Министерство образования Республики Беларусь

Учреждение образования

БелорусскиЙ государственный университет

информатики и радиоэлектроники

Факультет информационных технологий и управления

Кафедра информационных технологий автоматизированных систем

|  |
| --- |
| *К защите допустить*: |
| Заведующий кафедрой ИТАС |
| \_\_\_\_\_\_\_\_\_\_\_А. А. Навроцкий |

Пояснительная записка

к дипломному проекту

на тему

**Программный комплекс поддержки принятия решений на основе нечетких множеств**

БГУИР ДП 1-53 01 02 06 006 ПЗ

|  |  |  |
| --- | --- | --- |
| Студент |  | Д. В. Дишук |
| Руководитель |  | С. А. Валяк |
| Консультанты: |  |  |
| *от кафедры ИТАС* |  | А. Л. Гончаревич |
| *по экономической части* |  | Т. Л. Слюсарь |
| Нормоконтролер |  | Е. В. Протченко |
| Рецензент |  |  |

Минск 2020

Министерство образования Республики Беларусь

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Учреждение образования БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | |
| Факультет | ИТиУ | | | | Кафедра | | | | ИТАС | | | | | | | | | | | | |
| Специальность | 1-53 01 02 | | | | Специализация | | | | | | | 06 | | | | | | | | | |
| УТВЕРЖДАЮ | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | |  | | | | | Зав. кафедрой | | | | | |
| « | | | | | | | | | | | 2 | | » | | апреля | | | 2020 | | г. |
| **ЗАДАНИЕ** | | | | | | | | | | | | | | | | | | | | | |
| **по дипломному проекту (работе) студента** | | | | | | | | | | | | | | | | | | | | | |
| Дишук Дарьи Валерьевны | | | | | | | | | | | | | | | | | | | | | |
| (фамилия, имя, отчество) | | | | | | | | | | | | | | | | | | | | | |
| 1 Тема проекта (работы): | | Программный комплекс поддержки принятия решений на | | | | | | | | | | | | | | | | | | | |
| основе нечетких множеств | | | | | | | | | | | | | | | | | | | | | |
| утверждена приказом по университету от | | | | | « | 01 | | » | | апреля | | | | 2020 г. | | | № | | 872-с | | |
| 2 Срок сдачи студентом законченной работы | | | | | | | 1 июня 2020 г. | | | | | | | | | | | | | | |
| 3 Исходные данные к проекту: | | | *Microsoft Windows* не ниже 7 версии, установленный | | | | | | | | | | | | | | | | | | |
| браузер | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | |
| Назначение разработки: | | | | | | | | | | | | | | | | | | | | | |
| разработать программный комплекс поддержки принятия решений на основе нечетких | | | | | | | | | | | | | | | | | | | | | |
| множеств | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | |
| 4 Содержание пояснительной записки (перечень подлежащих разработке вопросов): | | | | | | | | | | | | | | | | | | | | | |
| Введение | | | | | | | | | | | | | | | | | | | | | |
| 1 Анализ предметной области | | | | | | | | | | | | | | | | | | | | | |
| 2 Проектирование структуры программного комплекса | | | | | | | | | | | | | | | | | | | | | |
| 3 Программная реализация программного комплекса | | | | | | | | | | | | | | | | | | | | | |
| 4 Технико-экономическое обоснование | | | | | | | | | | | | | | | | | | | | | |
| Заключение | | | | | | | | | | | | | | | | | | | | | |
| Список использованных источников | | | | | | | | | | | | | | | | | | | | | |
| Приложение А (обязательное) Программный код обработки пользовательских данных | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | |
| 5 Перечень графического материала (с точным указанием обязательных чертежей): | | | | | | | | | | | | | | | | | | | | | |
| Структура программного комплекса (ПД) – формат А1, лист 1. | | | | | | | | | | | | | | | | | | | | | |
| Схема алгоритма обработки пользовательских данных (ПД) – формат А1, лист 1. | | | | | | | | | | | | | | | | | | | | | |
| Диаграмма вариантов использования системы (ПЛ) – формат А1, лист 1. | | | | | | | | | | | | | | | | | | | | | |
| Диаграмма деятельности (ПЛ) – формат А1, лист 1. | | | | | | | | | | | | | | | | | | | | | |
| Диаграмма состояний (ПЛ) – формат А1, лист 1. | | | | | | | | | | | | | | | | | | | | | |
| Интерфейс программы (Ввод данных) (ПЛ) – формат А1, лист 1. | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | |
| 6 Содержание задания по технико-экономическому обоснованию: | | | | | | | | | | | | | | | | | | | | | |
| Технико-экономическое обоснование разработки и использования программного | | | | | | | | | | | | | | | | | | | | | |
| комплекса поддержки принятия решений на основе нечетких множеств | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | |
| Задание выдал |  | | | Т. Л. Слюсарь | | | | | | | | | | | | | | | | | |

КАЛЕНДАРНЫЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапов дипломного проекта (работы) | Объём этапа, % | Срок выполнения этапа | Примечание |
| Сбор и изучение материалов по теме дипломного проектирования. Написание раздела 1 пояснительной записки, расчёт технико-экономического обоснования | 40 | 20.04.2020 |  |
| Проектирование программного модуля. Написание раздела 2 пояснительной записки и соответствующего графического материала | 60 | 04.05.2020 |  |
| Реализация программного модуля. Написание раздела 3 пояснительной записки и соответствующего графического материала | 80 | 18.05.2020 |  |
| Оформление пояснительной записки и подготовка презентации | 100 | 01.06.2020 |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Дата выдачи задания | 02.04.2020 | | Руководитель | |  | С.А. Валяк |
| Задание принял к исполнению | |  | | Д.В. Дишук | |

**РЕФЕРАТ**

ПРОГРАММНЫЙ КОМПЛЕКС ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ НА ОСНОВЕ НЕЧЕТКИХ МНОЖЕСТВ: дипломный проект / Д. В. Дишук. – Минск : БГУИР, 2020, – п.з. –66 с., чертежей (плакатов) – 6 л. формата А1.

Дипломный проект на тему «Программный комплекс поддержки принятия решений на основе нечетких множеств» разработан с целью облегчения принятия решений в условиях нечетких данных, которые заданы не в числовом виде.

Дипломный проект представлен в виде графического материала на 6 листах формата А1 и пояснительной записки на 66 страницах, состоящей из 4 разделов.

В первом разделе выполняется анализ предметной области и ставится задача на дипломное проектирование.

Во втором разделе описан процесс проектирования системы.

В третьем разделе описаны используемые средства реализации и непосредственно программная часть проекта, а также приводятся примеры работы.

В четвертом разделе осуществляется технико-экономическое обоснование проекта, а именно расчет затрат и отпускной цены, а также прибыль от реализации программного продукта.

**СОДЕРЖАНИЕ**

[Введение 6](#_Toc42144317)

[1 Анализ предметной области 7](#_Toc42144318)

[1.1 Многокритериальный анализ 7](#_Toc42144319)

[1.2 Нечеткие множества 11](#_Toc42144320)

[1.3 Применение нечетких множеств при принятии решений 15](#_Toc42144321)

[1.4 Системы поддержки принятия решений 16](#_Toc42144322)

[1.5 Существующие решений программные средства для решения задач на основе нечетких множеств 18](#_Toc42144323)

[2 Проектирование структуры и ее основных компонентов 20](#_Toc42144324)

[2.1 Общая характеристика организации сущности задачи 20](#_Toc42144325)

[2.2 Структура системы 20](#_Toc42144326)

[2.3 Математическое/алгоритмическое обеспечение 21](#_Toc42144327)

[2.4 Входные и выходные данные 26](#_Toc42144328)

[2.5 Техническое системное программное обеспечение 28](#_Toc42144329)

[2.6 Эргономическое обеспечение 28](#_Toc42144330)

[3 Программная реализация программного комплекса 30](#_Toc42144331)

[3.1 Выбор программных средств реализации 30](#_Toc42144332)

[3.2 Структура программного обеспечения 31](#_Toc42144333)

[3.3 Разработка программного кода 33](#_Toc42144334)

[3.4 Руководств пользователя 36](#_Toc42144335)

[3.5 Контрольные примеры 45](#_Toc42144336)

[4 Технико-экономическое обоснование 54](#_Toc42144337)

[4.1 Характеристика программного средства 54](#_Toc42144338)

[4.2 Расчет инвестиций в разработку программного комплекса 54](#_Toc42144339)

[4.3 Расчет экономического эффекта от использования программного средства 57](#_Toc42144340)

[4.4 Расчет показателей экономической эффективности разработки и использования программного средства 58](#_Toc42144341)

[4.5 Вывод по результатам расчета 59](#_Toc42144342)

[Заключение 60](#_Toc42144343)

[Список использованных источников 61](#_Toc42144344)

[Приложение А (обязательное) Код обработки пользовательских данных (к подразделу 3.3) 62](#_Toc42144345)

[Ведомость документов 66](#_Toc42144346)

# ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ

апммм

# ВВЕДЕНИЕ

Задачи принятия решений представляют обширный класс задач, относящихся к исследованию операций, в том числе задачи многокритериального альтернативного выбора.

Существует большой класс задач, когда оценки критериального соответствия могут быть заданы только приблизительно, либо имеются обоснованные сомнения в точности числовых значений, либо оценки вообще могут быть заданы только в виде лингвистических утверждений типа «большое соответствие», «незначительное соответствие» и т. п. В этих условиях для решения задач многокритериального альтернативного выбора вполне обосновано использование аппарата нечетких множеств.

Постановка всякой задачи многокритериального выбора включает три объекта − множество возможных решений, векторный критерий и отношение предпочтения Р. Решить эту задачу – означает на основе векторного критерия и имеющихся сведений об отношении предпочтения ЛПР найти множество выбираемых решений.

При решении прикладных многокритериальных задач основным типом дополнительной информации, с которым чаще всего приходится иметь дело является информация об относительной важности критериев. Поэтому многие из существующих подходов к решению многокритериальных задач используют именно эту информацию, чаще всего в виде так называемых коэффициентов относительной важности критериев. Обычно считается, что эти коэффициенты должны назначаться экспертами

Целью дипломного проекта является разработка системы поддержки принятия решений на основе нечетких множеств.

# АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

## Многокритериальный анализ

**1.1.1** Методы математического программирования, интенсивно развиваемые в исследовании операций, изначально ориентировались на решение однокритериальных задач. Однако со временем росла необходимость учитывать существование более одного показателя эффективности, оптимальные решения по которым не совпадают. С этого периода началось бурное развитие многокритериальных методов принятия решений и, в частности, методов многокритериального математического программирования.

Многокритериальность может быть обусловлена одной из трех причин:

* цель не может быть адекватно представлена (покрыта) одним критерием;
* принимающий решения ставит более одной цели, которые связаны общими активными средствами;
* решения принимаются группой лиц с несовпадающими интересами [1].

**1.1.2** Постановка всякой задачи многокритериального выбора включает три объекта − множество возможных решений, векторный критерий и отношение предпочтения ЛПР. Решить эту задачу – означает на основе векторного критерия и имеющихся сведений об отношении предпочтения ЛПР найти множество выбираемых решений.

Важнейшим инструментом решения многокритериальных задач является принцип Эджворта-Парето (принцип Парето), который стали успешно применять еще в XIX веке. Однако принцип Эджворта-Парето имеет вполне определенные границы применимости и его использование при решении некоторых задач рискованно или же вообще не допустимо. Для того чтобы сформулировать принцип Эджворта-Парето, постановку обычной многокритериальной задачи, включающей множество возможных решений и набор критериев (векторный критерий) необходимо дополнить бинарным отношением предпочтения лица, принимающего решение (ЛПР) [2, 32]. Расширенная подобным образом многокритериальная задача названа задачей многокритериального выбора. Ее решение заключается в отыскании так называемого множества выбираемых решений, которое может состоять из одного элемента, но в общем случае, оно является подмножеством множества возможных решений.

Применение принципа Эджворта-Парето позволяет из множества всех возможных исключить заведомо неприемлемые решения, т.е. те, которые никогда не могут оказаться выбранными, если выбор осуществляется достаточно «разумно». После такого исключения остается множество, которое называют множеством Парето или областью компромиссов.

Оно, как правило, является достаточно широким и в процессе принятия решений неизбежно встает вопрос о том, какое именно возможное решение выбрать среди парето-оптимальных. Выражаясь иначе, какие из парето-оптимальных решений следует удалить для того, чтобы произвести дальнейшее сужение области компромиссов и, тем самым, получить более точное представление об искомом множестве выбираемых решений? Этот вопрос при решении практических многокритериальных задач является наиболее трудным и наименее проработанным к настоящему времени.

В общем случае, располагая лишь множеством возможных решений и набором критериев (т.е. оставаясь в рамках модели многокритериальной задачи), обоснованного ответа на поставленный вопрос не сможет дать ни один специалист по принятию решений, поскольку осуществление компромисса (выбора того или иного парето-оптимального решения) возможно лишь при расширении модели выбора за счет привлечения дополнительной информации об отношении предпочтения ЛПР. В зависимости от типа, характера и объема имеющейся в наличии дополнительной информации используют тот или иной метод принятия решений (или же их комбинацию).

**1.1.3** Основной тип дополнительной информации, с которым чаще всего приходится иметь дело при решении прикладных многокритериальных задач – это информация об относительной важности критериев. Поэтому многие из существующих подходов к решению многокритериальных задач используют именно эту информацию, чаще всего в виде так называемых коэффициентов относительной важности критериев. Обычно считается, что эти коэффициенты должны назначаться экспертами.

В формальном представлении критерии (целевые функции), по которым оценивается решение Х, будет записываться в виде . Критерий называют также частными. Для удобства рассуждений примем, что для всех *i* чем больше значение критерия, тем лучше. Тогда задача многокритериального математического программирования запишется в виде:

где *D* – множество допустимых решений.

Иначе говоря, задача состоит в максимизации вектора критериев по .

