

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
ДОНЕЦКОЙ НАРОДНОЙ РЕСПУБЛИКИ**



ИНФОРМАТИКА И КИБЕРНЕТИКА

4(10)

Донецк – 2017

УДК 004.3+004.9+004.2+51.7+519.6+519.7

ИНФОРМАТИКА И КИБЕРНЕТИКА, № 4(10), 2017,
Донецк, ДонНТУ.

Представлены материалы по вопросам приоритетных направлений научно-технического обеспечения в области информатики, кибернетики и вычислительной техники.

Материалы предназначены для специалистов народного хозяйства, ученых, преподавателей, аспирантов и студентов высших учебных заведений.

Редакционная коллегия

Главный редактор: Павлыш В.Н., д.т.н., проф.

Зам. глав. ред.: Андрюхин А.И., к.т.н., с.н.с.

Ответственный секретарь: Завадская Т.В., к.т.н., доц.

Члены редакционной коллегии: Аверин Г.В., д.т.н., проф. Аноприенко А.Я., к.т.н., проф. Григорьев А.В., к.т.н., доц., Грунский И.С., к.т.н., проф., Зинченко Ю.Е., к.т.н., доц., Звягинцева А.В., к.т.н., доц., Миненко А.С., д.т.н., проф., Скобцов Ю.А., д.т.н., проф., Шелепов В.Ю., д.т.н., проф., Карабчевский В.В., к.т.н., доц. Федяев О.И., к.т.н., доц.

Рекомендовано к печати ученым советом ГОУ ВПО «Донецкий национальный технический университет» Министерства образования и науки ДНР. Протокол № 11 от 22.12.2017.

Электронное периодическое издание «Научный журнал «Информатика и кибернетика» зарегистрирован в Министерстве информации ДНР.

Свидетельство о регистрации: серия ААА № 000145 от 20.06.2017.

Контактный адрес редакции

ДонНТУ, ул. Артема, 58, Донецк, 83001

Тел.: +380 (62) 301-08-56. Эл.почта: infcyb.donntu@yandex.ru

Интернет: <http://infcyb.donntu.org>

© Донецкий национальный технический университет
Министерство образования и науки ДНР, 2017

СОДЕРЖАНИЕ

Рефлексивные представления Андрюхин А.И	4
Компьютерное решение проблем. Ассоциативный поиск и рефлексивный вывод в когнитивных системах на основе SQL и NoSQL СУБД.I. Андрюхин А.И, Новиков Е.Т.....	13
Статистические компьютерные оценки характеристик многомерных операций и объектов.II. Артеменко В.А., Андрюхин А.И.,Филер З.Е.....	27
Алгоритмы размещения файлов в компьютерных сетях Д.В. Бельков.....	39
Усовершенствование композиционного устройства управления с оптимальной адресацией микрокоманд К.Н. Ефименко.....	45
Модельный базис оценки риска взаимодействия в архитектуре партнерской сети Казакова Е.И., Коломыцева А.О.....	52
Программная система для оценки качества настройки струнного инструмента Леснов Е.В., Андрюхин А.И.....	57
Разработка программного обеспечения для формирования оптимальных маршрутов быстрой доставки товаров С.А. Макогон, О.Д. Ситникова.....	63
Исследование параметров гидросистемы механизированных крепей методом математического моделирования. Павлыш В.Н., Зензеров В.И.....	71
Эмулятор системы диагностики теплового состояния кристаллизатора МНЛЗ в среде LabVIEW. А.Ю. Харитонов, А.Б. Бирюков, А.В. Алиференко, А.В. Сныткина	78
Подсистема защиты авторского права в сфере программного обеспечения А.В. Чернышова, Д.В. Кубашевский.....	86
Content.....	91

Рефлексивные представления

Андрюхин А.И., к.т.н,
Донецкий национальный технический университет, г. Донецк,
alexandruckin@rambler.ru

Андрюхин А.И. Рефлексивные представления. Целью исследования является применение рефлексивных моделей несовместных фактов, описываемых булевыми уравнениями от конечного числа переменных. Представлены все несовместные системы булевых уравнений от двух переменных. Построены все несовместные системы булевых уравнений от трех переменных. Приведены все возможные рефлексивные представления несовместных систем булевых уравнений от двух переменных. Построены все возможные рефлексивные представления несовместных систем булевых уравнений от трех переменных. Приведены результаты компьютерных расчетов. В расчетах использовался пакет Mathematica.

Ключевые слова: рефлексия, ранг рефлексии, парадоксы, модель, булевы функции

Введение

Известное утверждение «Познай себя и ты познаешь весь мир» имело в дальнейшем развитии человеческой мысли различные модификации. Широко известно высказывание Канта о звездном небе и нравственном законе у людей, однако для целей статьи необходимо более подробно рассмотреть воззрение Д.Локка[1] и также известные в истории человеческой познания положения таких авторов, как Фа Цзан (буддийская школа хуаянь)[2], Аристотель[3], Секст Эмпирик[4].

