpact : public Impact {
	ThirdImpact() {
		charset[0] = 0;
		charset[1] = 0;
		charset[2] = 1;
	}
};

struct Chromosome {
	std::vector<Impact*> genes;
	double fit;
	Chromosome() : fit(0.0) {}

	void fitness( double (*f)(std::vector<int> solution) ) {

		std::vector<int> positions;
		for (int i = 0; i <= chromosome_lenth - 1; i++) {
			int imp = 0;
			if (*genes[i] == SecondImpact())
				imp = 1;
			else if (*genes[i] == ThirdImpact())
				imp = 2;
			positions.push_back(imp);
		}

		fit = f(positions);

	}
};

struct Impacts {
private:
	std::vector<Impact*> impacts;
public:
	Impact* operator[] (int t) {
		return impacts[t];
	}

	void add(Impact* imp) {
		impacts.push_back(imp);
	}
};



class Population {
```

```
private:
	double record;
	int populationSize;
	std::vector<Chromosome> population;
	Impacts impacts;

public:
	Population(int size, const Impacts & imp)
		: populationSize(size)
		, impacts(imp) {}

	void setPopulation( double (*f)(std::vector<int> solution) );
	void setRecord();
	double getRecord();
	void changeRecord(double newRecord);
	void changePopulation(Chromosome ch);
	Chromosome* createChromosome();
	Chromosome getBestChromosome();
	Chromosome getRandomChromosome();
	std::vector<Chromosome> getPopulation();
};



Chromosome* Population::createChromosome() {
	Chromosome * chromosome = new Chromosome;
	for (int j = 0; j <= chromosome_lenth - 1; j++) {
		int imp_count = 0 + rand()%3;
		chromosome->genes.push_back(impacts[imp_count]);
	}
	return chromosome;
}

void Population::setPopulation( double (*f)(std::vector<int> solution) ) {
	for (int i = 1; i <= populationSize; i++) {

		Chromosome ch = *createChromosome();
		ch.fitness(f);
		while (ch.fit< 0.55) {
			ch = *createChromosome();
			ch.fitness(f);
		}
		population.push_back(ch);

		//population.push_back(*createChromosome());
	}
}



Chromosome Population::getBestChromosome() {
	double minFit = population[0].fit;
	Chromosome bestChromosome = population[0];
	for(std::vector<Chromosome>::iterator it = population.begin(); it !=
population.end(); it++) {
		if (it->fit < minFit) {
			minFit = it->fit;
			bestChromosome = *it;
		}
	}
	return bestChromosome;
```

```
}

Chromosome Population::getRandomChromosome() {
	int pos = 0 + rand() % population.size();
	return population[pos];
}

std::vector<Chromosome> Population::getPopulation() {
	return population;
}

double Population::getRecord() {
	return this->record;
}

void Population::setRecord() {
	double min = population[0].fit;
	for each(auto it in population) {
		if (it.fit < min) min = it.fit;
	}

	this->record = min;
}

void Population::changeRecord(double newRecord) {
	if (newRecord < this->record)
		this->record = newRecord;
}

void Population::changePopulation(Chromosome ch) {
	int pos = 0;
	double max = population[0].fit;
	for (int i = 0; i <= population.size() - 1; i++) {
		if (population[i].fit > max) {
			max = population[i].fit;
			pos = i;
		}
	}
	population.erase( population.begin() + pos );
	population.push_back(ch);
```

Приложение Е

```
struct Node {
	bool used;
	double dist;
	Node * parent;
	std::vector<Node*> childs;
	std::pair<int, int> position;

	Node(Node * _parent) {
		parent = _parent;
		used = false;
		for (int i = 0; i < childsAmount; i++) {
			childs.push_back( NULL );
		}
	}

	void setPosition(std::pair<int, int> _position) {
```

```
		this->position = _position;
	}

	std::pair<int, int> getPosition() {
		return this->position;
	}

	void setChilds(int time) {
		for (int i = 0; i < childsAmount; i++) {
			//childs.push_back( new Node(this) );
			childs[i] = new Node(this);
			childs[i]->setPosition(std::pair<int, int>(time + 1, i));
		}
	}

	Node * getChild(int pos) {
		return childs[pos];
	}

	Node * getParent() {
		return this->parent;
	}

	void setDistance(const double & _dist) {
		if (parent != NULL)
			this->dist = _dist + parent->getDistance();
		else
			this->dist = _dist;
	}

	double getDistance() {
		return this->dist;
	}

	void setUsed() {
		this->used = true;
	}

	bool isUsed() {
		return this->used;
	}
};



class Tree {
private:
	std::stack<int> solution;
	double min;
	Node * minElem;
	double (*foo)(int, int);
	Node * root;
	int timesAmount;
	std::vector< std::vector<Node*> > matrix;
public:
	Tree(int _timesAmount, double (*_foo)(int, int)) : timesAmount(_timesAmount),
foo(_foo), min(10000.0) {}
	void setMatrix();
	void setStructure();
	void breadthFirstSearch();
	void reverseMotion();
	void showSolution();
```

```
};



void Tree::setMatrix() {
	int lvlSize = childsAmount;
	for (int i = 1; i <= timesAmount; i++) {
		std::vector<Node*> childs;
		for (int j = 0; j < lvlSize; j++) {
			//childs.push_back(new Node( std::pair<int, int>(i, j) ));
		}
		matrix.push_back(childs);
		lvlSize = lvlSize * childsAmount;
	}
}

void Tree::setStructure() {
	std::queue<Node*> q;
	std::queue<Node*> p;
	root = new Node(NULL);
	root->setDistance(0.0);
	root->setPosition(std::pair<int, int>(0, 0));
	q.push(root);
	for (int i = 0; i <= timesAmount - 1; i++) {
		while( !q.empty() ) {
			Node * elem = q.front();
			elem->setChilds(i);
			q.pop();
			for(int i = 0; i < childsAmount; i++) {
				p.push(elem->getChild(i));
			}
		}
		while(!p.empty()) {
			q.push(p.front());
			p.pop();
		}
	}
}

void Tree::breadthFirstSearch() {
	std::queue<Node*> que;
	que.push(root);
	root->setUsed();
	while(!que.empty()) {
		Node * elem = que.front();
		que.pop();
		if (elem->getChild(0) != NULL) {
			for(int i = 0; i < childsAmount; i++) {
				Node * child = elem->getChild(i);
				if(!child->isUsed()) {
					child->setUsed();
					que.push(child);

					child->setDistance( foo(child->getPosition().second,
child->getPosition().first-1) );
				}
			}
		} else if (elem->getDistance() < min) {
			min = elem->getDistance();
			minElem = elem;
		}
```

```
	}
}

void Tree::reverseMotion() {
	while (minElem->getParent() != NULL) {
		solution.push(minElem->getPosition().second);
		minElem = minElem->getParent();
	}
}

void Tree::showSolution() {
	while(!solution.empty()) {
		std::cout << solution.top() << "  ";
		solution.pop();
	}
}
```