Существенное отличие этой задачи от традиционной однокритериальной состоит в понятии оптимальности. В однокритериальной задаче под оптимальным понимается решение, обеспечивающее максимальное значение критерия. При многих критериях увеличение одних критериев приводит к уменьшению других (редкие исключения не представляют практического интереса) и поэтому понятие оптимальности требует принципиальных уточнений. Очевидно, что без дополнительной информации о предпочтениях ЛПР бессмысленно говорить об оптимальном решении и тем более формализовано искать его.

**1.1.4** Успешное решение многокритериальных задач невозможно без использования различного рода сведений о предпочтениях лица, принимающего решение. При этом одним из самых главных источников таких сведений является информация об относительной важности критериев.

Без установления принципа оптимальности, отражающего предпочтения ЛПР, невозможно формально распознать оптимальное решение. Однако учитывая стремление ЛПР к увеличению значений всех частных критериев, можно формальными методами исключить не перспективные точки и тем самым облегчить решение задачи [3, 7].

В специальной литературе предложены различные способы вовлечения ЛПР в процесс принятия решений. В зависимости от того, на какой стадии процесса выявляются и используются предпочтения ЛПР, можно выделить три группы многокритериальных методов принятия решений:

* основаны на том, что ЛПР может выразить свои предпочтения до начала процесса многокритериальной оптимизации;
* интерактивные (диалоговые) методы;
* методы построения множества эффективных решений с последующим представлением его ЛПР.

В методах первой группы используются различные способы свёртки критериев, лексикографическое упорядочение критериев, установление желаемых уровней критериев и др. Вторая группа методов основана на непосредственном участии ЛПР в процессе оптимизации, когда на каждой итерации компьютер предлагает решения, а ЛПР их оценивает, и с учётом этих оценок компьютер ищет новые решения. Методы третьей группы отличаются друг от друга различными способами построения и представления множества эффективных решений.

**1.1.5** ЛПР может выразить свои предпочтения в различной форме. Это зависит от особенностей самого ЛПР, новизны задачи, типа и числа критериев и других факторов. Поэтому методы данной группы отличаются тем, что используют разные представления предпочтений и способы их формализации. Однако все они в конечном итоге сводят многокритериальную задачу к одной или ряду задач с одним (иногда обобщенным) критерием. К методам первой группы относятся:

* функция полезности;
* решение на основе лексикографического упорядочения критериев;
* метод главного критерия;
* линейная свертка;
* максиминная свертка;
* метод идеальной точки;
* целевое программирование (ЦП).

**1.1.6** Интерактивный процесс решения многокритериальной задачи реализуется путем диалога ЛПР с компьютером. При этом происходит чередование этапов вычислений, выполняемых компьютером, и корректировки и принятия решений ЛПР. Такая процедура позволяет ЛПР более полно и глубоко оценить взаимосвязи критериев и возможности оптимизируемой системы. Более того, в интерактивном процессе может развиваться формирование предпочтений, компромиссов и даже системы ценностей. Все это облегчает ЛПР нахождение решения, наилучшего с его точки зрения, и повышает уверенность в правильности выбора.

Существуют следующие интерактивные методы принятия решений:

* метод уступок;
* интерактивное компромиссное программирование;
* метод STEM;
* метод взвешенных метрик Чебышева;
* прогрессивный алгоритм принятия многокритериальных решений.

**1.1.7** Также существуют методы построения эффективного множества. Очевидно, что если ЛПР показать эффективное множество в целом, то он сам сможет выбрать тот эффективный вектор, который предпочитает всем остальным. В таком случае для решения задачи не потребуется извлекать знания о предпочтениях ЛПР. Однако при числе критериев свыше двух построение эффективного множества оказывается непростой задачей [2, 78].

Эти методы особенно эффективны в том случае, когда ЛПР — не один человек, а группа лиц, из которых "каждый имеет свои собственные цели. Кроме того, в этом подходе открывается возможность для публичного обсуждения достоинств п недостатков принятого решения

## Нечеткие множества

**1.2.1** Множеством называется совокупность определенных вполне различаемых объектов, рассматриваемых как единое целое.

Л. Заде расширил двузначную оценку 0 или 1 до неограниченной многозначной оценки выше 0 и ниже 1 на [0, 1] и впервые ввел понятие нечеткого множества, заменив характеристическую функцию на функцию принадлежности, которая может принимать любые значения в интервале [0, 1] для . В соответствии с этим элемент множества *U* может не принадлежать , может быть элементом А в небольшой степени ( близко к нулю), может более или менее принадлежать А ( не слишком близко к 0, не слишком близко к единице), может быть в значительной степени элементом А ( близко к единице) или, наконец, может быть элементом . С точки зрения характеристической функции нечеткие множества есть естественное обобщение обычных множеств, когда мы отказываемся от бинарного характера этой функции и предполагаем, что она может принимать любые значения на отрезке [0, 1]. Множество значений х, на котором определена функция принадлежности, получило название нечеткого множества [4,].

Чаще всего определение нечеткого множества объясняют следующим образом: величина обозначает субъективную оценку степени принадлежности *х* множеству A, например, означает, что *х* на 80 % принадлежит множеству A.

**1.2.2** Понятие «функция принадлежности» – одно из базовых в теории нечетких множеств, поэтому вопрос о возможных методах ее построения имеет важное значение. Здесь необходимо отметить два обстоятельства. Функция принадлежности нечеткого множества определяется вне самой теории нечетких множеств, т. е. проверить корректность построения функции принадлежности невозможно методами самой теории. Насколько удачно выбран тип функции принадлежности, можно видеть в конце решения задачи. С одной стороны, указанное обстоятельство критиками теории нечетких множеств рассматривается как существенный недостаток, с другой – функция принадлежности – это способ формализации нечетких субъективных утверждений или оценок, и доказать справедливость или несправедливость этих оценок можно лишь в процессе или по окончании решения задачи. Второе обстоятельство состоит в том, что введенное в начале рассмотрения определение нечеткого множества не накладывает ограничений на выбор вида функции принадлежности. Вместе с тем удобно использовать такие варианты функций принадлежности, которые допускают аналитическое представление, что позволяет более эффективно расходовать вычислительные ресурсы. Кроме того, определенные типы функций принадлежности являются стандартными в ряде соответствующих программных средств.

Функция принадлежности – это некоторая невероятностная субъективная мера нечеткости, определяемая в результате опроса экспертов о степени соответствия элемента x понятию, формализуемому нечетким множеством A. В отличие от вероятностной меры, которая считается оценкой стохастической неопределенности, имеющей дело с неоднозначностью наступления некоторого события в различные моменты времени, нечеткая мера выступает численной оценкой лингвистической неопределенности, связанной с неоднозначностью и расплывчатостью категорий человеческого мышления. При построении функции принадлежности с каждым нечетким множеством A ассоциируется некоторое свойство, признак или атрибут R, который характеризует некоторую совокупность объектов X. Чем в большей степени конкретный объект *x* ∈ X обладает этим свойством R, тем более близко к единице соответствующее значение . Если элемент *x* ∈ X определенно обладает этим свойством R, то , если же *x* ∈ X определенно не обладает этим свойством R, то

**1.2.3** Несмотря на то что функции принадлежности строятся на основе экспертных заключений, для исключения произвола были сформулированы некоторые правила, которых необходимо придерживаться. Прежде чем переходить непосредственно к построению функций принадлежности, следует задать универсальное множество, к которому принадлежит соответствующее нечеткое множество. Выбор универсального множества определяется характером решаемой задачи и типом нечеткой переменной. В некоторых случаях бывает удобно использовать в качестве универсального множества отрезки [0, 1] или [–1, 1]. В большинстве задач универсальное множество определяется на некоторой числовой оси. При расположении нечетких множеств на универсальном придерживаются принципа естественной упорядоченности, когда нечеткие множества, соответствующие лингвистическим значениям «малое», «низкое», «плохое» и т. п., располагаются в левой части области определения универсального множества, соответствующие значениям «большое», «высокое», «хорошее» и т. п. – в правой части.

**1.2.4** Типизация функций принадлежности в контексте решаемой задачи существенно упрощает соответствующие аналитические и численные расчеты при применении методов теории нечетких множеств. Выделяют кусочно-линейные, типовые функции принадлежности. к которым относятся треугольные и трапецеидальные функции принадлежности, использующиеся для задания неопределенностей типа «приблизительно равно», «среднее значение», «расположен в интервале», «подобен объекту», «похож на предмет» и т. п.

Достоинства кусочно-линейных функций принадлежности:

* для их задания требуется малый объем данных;
* простота модификации параметров (модальных значений) функции принадлежности на основе измеряемых значений входных и выходных величин системы;
* возможность получения в рамках модели отображения «вход—выход» в виде гиперповерхности, состоящей из линейных участков;
* легко обеспечивается выполнение условия пересечения функций принадлежности, соответствующих соседним лингвистическим значениям в точке 0,5.

К недостаткам кусочно-линейных функций принадлежности можно отнести то, что такие функции принадлежности не являются непрерывно дифференцируемыми. Соответственно модель системы, содержащая подобные функции, также не является непрерывно дифференцируемой.

**1.2.5** Методы построения функций принадлежности можно разделить на две группы: прямые и косвенные (изображение 1.1).

Прямые методы характеризуются тем, что эксперт непосредственно задает правила определения значений функции принадлежности , характеризующей элемент .

Как правило, прямые методы задания функции принадлежности используются для измеримых понятий или, когда выделяются полярные значения. Разновидностями прямых методов можно назвать прямые групповые методы, когда, например, группе экспертов предъявляют конкретный объект, и каждый должен дать один из двух ответов: принадлежит или нет этот объект к заданному множеству. Тогда число утвердительных ответов, деленное на общее число экспертов, дает значение функции принадлежности объекта к данному нечеткому множеству. Прямыми методами являются также непосредственное задание функции принадлежности таблицей, графиком или формулой.

Из анализа результатов исследований и решения практических задач, связанных с необходимостью обрабатывать информацию, известно, что прямые методы в основном используются в качестве вспомогательных, так как характеризуются большой долей субъективизма.

  
  
Изображение 1.1 — Методы построения функций принадлежности

Косвенные методы построения значений функции принадлежности используются в случаях, когда нет элементарных измеримых свойств, через которые определяются нечеткие множества. В косвенных методах значения функции принадлежности выбираются таким образом, чтобы удовлетворить ранее сформулированные условия. Экспертная информация служит только исходной для дальнейшей обработки. Дополнительные условия могут налагаться как на вид получаемой информации, так и на процедуру ее обработки. К таким методам относятся статистический, метод парных сравнений, экспертных оценок и ряд других.

**1.2.6** Выбор функции принадлежности значительно определяется объемом имеющейся информации о моделируемой системе, а также качеством имеющихся в распоряжении исследователя методов настройки модели.

При малом объеме имеющейся информации о системе следует использовать простейшие кусочно-линейные функции принадлежности, состоящие из прямолинейных участков, для нахождения параметров, которых требуется значительно меньшее по сравнению с остальными функциями принадлежности количество информации.

Наличие большого объема информации о системе в форме измеренных входных и выходных данных дает возможность идентификации большего числа параметров нечеткой модели, что позволяет использовать более сложные функции принадлежности, такие как П-образные, и тем самым приводит к моделям наиболее точным, чем в случае простых функций, состоящих из прямолинейных участков. Вместе с тем для идентификации большого числа параметров нечеткой модели требуются высокоэффективные методы ее адаптации (настройки), которые не всегда имеются в распоряжении исследователя. Опыт ряда исследователей позволяет говорить о преимуществе в данной ситуации более простых, состоящих из прямолинейных участков функций принадлежности, упрощающих процесс настройки (обучения) нечеткой модели, обеспечивая при этом ее высокую точность.

**1.2.7** Для нечетких множеств определено достаточно много различных операций, часть из которых справедлива и для классических множеств. Следует отметить, что при выполнении различных операций над нечеткими множествами необходимые преобразования выполняются над соответствующими функциями принадлежности.

Сравнение нечетких множеств и выполнение над ними различных операций будет возможно, когда соответствующие нечеткие множества определены на одном и том же универсуме.

## Применение нечетких множеств при принятии решений

Задачи принятия решений представляют обширный класс задач, относящихся к исследованию операций. В зависимости от исходной формулировки они могут решаться методами теории игр и статистических решений. Отдельный класс составляют задачи многокритериального альтернативного выбора.

Существует большой класс задач, когда оценки критериального соответствия могут быть заданы только приблизительно, либо имеются обоснованные сомнения в точности числовых значений, либо оценки вообще могут быть заданы только в виде лингвистических утверждений типа «большое соответствие», «незначительное соответствие» и т. п. В этих условиях для решения задач многокритериального альтернативного выбора вполне обосновано использование аппарата нечетких множеств.

Описание в форме нечетких множеств гораздо менее требовательно к квалификации экспертов и зачастую точнее отражает суть дела и имеющуюся у лица, принимающего решения, информацию. Предлагаемые теорией решения, основанные на нечеткой информации, и сами несут на себе печать нечеткости. Они могут рассматриваться лишь как рекомендации для лица, принимающего решения, требуя от него выбора одного из предлагаемых вариантов. Тем не менее, даже этот факт можно рассматривать как достоинство теории: он показывает, как увеличение информированности ЛПР сказывается на достоверности и правильности принимаемых решений.

В настоящее время все методы применения нечетких множеств для решения указанных задач можно разделить на методы на основе отношений нечеткого предпочтения, критериальной свертки, правил нечеткого вывода.

## Системы поддержки принятия решений

**1.4.1** Строгого общепринятого определения СППР нет. В общем случае СППР определяется как диалоговые автоматизированные системы, которые позволяют лицам, принимающим решения (ЛПР), использовать данные, знания, объективные и субъективные модели, правила принятия решений для идентификации, анализа и решения слабоструктурированных и неструктурированных ЗПР. Целью СППР является помощь людям, принимающим решение в сложных условиях, при осуществлении полного и объективного анализа предметной деятельности.