Так в работах влиятельного английского философа Д.Локка (напомним, создатели столь повлиявшей на исследование языка системы Margie из лаборатории ИИ в Стэнфорде разделяли его взгляды [5]) подчеркиваются следующие важные моменты наблюдений и рассуждений.

Так Локк считал, что первым шагом научного исследования является исследование собственного разума, т.е. использование рефлексии.

Под умом человека, его психикой Локк подразумевал некий присущий человеку аппарат, который подобно зрительной сенсорной системе, воспринимает идеи.

Ибо, на мой взгляд, первый шаг к разрешению различных вопросов, с которыми почти наверняка должна была столкнуться человеческая душа, состоит в исследовании нашего собственного разума, изучении своих собственных сил и усмотрении того, к чему они применимы. На мой взгляд, до тех пор, пока этого не было сделано, мы начинали не с того конца и напрасно искали удовлетворения в спокойном и надежном обладании наиболее важными для нас истинами в то самое время, как пускали свои мысли в обширный океан бытия, как будто бы все это бесконечное пространство является естественным и несомненным владением нашего разума, в котором ничто не избегает его определений, ничто не ускользает от его понимания.

Заметим, что, согласно следующему фрагменту, Локк согласен с непознаваемостью внешнего мира.

Если, таким образом, люди, направляя свои исследования за пределы своих способностей, пускают свои мысли странствовать по таким глубинам, где они не достают твердой опоры, то не удивительно, что они поднимают вопросы и умножают споры, которые никогда не приводят ни к какому исполному решению, а только поддерживают и увеличивают их сомнения и в конце концов утверждают их в абсолютном скептицизме. А между тем, если бы как следует были изучены способности нашего разума, выявлены пределы нашего познания и найдены границы освещенной и темной части вещей, постигаемой и непостижимой нами, люди примирились бы с открыто признанным невежеством одной части и с большей пользой и удовлетворенностью обратили бы свои мысли и рассуждения на другую.

Необходимо указать, что объем понятия идея у Локка чрезвычайно велик. Любой объект внутренней когнитивной деятельности человека он считал возможным характеризовать этим словом.

8. *Что означает слово «идея».* Вот что считал я нужным сказать о поводе к этому исследованию о человеческом разуме. Но прежде чем перейти к своим мыслям об этом предмете, я должен здесь, во вступлении, попросить у читателя прощения за частое употребление слова «идея» в последующем изложении. Так как этот термин, на мой взгляд, лучше других обозначает все, что является объектом мышления человека, то я употребил его для выражения того, что подразумевают под словами «фантом», «понятие», «вид», или всего, чем может быть занята душа во время мышления. И я не мог избежать частого употребления его.

Я думаю, со мною легко согласится в том, что такие идеи есть в человеческой душе. Каждый познает их в себе, а слова и действия других убеждают его в том, что они есть и у других.

Некоторые положения Локка являются спорными с позиции знаний сегодняшнего времени. Так рассматривая следующий фрагмент[1]:

5. Все наши идеи происходят или из одного, или из другого источника. Ни из какого другого источника, кроме указанных двух, разум, на мой взгляд, не получает ни малейших проблесков идей. Внешние вещи доставляют уму идеи чувственных качеств, которые все суть вызываемые в нас вещами различные восприятия, а ум снабжает разум идеями своей собственной деятельности. напомним известный вывод Фреге о непринадлежности человеческой мысли ни внешнему миру, ни внутреннему миру человека, сделанный им прилогическом исследовании человеческой мысли. Этим выводом Фреге указывает на более сложное построение мира, нежели это принято в обыденном сознании [6]. ДжЛокк однако, часто повторяет тезис о двух источниках человеческой информации.

4. Деятельность нашего ума — другой из источников. Во-вторых, другой источник, из которого опыт снабжает разум идеями, есть внутреннее восприятие действий (operations) нашего ума, когда он занимается приобретенными им идеями. Как только душа начинает размышлять и рассматривать эти действия, они доставляют нашему разуму (understanding) идеи другого рода, которые мы не могли бы получить от внешних вещей. Таковы «восприятие», «мышление», «сомнение», «вера», «рассуждение», «познание», «желание» и все различные действия нашего ума (mind). Когда мы сознаем и замечаем их в себе, то получаем от них в своем разуме такие же отличные друг от друга идеи, как и те, которые мы приобретаем от тел, действующих на наши чувства. Этот источник идей каждый человек целиком имеет внутри себя, и, хотя этот источник не есть чувство, поскольку не имеет никакого дела с внешними предметами, тем не менее он очень сходен с ним и может быть довольно точно назван внутренним чувством. Но, называя первый источник *ощущением*, я называю второй *рефлексией*, потому что он доставляет только такие идеи, которые приобретаются умом при помощи размышления о своей собственной деятельности внутри себя. Итак, мне бы хотелось, чтобы поняли, что под *рефлексией* в последующем изложении я подразумеваю то наблюдение, которому ум подвергает свою деятельность и способы ее проявления, вследствие чего в разуме возникают идеи этой деятельности. Эти два источника, повторяю я, т. е. внешние материальные вещи, как объекты *ощущения* и внутреннего деятельность нашего собственного ума как объект *рефлексии*, по-моему, представляют собой единственное, откуда берут начало все наши идеи.