СППР представляет собой скоординированный набор данных, систем, инструментов и технологий, программного и аппаратного обеспечения, с помощью которого предприятие собирает и обрабатывает информацию о возникшей проблемной ситуации с целью повышения обоснованности принимаемых управленческих действий.

**1.4.2** Основу СППР составляет комплекс взаимосвязанных моделей с соответствующей информационной поддержкой исследования, экспертные БД, включающие опыт решения задач управления и обеспечивающие участие коллектива экспертов в процессе выработки рациональных решений.

Построение моделей автоматизации процесса принятия решений для задач управления на основе экспертной информации, имеющих нечеткое описание, базируется на понятиях «нечеткое множество» и «лингвистическая переменная».

**1.4.3** СППР обладает следующими основными характеристиками:

* СППР использует и данные, и модели;
* СППР предназначены для помощи ЛПР в принятии решений для слабоструктурированных и неструктурированных задач;
* они поддерживают, а не заменяют выработку решений ЛПР;
* цель СППР повысить обоснованность и эффективность принимаемых решений.

Основными функциями СППР являются:

* оказание помощи ЛПР при анализе исходной информации (оценка сложившейся обстановки и ограничений, накладываемых средой);
* выявления и ранжирование приоритетов, учет неопределенности в оценках ЛПР и формирование его предпочтений;
* генерация возможных решений (формирование списка альтернатив;
* оценка возможных альтернатив, исходя из предпочтений ЛПР и ограничений среды;
* анализ возможных последствий принимаемых решений;
* рекомендация по выбору лучшего с точки зрения ЛПР решения;
* в условиях коллективного принятия решений обеспечение постоянного обмена информацией о принимаемых решениях и помощь в согласовании групповых решений;
* сбор данных о результатах реализации принятых решений и оценка результатов.

Обычно в СППР выделяют три основные части:

* Система данных для сбора и хранения информации, получаемой из внутренних и внешних источников. Обычно это Хранилище данных.
* Система диалога, позволяющая пользователю задавать, какие данные следует выбирать и как их обрабатывать.
* Система моделей и методов - идеи, алгоритмы и процедуры, которые позволяют обрабатывать данные и проводить их анализ.

**1.4.4** В настоящее время большинство СППР еще носят всего лишь информационно-справочный характер, однако все шире для анализа и выработки предложений в СППР используются разные методы. Это могут быть: информационный поиск, интеллектуальный анализ данных, поиск знаний в базах данных, рассуждение на основе прецедентов, имитационное моделирование, эволюционные вычисления и генетические алгоритмы, нейронные сети, ситуационный анализ, когнитивное моделирование и др. Некоторые из этих методов были разработаны в рамках теории искусственного интеллекта. Если в основе работы СППР лежат методы искусственного интеллекта, то говорят об интеллектуальной СППР, или ИСППР.

Главное отличие интеллектуальных систем поддержки принятия решений от информационно- справочных систем состоит в том, что обязательным элементом функционирования является формирование рекомендаций, или проектов решений.

Перспективно применение в СППР комбинированных методов принятия решений в сочетании с методами искусственного интеллекта и компьютерным моделированием, различные имитационно-оптимизационные процедуры, принятие решений в сочетании с экспертными процедурами.

## Существующие решений программные средства для решения задач на основе нечетких множеств

В настоящее время известно достаточно большое число программных систем для решения различных задач с применением нечетких множеств. Подробное описание этих систем, а также методы их использования для решения конкретных задач изложены в различных источниках.

**1.5.1** Пакет MATLAB (MATrix LABoratory) компании MathWoks (США), представляющий собой интегрированную среду для выполнения численных расчетов, компьютерного моделирования и вычислительных экспериментов, содержит широкий набор готовых функций, используемых при анализе и синтезе систем автоматического управления.

Для решения задач методами теории нечетких множеств в пакете MATLAB предусмотрен пакет нечеткой логики Fuzzy Logic Toolbox.

Основные возможности пакета:

* построение систем нечеткого вывода (экспертных систем, регуляторов, аппроксиматоров зависимостей);
* построение адаптивных нечетких систем (гибридных нейронных сетей);
* интерактивное динамическое моделирование в среде Simulink.

Нечеткое управление моделируется с помощью системы нечеткого вывода FIS (Fuzzy Inference System).

FIS-редактор обеспечивает высокий уровень общения с системой, не имеет ограничений на число входных и выходных переменных, которое ограничивается лишь доступным объемом памяти ЭВМ.

**1.5.2** Пакет нечеткого моделирования fuzzyTECH – разработанная и постоянно обновляемая программа компании INFORM GmbH (Inform Software Corporation, Германия), предназначенная для решения различных задач нечеткого моделирования. В отличие от MATLAB программа fuzzyTECH служит специализированным средством, которое позволяет разрабатывать и исследовать разнообразные нечеткие модели в графическом режиме, а также преобразовывать их в программный код на одном из языков программирования с возможностью последующей реализации в программируемых микроконтроллерах.

Программу fuzzyTECH можно использовать в качестве сервера или клиента при нечетком управлении удаленными объектами. Важной особенностью программы fuzzyTECH считается возможность автоматической генерации документации для разработанных нечетких моделей в виде текста с иллюстрациями в формате RTF.

**1.5.3** Программная система Fexcel, предназначенная для решения аналитических задач в условиях неопределенности, является программной надстройкой для широко распространенной электронной таблицы Microsoft Excel. Это четвертая версия программного обеспечения, разработанная консалтинговой группой INEX-FT (Украина). Fexcel основан на использовании авторских алгоритмов нечеткой технологии для решения аналитических задач в условиях неопределенности. С помощью Fexcel пользователь может решать три типа аналитических задач:

* любые расчеты с использованием нечетких чисел;
* задачи оценки и классификации альтернатив (задачи соотнесения альтернатив к описанным классам);
* задачи прогнозирования.

Fexcel создан как обычная надстройка программного обеспечения Microsoft Excel и не требует специальных знаний нечеткой математики. Работа Fexcel скрыта от пользователя, который, однако, должен понимать природу неопределенности в аналитической задаче и иметь некоторые навыки интерпретации нечетких чисел.

## Постановка задачи

Рассмотренные программные средства обладают огромным набором реализованных возможностей, однако для рядового пользователя это является скорее недостатком т.к. часто слишком сложны и могут приводить к ошибкам, вызванным с человеческим фактором, и связанным с непониманием устройства системы и принципа ее работы.

Объем руководства пользователя для каждого из приведенных инструментов составляет не менее 60 страниц. Это связано с тем, что ПС предназначены для узкого круга технических специалистов и инженеров, имеющих соответствующие знания для работы с такими инструментами. Таким образом для использования решения задач необходимо нанимать сторонних специалистов либо обучать сотрудников, это влечет за собой трудо- и ресурсозатраты.

Кроме того программные средства, рассмотренные выше требуют являются коммерческими. Например пакет MATLAB является платным и

На основании проведенного анализа были определены следующие требования к проектируемому программному комплексу принятия решений на основе нечетких множеств:

* создание пользовательских критериев;
* сохранение шаблонов для использования;
* использование существующих шаблонов;
* использование существующих примеров для обучения;
* задание альтернатив, из которых необходимо выбрать одну наилучшую;
* ранжирование альтернатив;
* динамическое изменение весов критериев и параметров.

# ПРОЕКТИРОВАНИЕ СТРУКТУРЫ И ЕЕ ОСНОВНЫХ КОМПОНЕНТОВ

## Общая характеристика организации сущности задачи

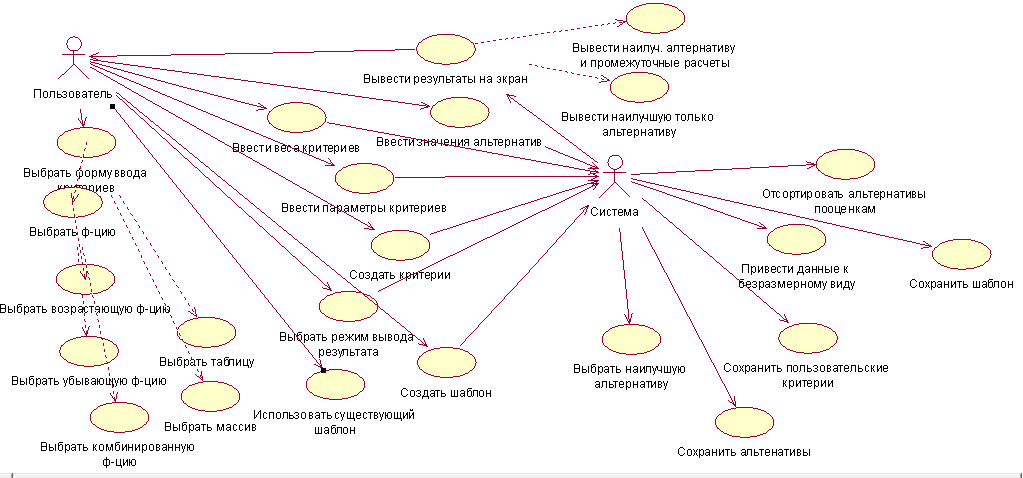
Существует класс задач, когда оценки критериального соответствия могут быть заданы только приблизительно, либо имеются обоснованные сомнения в точности числовых значений, либо оценки вообще могут быть заданы только в виде утверждений типа «хорошо», «плохо» и т. п. В этих условиях для решения задач многокритериального альтернативного выбора вполне обосновано использование аппарата нечетких множеств.

Разрабатываемая система предназначена для поддержки принятия решений на основе нечетких множеств.

В системе должны быть реализованы следующие функции:

* возможность выбор количества значимых критериев;
* создание пользовательских критериев оценки альтернативы разными способами;
* возможность задания значимости (весов) критериев;
* ввод пользовательских оценок для альтернатив;
* выбор наилучшей альтернативы;
* вывод результата на экран;
* вывод промежуточных расчетов
* создание пользовательских критериев;
* сохранение шаблонов для использования;
* использование существующих шаблонов;
* использование существующих примеров для обучения;
* задание альтернатив, из которых необходимо выбрать одну наилучшую;
* ранжирование альтернатив;
* динамическое изменение весов критериев и параметров.

Для отображения возможных вариантов разработана диаграмма прецендентов представленная на рисунке 2.1.

  
  
Рисунок 2.1 — Диаграмма прецедентов

Программный комплекс предоставляет возможность принятия решения о выборе наилучшей альтернативы по критериям, заданным пользователем и может работать в двух режимах:

* принятие решения с выводом на экран только результата (режим предназначен только для принятия решения);
* принятие решения с выводом на экран результата и результата промежуточных расчетов, совершаемых в процессе принятия решения системы (режим может использоваться для корректировки весов или при обучении студентов/школьников/сотрудников).

## Структура системы

Система разделена на следующие модули:

* «Выявление системы предпочтения пользователя»;
* «Определение важности критериев»;
* «Комплексная оценка альтернатив».

Предпочтения пользователя могут выражаться в численном и качественном виде. Для этого предназначен модуль задания критериев и параметров этих критериев в разных формах.

Для комплексной оценки критериев разработан блок ввода альтернатив, их обработки и вывода на экран в одном из режимов: вывод только наилучшей альтернативы без промежуточных данных и вывод наилучшей альтернативы вместе со значениями, полученными в процессе обработки входных данных.

## Математическое/алгоритмическое обеспечение

Одним из способов задания пользовательских критериев является задание функции принадлежности. Функция принадлежности нечёткого множества — обобщение индикаторной (или характеристической) функции классического множества. В нечёткой логике она представляет степень принадлежности каждого члена пространства рассуждения к данному нечёткому множеству.

Функция принадлежности элемента к множеству *А* ставит в соответствие каждому элементу этого множества число из интервала [0,1], характеризующее степень принадлежности данного элемента заданному множеству. Т.е. это некоторая не вероятностная субъективная мера нечеткости, определяемая в результате опроса экспертов о степени соответствия элемента x понятию, формализуемому нечетким множеством A. В отличие от вероятностной меры, которая является оценкой стохастической неопределенности, имеющей дело с неоднозначностью наступления некоторого события в различные моменты времени, нечеткая мера является численной оценкой лингвистической неопределенности, связанной с неоднозначностью и расплывчатостью категорий человеческого мышления.

При построении функции принадлежности с каждым нечетким множеством *A* ассоциируется некоторое свойство, признак или атрибут, который характеризует некоторую совокупность объектов X. Чем в большей степени конкретный объект обладает этим свойством, тем более близко к 1 соответствующее значение . Если элемент определенно обладает этим свойством, то , если же x ∈ X определенно не обладает этим свойством, то .

На практике удобно использовать те функции принадлежности, которые допускают аналитическое представление в виде некоторой простой математической функции.

Кусочно-линейные функции принадлежности используются для задания неопределенностей типа: «приблизительно равно», «среднее значение», «расположен в интервале», «подобен объекту», «похож на предмет» и т.п.

Методы построения функций принадлежности:

* прямые;
* косвенные.

В зависимости от числа привлеченных к опросу экспертов как прямые, так и косвенные методы делятся на одиночные и групповые.

В прямых методах эксперт либо группа экспертов просто задают для каждого значение функции принадлежности .

Как правило, прямые методы построения функций принадлежности используются для таких свойств, которые могут быть измерены в некоторой количественной шкале. Например, такие физические величины, как скорость, время, расстояние, давление, температура и другие имеют соответствующие единицы и эталоны для своего измерения.

При прямом построении функций принадлежности следует учитывать, что теория нечетких множеств не требует абсолютно точного задания функций принадлежности. Зачастую бывает достаточно зафиксировать лишь наиболее характерные значения и вид функции принадлежности.

Процесс построения или задания нечеткого множества на основе некоторого известного заранее количественного значения измеримого признака получил даже специальное название — фаззификация или приведение к нечеткости. Речь идет о том, что хотя иногда нам бывает известно некоторое значение измеримой величины, мы признаем тот факт, что это значение известно неточно, возможно с погрешностью или случайной ошибкой. При этом, чем меньше мы уверены в точности измерения признака, тем большим будет интервал носителя соответствующего нечеткого множества. Следует помнить, что в большинстве практических случаев абсолютная точность измерения является лишь удобной абстракцией для построения математических моделей. Именно по этой причине фаззификация позволяет более адекватно представить объективно присутствующую неточность результатов физических измерений.