Локк указывает на иерархичность рефлексии согласно авторского рис.1, на котором рефлексивному процессу осознания собственного сознания приписывается больший ранг рефлексии.

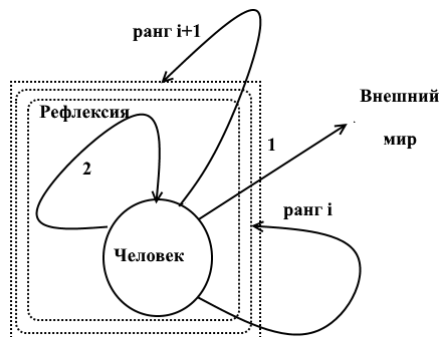


Рисунок 1 – Иерархичность рефлексии

Фрагмент текста Локка представлен ниже:

1. Простые идеи рефлексии — это действия ума в отношении его других идей. Получая извне упомянутые в предыдущих главах идеи, душа, обращая свой взор вовнутрь, на себя, и наблюдая свои действия в отношении этих приобретенных ею идей, получает отсюда другие идеи, которые так же способны быть объектами ее созерцания, как и идеи, воспринимаемые от внешних вещей.

2. Идею восприятия и идею воли мы получаем от рефлексии. Два основных главных вида деятельности души, которые чаще всего исследуются и настолько часто встречаются, что каждый, кто желает, может заметить их в себе, — это *восприятие*, или *мышление*¹⁶, и *желание*, или *хотение*. Сила мышления называется *разумом*, а сила желания называется *волею*; обе эти силы ума именуются *способностями*. О некоторых модусах данных простых идей рефлексии, таких, как *вспоминание*, *различение*, *рассуждение*, *суждение*, *познавание*, *вера* и т. д., я буду иметь случай поговорить позже¹⁷.

Подчеркнем известный факт большого влияния Локка на Шопенгауэра и то, что в основном произведении А.Шопенгауэра «Мир, как воля и представление» выполняется разработка принципов, высказанных Локком.

Постановка задачи

В настоящее время достаточно укоренившимся является мнение, что функционирование интеллектуальной системы должно быть представлено не менее чем двумя подсистемами. Первая подсистема в нашем сознании ассоциируется с символично-логическими операциями (Д.Буль считал, что анализируя и исследуя операции над символами, он исследует законы мышления), вторая выполняет динамические преобразования над модельными ситуациями, т.е. выполняет физическое моделирование.

В данной работе выполняется попытка представления текущей ситуации символическими отношениями между элементами описания (в число которых входят как свойства, так и отношения между элементами наблюдений сознания). Ясно, что здесь необходимо прозрачное понимание и решение проблемы символа, т.е. как с помощью символов, обладающих в нашем сознании определенными свойствами представлять динамику и другие свойства (отличные от свойств символов) наблюдаемого и ненаблюдаемого. Естественно понятно, что следует ожидать в нашем описании элементов наблюдаемого мира парадоксальных для субъекта фактов и ситуаций.

Предлагается обобщающее рефлексивное описание (если возможно) и следуемый из него рефлексивный вывод для «парадоксальных» или несовместных логических фактов (отношений). Следующим пунктом исследований является переход от булевой логики к вероятностной логике, которая присуща человеческому мышлению.

Здесь мы применяем известные формулы и правила для интерпретации наблюдаемых фактов на основе [7].

Теория причинности в буддийской логике

В раннем буддизме теория причинности, согласно которой возникновение и существование любого явления находится в зависимости от другого явления, занимает центральное место (см.рис.2а).Однако в нем не анализируется источник причинной зависимости. Для авторитетной школы буддийской школы хуаянь играл большую роль четвертый пункт 12 –звенной цепи причинной зависимости “Из сознания возникают имя и форма”.

Необходимо подчеркнуть, что в этой школе считали справедливым положение порождения любого явления другим явлением только совместно с сознанием (см. рис.2б). Необходимо наличие двух составляющих для причины чего-либо, т.е. сознания и объекта внешнего мира, причем первое(сознание) есть основная причина, а второй(объект) является дополнительной.