В контексте нечеткой логики под фаззификацией понимается не только отдельный этап нечеткого вывода, но и собственно процесс или процедура нахождения значений функции принадлежности нечетких множеств (термов) на основе обычных (не нечетких) исходных данных. Фаззификацию еще называют введением нечеткости.

Целью этапа фаззификации является установление соответствия между конкретным, обычно численным, значением отдельной входной переменной системы нечеткого вывода и значением функции принадлежности соответствующего ей терма входной лингвистической переменной. После завершения этого этапа для всех входных переменных должны быть определены конкретные значения функций принадлежности по каждому из лингвистических термов, которые используются в подусловиях базы правил системы нечеткого вывода.

Дефаззификация в системах нечеткого вывода представляет собой процедуру или процесс нахождения обычного (не нечеткого) значения для каждой из выходных лингвистических переменных множества. Цель дефаззификации заключается в том, чтобы, используя результаты аккумуляции всех выходных лингвистических переменных, получить обычное (не нечеткое) значение каждой из выходных переменных. Дефаззификацию также называют приведением к четкости.

Косвенные методы построения функций принадлежности используются при решении задач, для которых свойства физических величин не могут быть измерены.

Наибольшее распространение среди косвенных методов получил метод парных сравнений.

Метод анализа иерархий (МАИ), предложенный Т.Л. Саати, основан на парных сравнениях альтернативных вариантов по различным критериям с использованием девятибалльной шкалы и последующим ранжированием набора альтернатив по всем критериям и целям. Этот метод относится к классу критериальных и занимает особое место, благодаря тому, что он получил исключительно широкое распространение и активно применяется по сей день. Взаимоотношения между критериями учитываются путем построения иерархии критериев и применением парных сравнений для выявления важности критериев и подкритериев.

МАИ не предписывает лицу, принимающему решение (ЛПР), какого-либо «правильного» решения, а позволяет ему в интерактивном режиме найти такой вариант (альтернативу), который наилучшим образом согласуется с его пониманием сути проблемы и требованиями к ее решению.

К основным процедурам метода анализа иерархий относятся следующие:

* генерация множества альтернативных вариантов;
* формирование множества критериев для оценки альтернативных вариантов и представление его в виде иерархии;
* выявление предпочтений экспертов на множестве альтернатив по различным критериям;
* установление относительной важности влияния критериев на общую цель и другие критерии;
* получение ранжированных наборов альтернатив по всем критериям и целям.

Все оценки определяются экспертами. Сначала эксперты генерируется множество допустимых альтернатив, среди которых необходимо провести выбор лучшей альтернативы или упорядочивание всех элементов.

Вершиной иерархий обычно является глобальная цель, на следующих уровнях присутствуют критерии и на самом нижнем уровне - альтернативы.

Иерархическая структура критериев и целей является моделью знаний конкретной предметной области, которая изменяется и уточняется с течением времени.

Элементы одного уровня иерархии попарно сравниваются по силе их влияния на элементы более высокого уровня. Результаты заносятся в матрицу попарных сравнений. При сравнении элемента с самим собой имеем равную значительность «1», т.е. главная диагональ матрицы состоит из единиц.

В МАИ используется 9-балльная шкала. Возможные значения и их объяснения отображены в таблице 2.1.

Таблица 2.1 — Значения по 9-ти бальной шкале

|  |  |  |
| --- | --- | --- |
| Интенсивности относительной важности | Определение | Объяснение |
| 1 | Равная важность | Равный вклад двух объектов в достижении цели |
| 3 | Умеренное превосходство одного над другим | Опыт и суждения дают легкое превосходство одному объекту над другим |
| 5 | Существенное или сильное превосходство | Опыт и суждения дают сильное превосходство одному объекту над другим |
| 7 | Значительное превосходство | Одному объекту дается настолько сильное превосходство над другим, что оно становится значимым |
| 9 | Очень сильное превосходство | Очевидность превосходства одного объекта над другим подтверждается наиболее сильно |
| 2,4,6,8 | Промежуточные значения между двумя соседними суждениями | Принимаются в компромиссных случаях |
| Обратные величины приведенных чисел | Если при сравнении одного объекта с другим получено одно из вышеуказанных чисел (например, 3), то при сравнении второго объекта с первым получим обратную величину (т.е.1/3) |  |

Следующий шаг состоит в вычислении вектора приоритетов по данной матрице. Существует несколько методов оценки этого вектора. Например, перемножить элементы каждой строки матрицы и нормировать извлечением корня степени равной количеству элементов в строе матрицы.

В рамках МАИ нет общих правил для формирования структуры модели принятия решения. Это является отражением реальной ситуации, поскольку всегда для одной и той же проблемы имеется целый спектр мнений. Метод позволяет учесть это обстоятельство с помощью построения дополнительной модели для согласования различных мнений, посредством определения их приоритетов. Таким образом, метод позволяет учитывать «человеческий фактор» при подготовке принятия решения. Это одно из важных достоинств данного метода перед другими методами принятия решений.

В рамках МАИ нет средств для проверки достоверности данных. Этот недостаток ограничивает отчасти возможности применения метода. Однако метод применяется главным образом в тех случаях, когда в принципе не может быть объективных данных, а ведущими мотивами для принятия решения являются предпочтения людей.

Работа по подготовке принятия решений часто является слишком трудоемкой для одного человека. Однако применение метода позволяет разбить большую задачу, на ряд малых самостоятельных задач. Благодаря этому для подготовки принятия решения можно привлечь экспертов, работающих независимо друг от друга над локальными задачами.

Метод дает только способ ранжирования альтернатив, но не имеет внутренних средств для интерпретации рейтингов, т.е. считается, что ЛПР, зная рейтинг возможных решений, должен в зависимости от ситуации сам сделать вывод. Это следует признать недостатком метода.

Метод дает удобные средства учета экспертной информации для решения различных задач. Он отражает естественный ход человеческого мышления и дает не только способ выявления наиболее предпочтительного решения, но и позволяет количественно выразить степень предпочтительности по средством ранжирования. Это способствует полному и адекватному выявлению предпочтений ЛПР. Кроме того, оценка меры противоречивости использованных данных позволяет установить степень доверия к полученному результату.

## Входные и выходные данные

**2.4.1** Источником данных является пользователь, который принимает решение о выборе наилучшей альтернативы. В таблице 2.2 приведено описание структур, разработанных для хранения вводимых пользователем данных.

Таблица 2.2 — Описание структур для хранения пользовательских данных

|  |  |
| --- | --- |
| Структура | Описание |
| criterions | Предназначена для хранения данных о критериях, созданных пользователем |
| alternatives | Предназначена для хранения данных об альтернативах, из которых будет выбираться наилучшая |
| criterionWeights | Предназначена для хранения весов критериев, заданных пользователем для оценивания альтернатив |

Каждая структура имеет свои поля, которых необходимы для удобного хранения вводимых данных.

**2.4.2** Структура *criterions* представляет собой массив коллекций ключ/значение, который разработан для сохранения информации о пользовательских критериях. Критерии могут быть введены пользователем в виде функции, таблицы и массива.

Информация о назначении полей данной структуры представлена в таблице 2.3.

Таблица 2.3 — Поля структуры *criterions*

|  |  |
| --- | --- |
| Поле | Описание |
| name | Предназначено для хранения названия критерия. Также используется для извлечения и добавления данных в структуру |
| criterionType | Предназначено для хранения информации о типе задания критерия. |
| paramValues | Предназначено для хранения вводимых пользователем значений критериев, которые используются для оценки альтернатив |
| paramnames | Предназначено для хранения вводимых пользователем названий параметров критериев, которые используются для оценки альтернатив |

**2.4.3** Структура *alternatives* представляет собой массив коллекций ключ/значение, который разработан для сохранения информации об альтернативах, из которых в результате работы системы должна быть выделена наилучшая.

Информация о назначении полей данной структуры представлена в таблице 2.4

Таблица 2.4 — Поля структуры *alternatives*

|  |  |
| --- | --- |
| Поле | Описание |
| %Название критерия% | Предназначено для хранения названия критерия. Также используется для извлечения и добавления данных в структуру. Для удобства название поля формируется по названию критерия |
| %Значение критерия для конкретной альтернативы% | Предназначено для хранения информации о значении критерия. |

**2.4.4** Структура *criterionWeights* представляет собой простой массив, предназначенный для хранения вводимых пользователем значений важности критериев. Важность критериев задается в числовом виде для облегчения восприятия.

**2.4.5** В процессе работы программы для обработки данных об альтернативах создаетсядвумерныймассив *alternativeCompare*. Он необходим для обработки структуры *alternatives*:

* нормировки значений критериев;
* подсчета итоговых значений;
* выделения наилучшей альтернативы.

Также массив используется для вывода данных в режиме работы с выводом промежуточных данных.

**2.4.6** Для вывода результата обработки данных на экран (наилучшей альтернативы) разработан массив *resultAlter.* Массив содержит данных в итоговом виде и используется для их вывода.

Т.к. имеется два режима работы системы, отличающиеся количеством выводимой на экран информации, для вывода во втором (расширенном) режиме с демонстрацией промежуточных расчетов используются дополнительные структуры данных.

## Эргономическое обеспечение

Для взаимодействия пользователя с разрабатываемой системой будет использоваться набор форм, содержащих поля и кнопки для ввода данных, а также таблицы для вывода результата работы системы.

Элементы появляются на экране пользователя по мере заполнения полей и нажатия кнопок. Это обеспечивает простоту использования системы и снижает риск возникновения ошибок ввода пользовательских данных, связанных с большим объемом отвлекающей информации на странице.

Простая монотонная цветовая палитра состоит из нескольких оттенков зеленого цвета, так как большое количество цветов может повлиять на концентрацию внимания пользователя. Так же были использованы шрифты одного стиля, но разных размеров для единства восприятия форм, которые постепенно выводятся на экран.

Целью такого упрощения интерфейса в том, является упрощение взаимодействия пользователя с системой и повышение тем самым ее полезности.

# Программная реализация программного комплекса

## Выбор программных средств реализации

JavaScript (часто сокращают до JS) — это лёгкий, интерпретируемый, объектно-ориентированный язык с функциями первого класса, самый известный скриптовый язык для веб-страниц, но также используется во многих не браузерных окружениях. Он является прототипно-ориентированным, мультипарадигменным языком сценариев, который поддерживает динамический, объектно-ориентированный, императивный и функциональный стили программирования.

JavaScript запускается на стороне клиента Интернета, который может использоваться для создания/программирования того, как веб-страницы будут вести себя при наступлении каких-либо событий.

JavaScript может функционировать и как процедурный, и как объектно-ориентированный язык. Объекты можно создавать программно во время выполнения, путем присоединения методов и свойств или пустых объектов во время выполнения, в отличие от синтаксических определений классов в компилируемых языках, таких как С++ или Java. После того, как объект был создан, он может быть использован в качестве плана (или прототипа) для создания похожих объектов.

Динамические возможности JavaScript включают: создание объектов во время выполнения, переменное число параметров, динамическое создание скриптов (с помощью eval), перебор объектов (с помощью for ... in), восстановление исходного кода (программы на JavaScript могут декомпилировать тела функций обратно в исходный код).

Интерфейс прикладного программирования (Application Programming Interfaces, APIs) - это готовые конструкции языка программирования, позволяющие разработчику строить сложный функционал с меньшими усилиями. Они "скрывают" более сложный код от программиста, обеспечивая простоту использования.

Для JavaScript на стороне клиента, в частности, существует множество API. Они не являются частью языка, а построены с помощью встроенных функций JavaScript для того, чтобы увеличить возможности при написании кода. Их можно разделить на две категории: API браузера и сторонние API.

API браузера — конструкции, встроенные в браузер, построенные на основе языка JavaScript, предназначенные для облегчения разработки функционала. Они способны использовать данные браузера и компьютерной среды для осуществления более сложных действий с этими данными. К примеру, API Геолокации (Geolocation API) предоставляет простые в использовании конструкции JavaScript для работы с данными местоположения. На самом деле, в браузере выполняется сложный низкоуровневый код (например, на C++) для подключения к устройству GPS (или любому другому устройству геолокации), получения данных и передачи их браузеру для обработки программой, но, эти детали скрыты благодаря API.

Сторонние API — конструкции, встроенные в сторонние платформы (такие как Twitter, Facebook) позволяющие использовать часть функционала этих платформ в своих собственных веб-страницах/приложениях (например, показывать последние Твиты на странице). Сторонние API не встроены в браузер по умолчанию. Такие API и информацию о них обычно необходимо искать в интернете. Например, Twitter API позволяет размещать последние твиты (tweets) на веб-сайте. В данном API определён набор конструкций, осуществляющих запросы к сервисам Twitter и возвращающих определённые данные.

Основой HTML-документа являются теги. В соответствии с объектной моделью документа («*Document Object Model*», коротко DOM), каждый *HTML*-тег является объектом. Вложенные теги являются «детьми» родительского элемента. Текст, который находится внутри тега, также является объектом.

Все эти объекты доступны при помощи *JavaScript*, можно использовать их для изменения страницы.

Объектная модель документа (DOM) - это интерфейс программирования для документов HTML и XML. Он представляет страницу, так что программы могут изменять структуру документа, стиль и содержание. DOM представляет документ в виде узлов и объектов. Таким образом, языки программирования могут подключаться к странице.

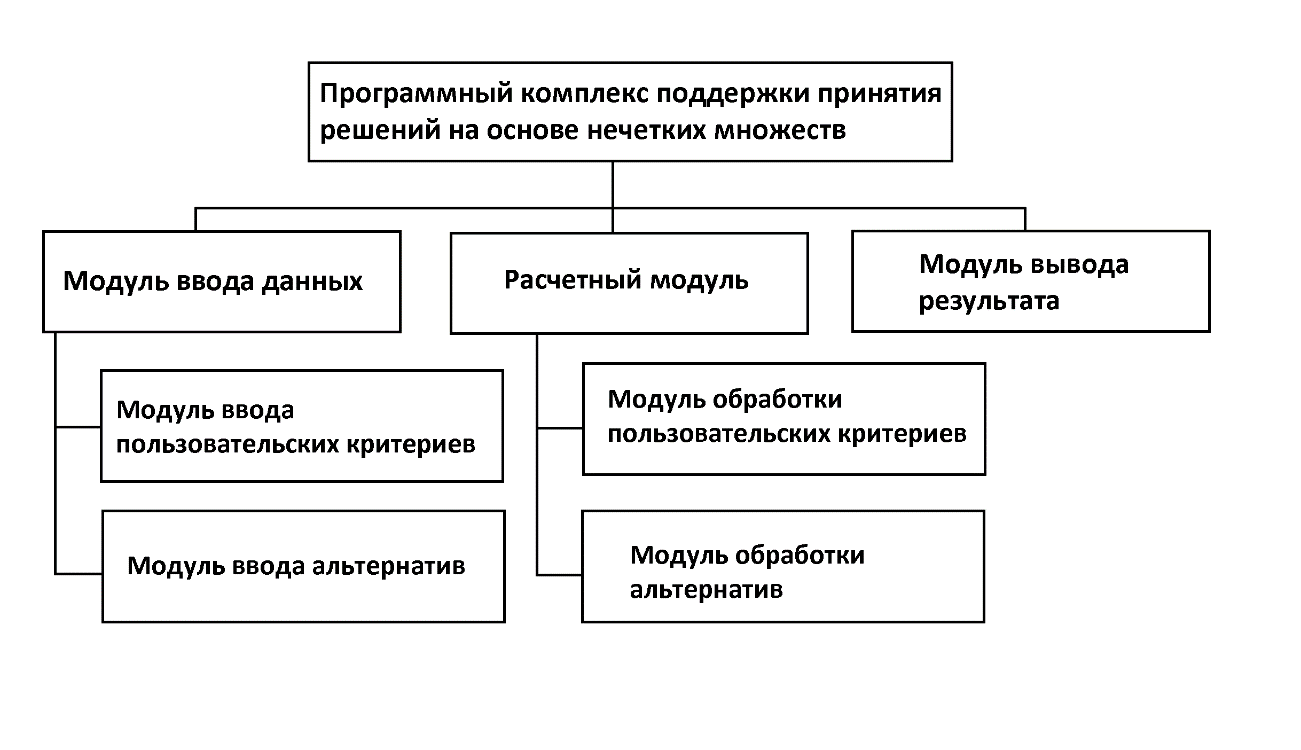
Веб-страница - это документ. Этот документ может отображаться либо в окне браузера, либо как источник HTML. Объектная модель документа (DOM) представляет тот же документ, поэтому им можно манипулировать. DOM — это объектно-ориентированное представление веб-страницы, которое можно изменить с помощью языка сценариев, такого как JavaScript.

DOM был разработан, чтобы быть независимым от какого-либо конкретного языка программирования, делая структурное представление документа доступным из единого, согласованного API. Реализации DOM могут быть построены для любого языка.

## Структура программного обеспечения

Программная реализация программного комплекса поддержки принятия решений представлена на изображении 3.1 и состоит из следующих модулей:

* «Модуль ввода данных»;
* «Расчетный модуль»;
* «Модуль вывода данных».

  
  
Изображение 3.1 — Структурная схема программного комплекса

Для поддержки принятия решения в системе используются следующие разработанные функции.

Функция *criterionsEnter()* используется для создания полей для ввода критериев и для выбора формы ввода пользовательских критериев.

*funcСriterionsChoosing()* — входными параметрами являются название критерия и его порядковый номер. Используется для выбора вида функции для ввода пользовательских критериев.

*funcСriterionsCreationO()* — входными параметрами являются название критерия и его порядковый номер. Используется для заполнения данных о критериях, вводимых в форме кусочно-линейных функций.

Функция *funcСriterionsSaveO()* — входными параметрами являются название критерия и его порядковый номер. Используется для сохранения данных о критериях, вводимых в форме кусочно-линейных функций.

Функция *altern()* — входным параметром является количество критериев. Используется для задания количества альтернатив и заполнения полей со значениями параметров критериев этих альтернатив.

*saveAlters()* — входными параметрами являются количество критериев и количествоальтернатив. Используется для сохранения значений полей параметров критериев этих альтернатив.

*tableNorm()* — Используется для приведения критериев к коллекции вида «параметр—степень принадлежности»

*altNorm()* — Используется для приведения альтернатив к безразмерному виду для дальнейшего использования в функции *range().*

*range()* — Используется для присвоения альтернативам оценок с учетом весов критериев и выбора наилучшей альтернативы.

*showTables()* — Используется для вывода промежуточных расчетов на экран в режиме с расширенным выводом.

## Разработка программного кода

Создание полей для ввода данных о пользовательских критериях реализовано с помощью *DOM API*:

*var newDiv = document.createElement("div");*

*var str=criterions[i].get("name");*

*newDiv.innerHTML ='<h1>'+str+'</h1>';*

*newDiv.setAttribute('id','div'+str);*

*document.body.appendChild(newDiv);*

*var newButtonL = document.createElement("input");*

*newButtonL.setAttribute('type','button');*

*newButtonL.setAttribute('id','funcChoosing'+str);*

*newButtonL.setAttribute('value','Функция');*

*newButtonL.setAttribute('onClick','funcСriterionsChoosing("'+str+'",'+i+')');*

*var newButtonR = document.createElement("input");*

*newButtonR.setAttribute('type','button');*

*newButtonR.setAttribute('id','tablChoosing'+str);*

*newButtonR.setAttribute('value','Таблица');*

*newButtonR.setAttribute('onClick','tablСriterionsChoosing("'+str+'",'+i+')');*

*var newForm=document.createElement("form");*

*newForm.setAttribute('id','form'+str);*

*newForm.appendChild(newButtonL);*

*newForm.appendChild(newButtonR);*

*newForm.appendChild(newButtonC);*

*document.body.appendChild(newForm);*

Сохранение введенных в поля данных переход к другим полям для критериев, которые вводятся через значения функций принадлежности реализованы в следующем виде:

*let x=new Array(3);*

*for (k=0;k<x.length-1;k++){*

*x[k]=parseFloat(document.getElementById(str+"x"+k).value);*

*}*

*x[2]=0;*

*criterions[i].set("paramValues",x);*

*document.getElementById("div"+str).remove();*

*document.getElementById("form"+str).remove();*

Для критериев, которые вводятся через таблицы попарного сравнения:

*let temp=new Array(j);*

*let names=new Array(j);*

*for (k=0;k<j;k++){*

*temp[k]=[];*

*names[k]=document.getElementById(str+0+k).value;*

*for (q=1;q<j+1;q++){*

*temp[k][q-1]=parseInt(document.getElementById(str+q+k).value, 10);*

*}}*

*criterions[i].set("paramValues",temp);*

*criterions[i].set("paramnames",names);*

*document.getElementById("div"+str).remove();*

*document.getElementById("form"+str).remove();*

*document.getElementById('tablSaving'+str).remove();*

Важным этапом принятия решения о выборе наилучшей альтернативы является создание удобной формы для ввода значений:

*for(j=0;j<altAmount;j++){*

*var newDiv=document.createElement("div");*

*newDiv.setAttribute("id","alternativeDiv"+j);*

*newDiv.innerHTML="<h1>Альтернатива "+j+"</h1>";*

*var newForm=document.createElement("form");*

*newForm.setAttribute("id","alternativeForm"+j);*

*document.body.appendChild(newDiv);*

*document.getElementById("alternativeDiv"+j).appendChild(newForm);*

*for(k=0;k<i;k++){*

*var newP=document.createElement("br");*

*var newField=document.createElement("input");*

*newField.setAttribute('type','text');*

*newField.setAttribute('id','alterCrit'+j+k);*

*newField.setAttribute('value',criterions[k].get("name")+" значение");*

*document.getElementById("alternativeForm"+j).appendChild(newField);*

*document.getElementById("alternativeForm"+j).appendChild(newP);*

*}}*

*document.getElementById("mainDiv").remove();*

*document.getElementById("altsDiv").remove();*

*var newDiv = document.createElement("div");*

*newDiv.setAttribute('id','altsSavingDiv');*

*var newButton = document.createElement("input");*

*newButton.setAttribute('type','button');*

*newButton.setAttribute('id','altsaving');*

*newButton.setAttribute('value','Сохранить');*

*newButton.setAttribute('onClick','saveAlters("'+altAmount+'",'+i+')');*

*document.body.appendChild(newDiv);*

*document.getElementById("altsSavingDiv").appendChild(newButton);*

Для сохранения введенных данных реальзована функция заполнения набора данных вида «название—значение»:

*for (k=0;k<altAmount;k++){*

*alternatives[k]=new Map();*

*for(j=0;j<i;j++){*

*alternatives[k].set(criterions[j].get("name"),*

*document.getElementById('alterCrit'+k+j).value);*

*}*

*document.getElementById("alternativeDiv"+k).remove();}*

В режиме расширенного вывода результатов на экран выводятся промежуточные таблицы. Например, вывод таблицы с данными приведенными к безразмерным значениям реализован с помощью следующих строк кода:

*innerHtmlText+='<center><table><tbody><tr>*

*<td><span><b>Критерий</h3></td>';*

*innerHtmlText+='<td><span><b>Вес</b></span></td>';*

*innerHtmlText+='<h1>Промежуточные данные</h1>';*

*innerHtmlText+='<h3>(после приведения к безразмерному виду)</h3>';*

*innerHtmlText+='<center><table><tbody><tr><td><span><b>Критерий</h3></td>';*

*innerHtmlText+='<td><span><b>Вес</b></span></td>';*

*for (j=0;j<alternatives.length;j++){*

*innerHtmlText+='<td><span><b>Альтернатива '+j+'</b></span></td>';}*

*innerHtmlText+='</tr>';*

*for(i=0;i<criterions.length;i++){*

*var critName=criterions[i].get("name");*

*innerHtmlText+='<tr><td><span><b>'+critName+'</b></span></td>';*

*innerHtmlText+='<td><span><b>'+criterionWeights[i]+'</b></span></td>';*

*for (j=0;j<alternatives.length;j++){*

*var critValue=alternativeCompare[j][i];*

*innerHtmlText+='<td><span>'+critValue+'</span></td>';}*

*innerHtmlText+='</tr>';}*

*innerHtmlText+='</tbody></table></center>';*

*innerHtmlText+='</tr>';*

*innerHtmlText+='</tbody></table></center>';*

*newField.innerHTML=innerHtmlText;*

*document.body.appendChild(newField);*

# Руководство пользователя

## Техническое системное программное обеспечение

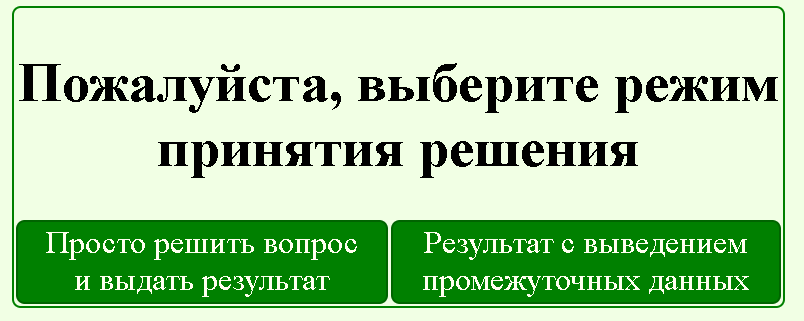
Компоненты системы должны функционировать на следующей минимальной конфигурации технических средств:

* компьютер класса Pentium III тактовая частота – 800 МГц;
* объем ОЗУ – 512 Мбайт для Windows Vista;
* видеоадаптер (допускается встроенный), позволяющий разрешающую способность – 1024х824;
* монитор размером 17'', обеспечивающий указанное выше разрешение (рекомендуется 19'');
* сетевой адаптер;
* клавиатура, манипулятор типа «мышь».

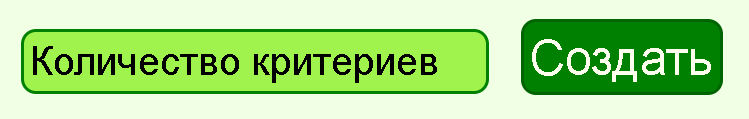
## Руководство пользования системой

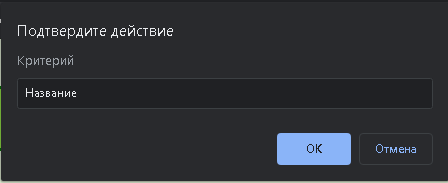
Реализуем режим работы, котором вместе с результатами выводятся промежуточные данный, полученные во время расчетов в процессе принятия решения.

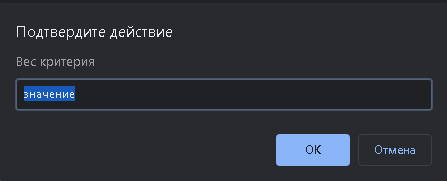
Необходимо нажать на соответствующую кнопку на главной странице. На изображении 3.2 представлена форма выбора режима на главной (разводящей) странице.

  
  
Изображение 3.2 — Форма выбора режима

**3.4.2** Далее необходимо заполнить поле для задания количества критериев, по которым будут оцениваться альтернативы, подтвердить создание введенного количества критериев (изображение 3.3) и ввести все необходимые данные о названиях критериев (изображение 3.4) и их весах (изображение 3.5).

  
  
Изображение 3.3 — Форма ввода количества критериев

  
  
Изображение 3.4 — Окно ввода названия критерия

  
  
Изображение 3.5 — Окно ввода веса критерия

**3.4.3** Следующим этапом задания пользовательских критериев является выбор формы ввода критерия. Возможность такого выбора реализована в виде формы с кнопками с соответствующими подписями, представленной на изображении 3.6.

  
  
Изображение 3.6 — Форма с кнопками выбора вида критерия

Критерии могут быть заданы в виде:

1. кусочно-линейных функций;
2. таблиц попарного сравнения параметров;
3. массивов типа степень принадлежности/параметр.

3.4.3.1 Кусочно-линейные функции принадлежности используются для задания неопределенностей типа: «приблизительно равно», «среднее значение», «расположен в интервале», «подобен объекту», «похож на предмет» и т.п.

В системе реализована возможность задания функций принадлежности в следующем виде: с возрастанием (изображение 3.7(а)), с убыванием (изображение 3.7(б)) и функция с участков (участком с постепенным возрастанием, фиксированным значением и участком постепенного убывания) (изображение 3.7(в)).

|  |  |  |
| --- | --- | --- |
|  |  |  |
| а) | б) | в) |

Изображение 3.7 — Виды функций, которыми можно задать критерий:  
а) — с возрастанием; б) — с убыванием; в) — с чередованием

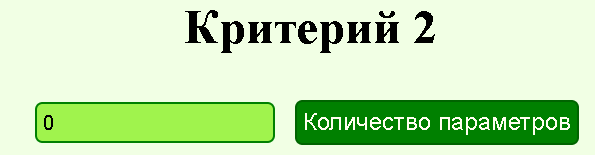
Для каждого из вида функций выводится форма для заполнений и схематичное отображение точек, данные о которых необходимо ввести. Формы для заполнения представлены на изображении 3.8.

|  |  |
| --- | --- |
|  |  |
| а) | б) |
|  |  |
|  | |
| в) | |

Изображение 3.8 — Задание точек для функций критериев разного вида:  
а) — с возрастанием; б) — с убыванием; в) — с чередованием

3.4.3.2 Парное сравнение представляет собой процедуру установления предпочтения объектов при сравнении всех возможных пар. При сравнении пары объектов возможно либо отношение строгого порядка, либо отношение эквивалентности. Подобное представление применяют для отражения критериев, заданных качественными значениями.

В форме, представленной на изображении 3.9 необходимо ввести в появившееся поле количество возможных параметров критерия.

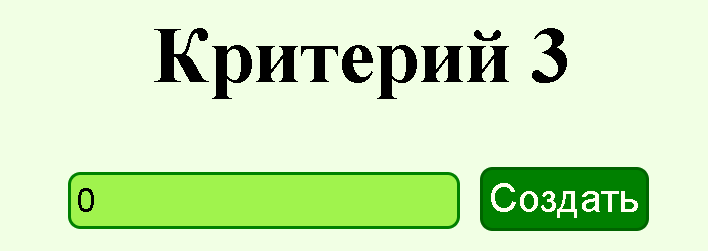
  
  
Изображение 3.9 — Форма ввода количества параметров

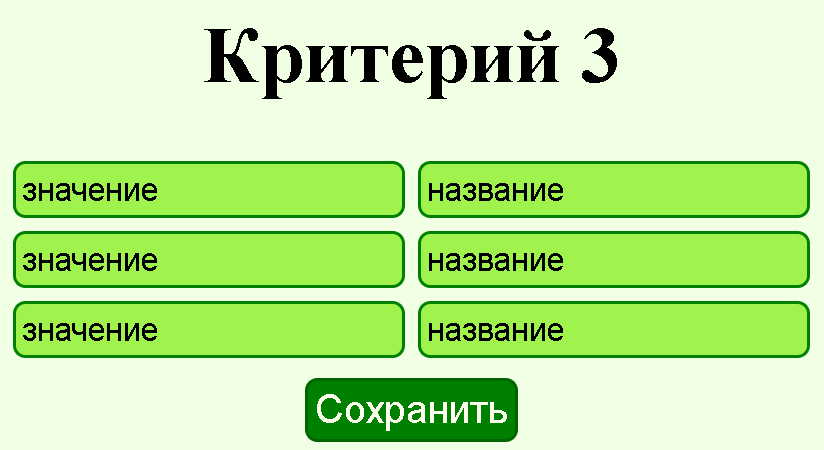
После ввода количества параметров необходимо ввести в таблицу значения для сравнения каждой пары параметров.

Верхнее поле каждого столбца предназначен для ввода названия параметра (например, «дождливо»/«солнечно»/и т.д.). Ниже в столбце расположены поля для попарного сравнения параметров. В таблицу, представленную на изображении 3.10, вводятся только значения >1. Обратные значения во избежание ошибочного ввода рассчитываются реализованной функцией и вносятся в готовом виде в структуру *criterions* в виде коллекции «параметр-оценка».

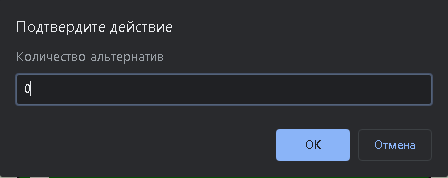
  
  
Изображение 3.10 — Заполнение таблицы попарного сравнения

3.4.3.3 Простое здание критериев реализован в форме ввода пар «степень принадлежности/параметр». Для этого необходимо ввести количество параметров, которые будут задаваться (изображение 3.11), а затем в появившиеся поля последовательно внести значения степени принадлежности параметра к заданному критерию и названия параметров (изображение 3.12).

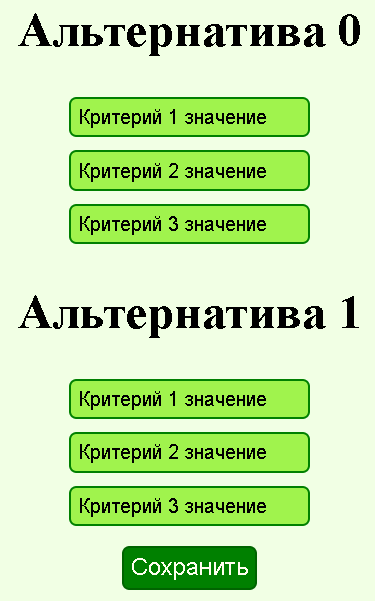
  
  
Изображение 3.11 — Поле для ввода количества параметров

  
  
Изображение 3.12 — Поля для параметров и значений их степеней принадлежности

**3.4.4**После задания критериев разного вида и параметров для этих критериев, пользователю предлагается перейти к созданию альтернатив, из которых будет выбираться одна наилучшая. Для этого необходимо в появившемся окне (изображение 3.13) ввести количество альтернатив, из которых в процессе работы системы будет выделена с максимальной суммарной оценкой по всем критериям.

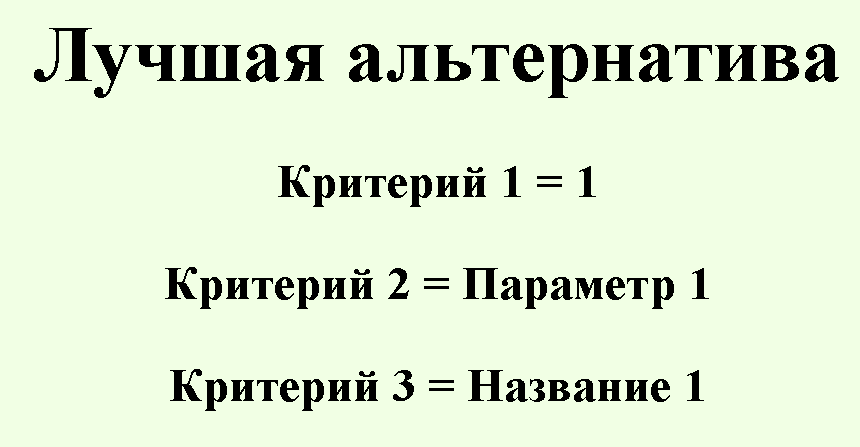
  
  
Изображение 3.13 — Окно ввода количества альтернатив

После задания количества альтернатив, формируются наборы полей для заполнения значений по каждому критерию для всех альтернатив. Форма для ввода данных представлена на изображении 3.14.

  
  
Изображение 3.14 — Форма ввода альтернатив

При нажатии на большую и красивую кнопку для подтверждения намерения выбрать наилучшую альтернативу (изображение 3.15) на экран выводится результат работы системы в виде информации о значениях параметров критериев наилучшей альтернативы. Вывод лучшей альтернативы представлен на изображении 3.16.

  
  
Изображение 3.15 — Кнопка подтверждения

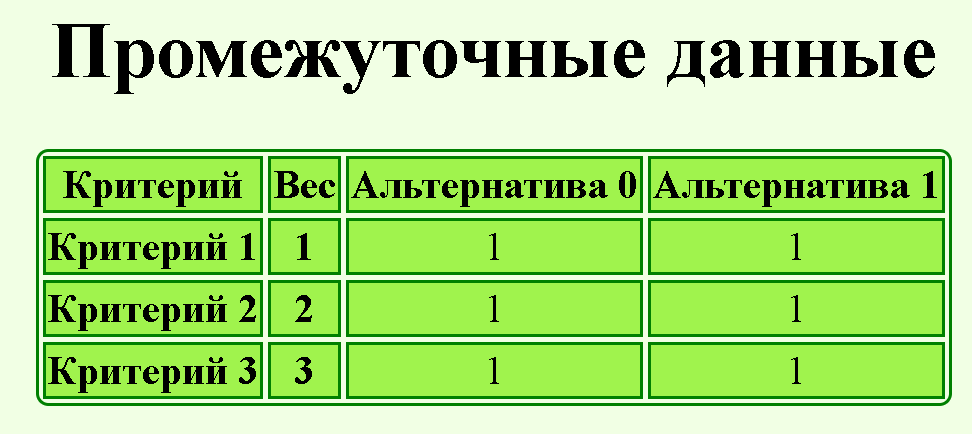
  
  
Изображение 3.16 — Вывод наилучшей альтернативы

Так как для реализации был выбран режим с выводом промежуточных расчетов, производимых в процессе работы системы и выбора наилучшей альтернативы, после вывода блока с информацией о результате решения на экран выводятся также таблицы со значениями всех совершенных алгоритмом расчетов.

Пользователю на экран выводится информация о введенных им альтернативах (изображение 3.17) вместе с названиями критериев и их весами

  
  
Изображение 3.17 — Таблица входных данных

Также выводятся таблица с альтернативами, приведенными к безразмерному виду и таблица со значениями весов альтернатив после учета значимости критериев. Таблицы представлены на изображениях 3.18, 3,19.

  
  
Изображение 3.18 — Таблица промежуточных данных

  
  
Изображение 3.17 — Таблица данных с итоговыми оценками альтернатив

## Контрольные примеры

Используем пример с выбором наилучшего решения о застройке местности для проверки корректности работы системы.

Существуют три альтернативы того, как можно построить здания. Они оцениваются по шести значимым критериям. Альтернативы представлены в виде таблицы 3.1.

Таблица 3.1 — Альтернативы застройки

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | Альтернатива 1 | Альтернатива 2 | Альтернатива 3 |
| Затраты | 7000 | 5000 | 8000 |
| Срок эксплуатации | 35 | 25 | 45 |
| Использование высокотехнологичных конструкций | 30 | 7 | 25 |
| Основной материал | Плита | Монолит | Плита |
| Подготовка площадки | Не требуется | Требуется | Не требуется |
| Удобство эксплуатации | Хорошо | Отлично | Отлично |

В данном примере есть шесть критериев, по которым можно оценивать альтернативы, и они задаются следующим образом:

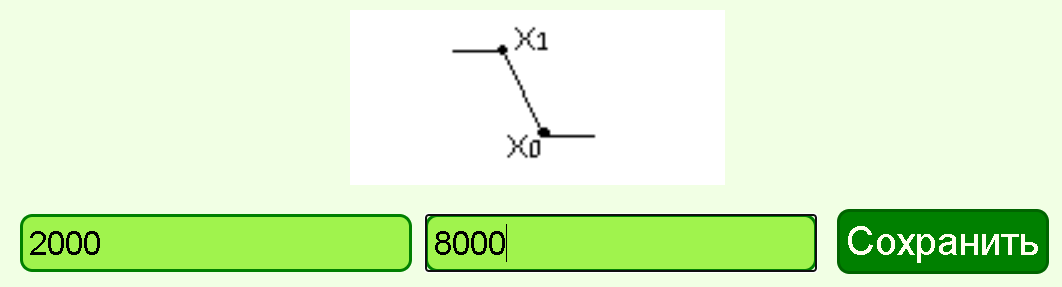
1. Затраты — функция принадлежности;
2. Срок эксплуатации — функция принадлежности;
3. Использование высокотехнологичных конструкций — функция принадлежности;
4. Основной материал — таблица попарных сравнений;
5. Подготовка площадки — таблица попарных сравнений;
6. Удобство эксплуатации — множество вида «степень принадлежности/параметр».

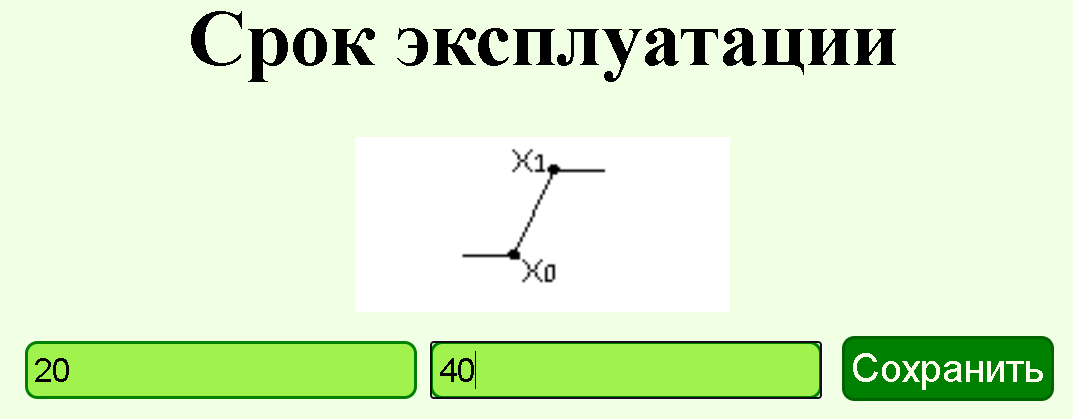
Веса критериев определены заранее на основе экспертных оценок и имеют следующие значения: .

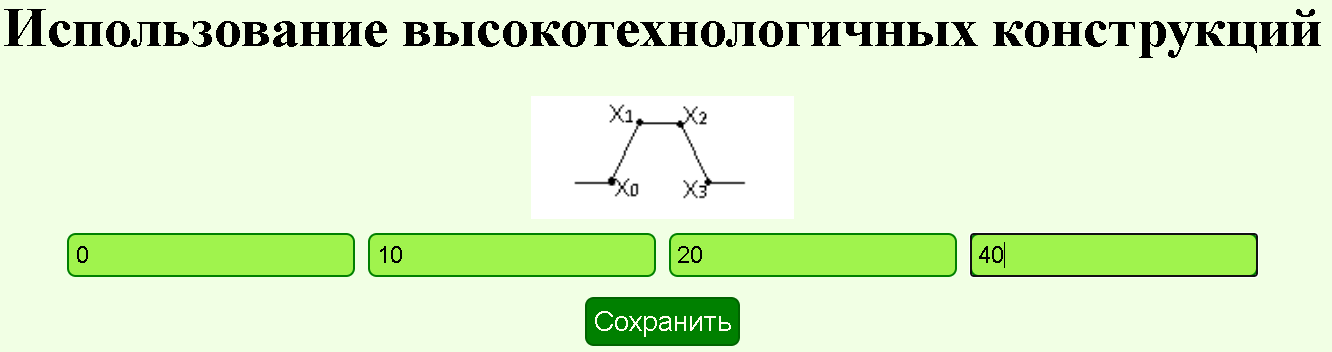
После ввода в систему количества критериев (изображение3.18 ) необходимо внести информацию о названиях критериев и их весах.

  
  
Изображение 3.18 — Задание количества критериев

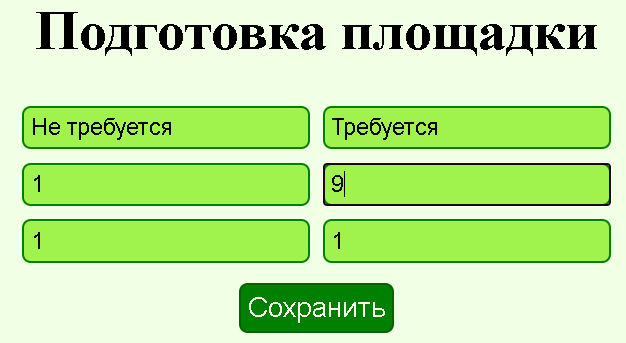
Важным этапом создание пользовательских критериев является задание формы ввода параметров. В данном примере задействованы все предложенные в системе формы задания (изображение 3.19-3.24).

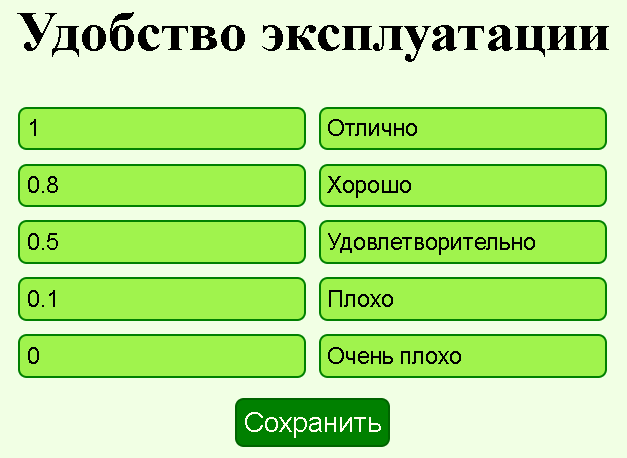
  
  
Изображение 3.19 — Задание функции принадлежности для затрат

  
  
Изображение 3.20 — Функция принадлежности срока эксплуатации

  
  
Изображение 3.21 — Функция принадлежности для третьего критерия

  
  
Изображение 3.22 — Таблица для критерия «Основной материал»

  
  
Изображение 3.23 — Таблица для критерия «Подготовка площадки»

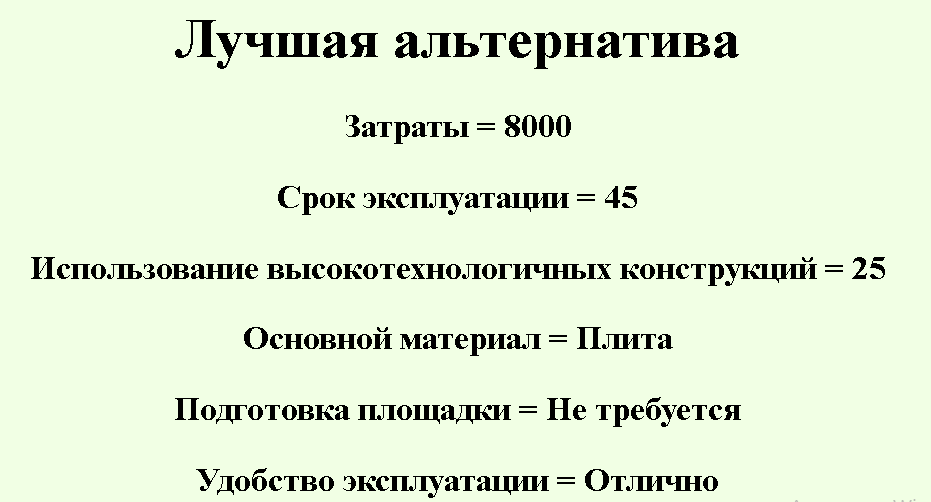
  
  
Изображение 3.24 — Таблица для критерия «Основной материал»

После создания пользовательских критериев необходимо внести в систему альтернативы, из которых на основе имеющихся данных должна быть выбрана одна наилучшая. Заполнение альтернатив представлено на изображении 3.25.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  | |

Р исунок 3.25 — Заполнение полей для альтернатив

После нажатия на кнопку на экран выводится информация о наилучшей альтернативе (изображение 3.26).

  
  
Изображение 3.26 — Вывод наилучшей альтернативы

Также для подтверждения корректности работы системы выводятся:

* таблица с данными об альтернативах и весах критериев, которые были введены пользователем (изображение 3.27),
* таблица с результатами промежуточных расчетов, производившихся в процессе работы системы (изображение 3.28)
* итоговая таблица с суммарной оценкой по каждой из альтернатив с учетом весов критериев (изображение 3.29).

  
  
Изображение 3.27 — Вывод входных данных на экран

  
  
Изображение 3.28 — Вывод промежуточных расчетов

  
  
Изображение 3.29 — Вывод итоговой таблицы с суммарными оценками

При проведении ручных расчетов по этим данным были получены таблицы приведения альтернатив к безразмерному виду (табл. 3.1) и итоговая таблица 3.2.

Таблица 3.1 — Альтернативы, приведенные к безразмерному виду

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | Альтернатива 1 | Альтернатива 2 | Альтернатива 3 |
| Затраты | 0,17 | 0,5 | 0 |
| Срок эксплуатации | 0,75 | 0,25 | 1 |
| Использование высокотехнологичных конструкций | 0,5 | 0,7 | 0,75 |
| Основной материал | 0,43 | 1 | 0,43 |
| Подготовка площадки | 1 | 0,11 | 1 |
| Удобство эксплуатации | 0,8 | 1 | 1 |

Таблица 3.2 — Итоговая таблица с суммарными оценками альтернатив

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | Альтернатива 1 | Альтернатива 2 | Альтернатива 3 |
| Затраты | 0,03 | 0,1 | 0 |
| Срок эксплуатации | 0,19 | 0,06 | 0,25 |
| Использование высокотехнологичных конструкций | 0,07 | 0,09 | 0,1 |
| Основной материал | 0,06 | 0,15 | 0,0,6 |
| Подготовка площадки | 0,12 | 0,01 | 0,12 |
| Удобство эксплуатации | 0,12 | 0,15 | 0,15 |
| Суммарная оценка | 0,59 | 0,57 | 0,68 |

Данные полученные в результате ручных расчетов и в процессе работы системы совпадают.

# ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ

## Характеристика программного средства

Целью разработки программного комплекса является создание системы способной поддерживать принятие решений на основе нечетких множеств. Комплекс является собственной разработкой, предполагает дальнейшее внедрение и использование в организации.

Проблема принятия решений на основе нечетких множеств является актуальной, т.к. многие понятия, связанные с человеческим мышлением и умозаключениями, являются нечеткими по своей природе и требуют соответствующего лингвистического описания.

Эффект, ожидаемый от использования программного комплекса, состоит в снижении трудоемкости работ для сотрудников, что в свою очередь приведет к экономии на заработной плате и приросту прибыли предприятия за счет сокращения расходов.

## Расчет инвестиций в разработку программного комплекса

Расчет затрат на основную заработную плату команды разработчиков осуществляется исходя из состава и численности команды, размера месячной заработной платы каждого участника команды, а также трудоемкости работ, выполняемых при разработке программного средства отдельными исполнителями

Основная заработная плата исполнителей проекта рассчитывается по формуле:

|  |  |
| --- | --- |
| , | (4.1) |

где Зо – основная заработная плата исполнителей, тыс. руб;

*n* − количество исполнителей, занятых разработкой ПО;

– часовая тарифная ставка i-го исполнителя, руб;

– количество часов работы, ч.

Расчетные данные для основной заработной платы приведены в таблице 4.1.

Таблица 4.1 — Расчет затрат на основную заработную плату команды разработчиков

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Категория  исполнителя | Месячная  заработная плата, р. | Часовая  заработная плата, р. | Трудоёмкость  работ, ч | Итого, р. |
| 1 | 2 | 3 | 4 | 5 |
| 1. Программист | 1 530,36 | 9,11 | 140 | 1 275,40 |
| 2. Тестировщик | 2 996,95 | 17,84 | 40 | 713,56 |
| Итого: | 4 527,31 | 26,95 | 180 | 1 988,96 |
| Всего затраты на основную заработную плату разработчиков | | | | 1 988,86 |
| Примечание — При расчете заработной платы использовалась среднемесячная заработная плата в Республике Беларусь для сотрудников различных категорий ИТ-отрасли за март 2020 года. | | | | |

Дополнительная заработная плата включает выплаты, предусмотренные законодательством о труде (оплата отпусков, льготных часов, времени выполнения государственных обязанностей и других выплат, не связанных с основной деятельностью исполнителей), и определяется по формуле:

|  |  |
| --- | --- |
|  | (4.2) |

где Зд − дополнительная заработная плата исполнителей, руб;

Нд − норматив дополнительной заработной платы, равный 10%.

Дополнительная заработная плата составит:

|  |  |
| --- | --- |
|  |  |

Отчисления в фонд социальной защиты населения и обязательное страхование определяются в соответствии с действующими законодательными актами по нормативу в процентном отношении к фонду основной и дополнительной зарплаты исполнителей, определенной по нормативу, установленному в целом по организации следующим образом:

|  |  |
| --- | --- |
|  | (4.3) |

где Нсоц ‒ ставка отчислений в ФСЗН и Белгосстрах (в соответствии с действующим законодательством по состоянию на 01.01.2020 г. ‒ 34,6 %).

Тогда отчисления на социальные нужды составят:

|  |  |
| --- | --- |
|  |  |

Прочие расходы рассчитываются по формуле:

|  |  |
| --- | --- |
|  | (4.4) |

где ‒ норматив прочих расходов, %.

В этом случае прочие расходы составят:

|  |  |
| --- | --- |
|  |  |

Общая сумма расходов на разработку программного продукта определяется по формуле:

|  |  |
| --- | --- |
|  | (4.5) |

Тогда общая стоимость разработки составит:

|  |
| --- |
|  |

Все расчеты сметы затрат и себестоимости ПС сведены в таблицу 4.2.

Таблица 4.2 – Смета затрат и себестоимость ПС

|  |  |  |
| --- | --- | --- |
| Наименование статьи затрат | Условное обозначение | Сумма, р. |
| 1. Основная заработная плата разработчиков |  | 1 988,96 |
| 2. Дополнительная заработная плата разработчиков |  | 298,34 |
| 3. Отчисления на социальные нужды |  | 688,18 |
| 4. Прочие расходы |  | 596,69 |
| 5. Общая сумма затрат (инвестиций) на разработку |  | 3 572,17 |

## Расчет экономического эффекта от использования программного средства

Разработка программного средства осуществляется с целью:

снижения трудоёмкости выполнения «ручных» операций и бизнес-процессов;

повышения качества сервиса.

Экономическим эффектом в результате использования программного средства является прирост чистой прибыли, полученный за счет:

‒ экономии на заработной плате и начислениях на заработную плату сотрудников за счет снижения трудоемкости работ.

Экономия на заработной плате и начислениях на заработную плату сотрудников за счет снижения трудоемкости работ рассчитывается по формуле:

|  |  |
| --- | --- |
|  | (4.6) |

где ‒ трудоемкость выполнения работ до внедрения ПС, ч;

‒ трудоемкость выполнения работ после внедрения ПС, ч;

‒ часовая тарифная ставка сотрудника, использующего программное средство, р.;

– плановый объем работ, выполняемых сотрудником;

– норматив дополнительной заработной платы, %;

– ставка отчислений от заработной платы, включаемых в себестоимость (34,6 %).

Прирост чистой прибыли, полученной за счет экономии на текущих затратах предприятия:

|  |  |
| --- | --- |
|  | (4.7) |

где – экономия на текущих затратах при использовании ПС, р.;

– прирост текущих затрат связанных с использованием ПС (затраты на сопровождение программного средства, затраты на интернет-трафик и т. п.), р.;

‒ ставка налога на прибыль согласно действующему законодательству, (по состоянию на 01.01.2020 г. – 18 %).

Результаты расчетов экономического эффекта от использования программного комплекса приведены в таблице 4.3.

Таблица 4.3 — Расчет экономического эффекта от использования программного средства

|  |  |  |
| --- | --- | --- |
| Экономический эффект | Условное обозначение | Итого, р. |
| 1. Экономия на заработной плате и начислениях на заработную плату сотрудников за счет снижения трудоемкости работ |  | 4 532,25 |
| 2. Прирост чистой прибыли, полученной за счет экономии на текущих затратах предприятия |  | 3 618,05 |

Таким образом, прирост прибыли, полученной за счет экономии на текущих затратах предприятия, составит 3618,05р. (*три тысячи шестьсот восемнадцать рублей пять копеек*) за год.

## Расчет показателей экономической эффективности разработки и использования программного средства

Оценка экономической эффективности разработки и использования программного комплекса зависит от результата сравнения затрат на его разработку и полученного экономического эффекта (годового прироста чистой прибыли).

Т.к. сумма инвестиций меньше суммы годового экономического эффекта, т. е. затраты окупятся менее чем за год, оценка экономической эффективности инвестиций в разработку программного комплекса осуществляется с помощью расчета простой нормы прибыли (рентабельности инвестиций) по формуле:

где ‒ прирост чистой прибыли, полученной от использования разработанного программного комплекса, р.;

‒ инвестиции (затраты) на разработку (модернизацию, совершенствование) программного средства, р.

## Вывод по результатам расчета

Общая сумма затрат на разработку составит 3572,17р. (*три тысячи пятьсот семьдесят два рубля семнадцать копеек*)

При этом прирост годовой прибыли, полученной за счет экономии на текущих затратах предприятия, составит 3618,05р. (*три тысячи шестьсот восемнадцать рублей пять копеек*).

При таких значениях рентабельность инвестиций принимает значение 101,28 %.

Таким образом разработка программного комплекса целесообразна и окупится менее, чем за год после разработки и внедрения.

# ЗАКЛЮЧЕНИЕ

Существует большой класс задач многокритериального альтернативного выбора. Во многих из них оценки критериального соответствия могут быть заданы только приблизительно, например, как степень принадлежности каждого параметра к данному критерию.

Аппарат теории нечетких множеств позволяет моделировать рассуждения человека. Необходимость на практике принимать решения в условиях неполной и нечеткой информации показывает, что теория нечетких множеств является мощным инструментов при решении сложных многокритериальных задач.

Целью дипломного проекта была разработка программного модуля поддержки принятия решения на основе нечетких множеств. Для функционирования данной системы было разработано два режима работы:

* режим выбора наилучшей альтернативы;
* режим выбора наилучшей альтернативы с выводом результата промежуточных расчетов.

Следствием использования программного комплекса является снижение трудоемкости работ для сотрудников, что в свою очередь ведет к экономии на заработной плате и приросту прибыли предприятия за счет сокращения расходов.

Одним из направлений дальнейшего развития является расширение перечня доступных режимов работы системы, добавление дополнительных функций для ввода данных, увеличение количества информации о принятии решения доступной пользователю.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Многокритериальный анализ решений [Элекстронный ресурс]. — Режим доступа: <https://www.hisour.com/ru/multiple-criteria-decision-analysis-35202/>
2. Бомас В.В., Судаков В.А. Поддержка субъективных решений в многокритериальных задачах. — М.: Изд-во МАИ, 2011. — 173 с.
3. Осипов В.П., Судаков В.А. Многокритериальный анализ решений при нечетких областях предпочтений // Препринты ИПМ им. М.В.Келдыша. 2017. № 6. — 16 с
4. Борисов, В. В. Основы теории нечетких множеств / В. В. Борисов, А. С. Федулов, М. М. Зернов. – М. : Горячая линия – Телеком, 2014. – Кн. 2. – 98 с.
5. Чернов, В. Г. Основы теории нечетких множеств : учеб. пособие / В. Г. Чернов ; Владим. гос. ун-т. – Владимир : Изд-во Владим. гос. ун-та, 2010. – 96 с.
6. Конышева, Л. К.Основы теории нечетких множеств: Учебное пособие / Назаров Д.М. — СПб.: Питер, 2011.— 192 с.
7. Intro: Fuzzy sets [Элекстронный ресурс]. — Режим доступа: https://www.youtube.com/watch?v=95Ydj1SoyuU
8. Соловьев, Н. А. Основы теории принятия решений для программистов: учебное пособие / Е. Н. Чернопрудова, Д. А. Лесовой; Оренбургский гос. ун-т. — Оренбург: ОГУ, 2012 — 187 с.
9. Доррер, Г. А. Теория принятия решений: Учебное пособие, Сибирский федеральный университет — Красноярск: ФГАОУ ВПО «Сибирский федеральный университет», 2013. — 180 с.
10. СППР [Элекстронный ресурс]. — Режим доступа: <http://sewiki.ru/%D0%A1%D0%9F%D0%9F%D0%A0>
11. Веселов Д.А. Задачи систем поддержки принятия решений и преимущества их использования // Современные научные исследования и инновации. 2015. № 4. Ч. 1 [Электронный ресурс]. URL: http://web.snauka.ru/issues/2015/04/51717.
12. Палицын, В. А. Технико-экономическое обоснование дипломных проектов: Метод. пособие для студ. всех спец. БГУИР. В 4–х ч. Ч. 4: Проекты программного обеспечения / В.А. Палицын. – Минск : БГУИР, 2006. – 76 с.
13. Носенко, А. А. Методические указания по технико-экономическому обоснованию дипломных проектов инженерного профиля/ А.А.Носенко. – Минск : БГУИР, 2011г. – 27 c.

# Приложение А (обязательное) Код обработки пользовательских данных (к подразделу 3.3)

*function tableNorm(){*

*for (i=0;i<criterions.length;i++){*

*if (criterions[i].get("criterionType")=="table"){*

*for (j=0;j<criterions[i].get("paramValues").length;j++){*

*for (k=0;k<criterions[i].get("paramValues").length;k++){*

*((criterions[i].get("paramValues"))[k][j]>1){*

*(criterions[i].get("paramValues"))[j][k]=1/(criterions[i].get("paramValues"))[k][j]*

*}}}*

*var temp=new Array();*

*sum=0;*

*for (j=0;j<criterions[i].get("paramValues").length;j++){*

*temp[j]=1;*

*for (k=0;k<criterions[i].get("paramValues").length;k++){*

*temp[j]\*=(criterions[i].get("paramValues"))[k][j];*

*}*

*temp[j]=Math.pow(temp[j],(1/criterions[i].get("paramValues").length));*

*sum+=temp[j];}*

*for (j=0;j<criterions[i].get("paramValues").length;j++){*

*temp[j]=(temp[j]/sum);}*

*maxT=Math.max.apply(null,temp);*

*for (j=0;j<criterions[i].get("paramValues").length;j++){*

*temp[j]=(temp[j]/maxT);}*

*for (j=0;j<criterions[i].get("paramValues").length;j++){*

*criterions[i].set((criterions[i].get("paramnames"))[j],temp[j]);*

*}}}}*

*function altNorm(){*

*for (i=0;i<alternatives.length;i++){*

*alternativeCompare[i]=new Array();*

*resaltTable[i]=new Array();}*

Продолжение приложения А

*for (j=0;j<criterions.length;j++){*

*if (criterions[j].get("criterionType")=="funcO"){*

*for (i=0;i<alternatives.length;i++){*

*cname=criterions[j].get("name");*

*avalue=alternatives[i].get(cname);*

*if (avalue<(criterions[j].get("paramValues")[0])){*

*alternativeCompare[i][j]=0;*

*} else if (avalue>(criterions[j].get("paramValues")[1])){*

*alternativeCompare[i][j]=1;*

*} else {*

*x=criterions[j].get("paramValues");*

*newavalue=((avalue-x[0]\*(1-0)))/(x[1]-x[0]);*

*alternativeCompare[i][j]=newavalue;}}*

*} else if (criterions[j].get("criterionType")=="funcTw"){*

*for (i=0;i<alternatives.length;i++){*

*cname=criterions[j].get("name");*

*avalue=alternatives[i].get(cname);*

*if (avalue<(criterions[j].get("paramValues")[0])){*

*alternativeCompare[i][j]=1;*

*} else if (avalue>(criterions[j].get("paramValues")[1])){*

*alternativeCompare[i][j]=0;*

*} else {*

*x=criterions[j].get("paramValues");*

*newavalue=(((avalue-x[0])\*(0-1))/(x[1]-x[0]))+1;*

*alternativeCompare[i][j]=newavalue;*

*}}*

*} else if (criterions[j].get("criterionType")=="funcTh"){*

*for (i=0;i<alternatives.length;i++){*

*cname=criterions[j].get("name");*

*avalue=alternatives[i].get(cname);*

*if ((avalue<(criterions[j].get("paramValues")[0])) ||*

*(avalue>(criterions[j].get("paramValues")[3]))){*

*alternativeCompare[i][j]=parseInt(0);*

*} else if ((avalue<(criterions[j].get("paramValues")[2])) && (avalue>(criterions[j].get("paramValues")[1]))){*

*alternativeCompare[i][j]=parseInt(1);*

Продолжение приложения А

*} else if (avalue<=(criterions[j].get("paramValues")[1])) {*

*x=criterions[j].get("paramValues");*

*newavalue=((avalue-x[0]\*(1-0)))/(x[1]-x[0]);*

*alternativeCompare[i][j]=newavalue;*

*} else {*

*x=criterions[j].get("paramValues");*

*newavalue=(((avalue-x[2])\*(0-1))/(x[3]-x[2]))+1;*

*alternativeCompare[i][j]=newavalue;*

*}}*

*} else {*

*for (i=0;i<alternatives.length;i++){*

*cname=criterions[j].get("name");*

*value=alternatives[i].get(cname);*

*avalue=criterions[j].get(value);*

*alternativeCompare[i][j]=avalue;*

*}}*

*for (i=0;i<alternatives.length;i++){*

*resaltTable[i][j]=alternativeCompare[i][j]\*criterionWeights[j];*

*}}*

*var maxI=0;*

*var maxV=0;*

*for (i=0;i<alternatives.length;i++){*

*rez[i]=0;*

*for (j=0;j<criterions.length;j++){*

*rez[i]+=resaltTable[i][j];}*

*if (maxV<rez[i]){*

*maxV=rez[i];*

*maxI=i;*

*}}*

*return maxI;}*

*function range(){*

*tableNorm();*

*var bestI=altNorm();*

*document.getElementById("rangeAlts").remove();*

*var newField = document.createElement("div");*

Продолжение приложения А

*newField.innerHTML='<h1>Лучшая альтернатива</h1>';*

*for(i=0;i<criterions.length;i++){*

*var critName=criterions[i].get("name");*

*var critValue=alternatives[bestI].get(critName);*

*newField.innerHTML+='<h3>'+critName+' = '+critValue+'</h3>';*

*}*

*document.body.appendChild(newField);*

*showTables();}*

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ВЕДЕМОСТЬ ДОКУМЕНТОВ | | | | | | | | | | | |
| *Обозначение* | | | | | *Наименование* | *Дополнительные сведения* | | | | | |
|  | | | | | *Текстовые документы* |  | | | | | |
|  | | | | |  |  | | | | | |
| *БГУИР ДП 1-53 01 02 06 006ПЗ* | | | | | *Пояснительная записка* | *66 с.* | | | | | |
|  | | | | | *Отзыв руководителя* |  | | | | | |
|  | | | | | *Рецензия* |  | | | | | |
|  | | | | |  |  | | | | | |
|  | | | | | *Графические документы* |  | | | | | |
|  | | | | |  |  | | | | | |
| *ГУИР.000000.001 ПД* | | | | | *Структура программного* |  | | | | | |
|  | | | | | *комплекса* | *Формат А1* | | | | | |
|  | | | | |  |  | | | | | |
| *ГУИР.000000.002 ПД* | | | | | *Схема алгоритма* |  | | | | | |
|  | | | | | *обработки пользовательских* |  | | | | | |
|  | | | | | *данных* | *Формат А1* | | | | | |
|  | | | | |  |  | | | | | |
| *ГУИР.000000.003 ПЛ* | | | | | *Диаграмма вариантов* |  | | | | | |
|  | | | | | *Использования системмы* | *Формат А1* | | | | | |
|  | | | | |  |  | | | | | |
| *ГУИР.000000.004 ПЛ* | | | | | *Диаграмма деятельности* | *Формат А1* | | | | | |
|  | | | | |  |  | | | | | |
| *ГУИР.000000.005 ПЛ* | | | | | *Диаграмма состояний* | *Формат А1* | | | | | |
|  | | | | |  |  | | | | | |
| *ГУИР.000000.006 ПЛ* | | | | | *Интерфейс программы* |  | | | | | |
|  | | | | | *(Ввод данных)* | *Формат А1* | | | | | |
|  | | | | |  |  | | | | | |
|  | | | | |  |  | | | | | |
|  | | | | |  |  | | | | | |
|  | | | | |  |  | | | | | |
|  |  |  |  |  | *БГУИР ДП 1-53 01 02 06 006 Д1* | | | | | | |
|  |  |  |  |  |
|  |  |  |  |  |
| *Изм.* | *Л.* | *№ докум.* | *Подп.* | *Дата* | *Программный комплекс поддержки принятия решений на основе нечетких множеств* | | *Лит* | | | *Лист* | *Листов* |
| *Разраб.* | | *Дишук* |  |  |  | *Т* |  | *66* | *66* |
| *Провер.* | | *Валяк* |  |  | *Кафедра ИТАС*  *гр. 620601* | | | | |
| *Т.контр* | | *Гончаревич* |  |  |  | |
| *Н.контр* | | *Протченко* |  |  |  | |
| *Утв.* | | *Навроцкий* |  |  |  | |