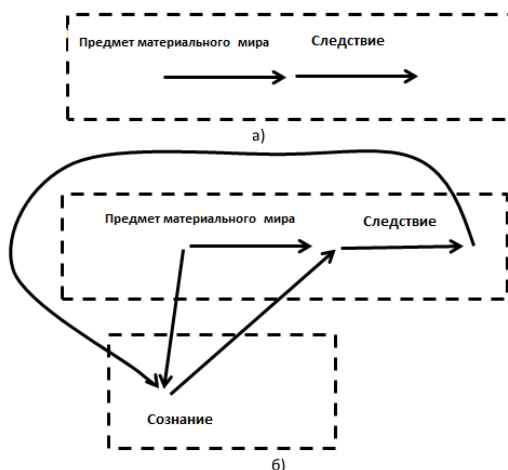


Рисунок 2 – а) порождения любого явления в раннем буддизме;б) две составляющих для причины чего либо

Рефлексия

По определению, под ней понимаем принцип человеческого мышления, направляющий его на осмысление и осознание собственных форм и предпосылок;

Рефлексия способствует формированию самосознания, «оборачивая» сознание само на себя.

Многие учёные, философы считают, что рефлексия отличает человека от животного, и рассматривают её как универсальный способ анализа самосознания.

Так, Тейяр де Шарден усматривал основное различие между человеком и животным в степени развития рефлексии.

«Для окончательного решения вопроса о «превосходстве» человека над животными ... я вижу только одно средство.....рассмотреть центральный феномен — рефлексию» [8].

С появлением рефлексивности, свойства в сущности элементарного (по крайней мере, вначале!), все меняется, и мы замечаем, что под более яркой реальностью коллективных преобразований скрытно происходило параллельное движение к индивидуализации.»

В этой работе Тейяр де Шарден подчеркивал, что «Под свободным и изобретательным воздействием сменяющих друг друга разумных существ со всей очевидностью нечтонеобратимо накапливается и передается, по крайней мере коллективно, путем воспитания, в ходе веков. Но это «нечто» — материальное сооружение или творение красоты, системы мысли или системы действия — в конечном счете всегда выражается в увеличении сознания, а сознание, в свою очередь, теперь мы это знаем, — не что иное, как субстанция и кровь развивающейся жизни.

Это означает, что, кроме феномена, относящегося к одному лицу — индивидуального подступа к рефлексии, — наука должна признать наличие феномена, также имеющего рефлексивную природу, но охватывающего целиком все человечество!». Это замечание раскрывается в [9].

Согласно В.Лефевру, справедливым можно считать, что человеку от рождения даны фундаментальные рефлексивные структуры с двумя рангами рефлексии (субъект чувствует себя и чувствует себя чувствующим себя) и автоматическим механизмом счета [10,11].

Рефлексивные булевы функции

Важность рассмотрения рефлексивных булевых функций обусловлена тем, что реализации сознания относится к базовым проблемам науки и любые прорывы в ней повлекут за собой резкое изменение в жизни человечества. Основная задача работы заключается в проверке свойств элементов рефлексивной булевой логики быть базовыми элементами сознания системы мозга. Подчеркнем, что система, построенная из исследуемых рефлексивных булевых элементов, должна сама быть рефлексивной.

Тогда при таком подходе, системы подобного рода является самоподобными и рефлексивными, и следовательно, обладать

фрактальными свойствами, т.е. теми свойствами, которые демонстрирует человеческий мозг.

Напомним, что согласно [12-14] рефлексивные булевы функции i -ого порядка обладают свойством

$$\Phi(x_1, x_2, \dots, x_{i-1}, \Phi(x_1, x_2, \dots, x_n), x_{i+1}, \dots, x_n) = \Phi(x_1, x_2, \dots, x_n).$$

Обозначим через $\Phi_{i,j}$ множество рефлексивных булевых функций i -порядка от j переменных. Также обозначим через Φ_j множество рефлексивных булевых функций от j переменных.

Задача определения рефлексивных булевых функций зависящих от 2, 3, 4 и более переменных решена в [12-13].

По аналогии с рефлексивными булевыми функциями рассматриваются негативные рефлексивные булевы функции i -ого порядка, которые обладают свойством $N(x_1, x_2, \dots, x_{i-1}, \neg N(x_1, x_2, \dots, x_n), x_{i+1}, \dots, x_n) = N(x_1, x_2, \dots, x_n)$.

Обозначим через $N_{i,j}$ множество негативных рефлексивных булевых функций i -порядка от j переменных. Также обозначим через N_j множество негативных рефлексивных булевых функций от j переменных.

В [13] выполнен анализ таких важных операций, как импликация и правило логического вывода «modus ponens» (MP), как представителей рефлексивных булевых функций.

Для определения рефлексивных булевых и негативных рефлексивных булевых функций и их характеристик был использован пакет Wolfram Mathematica 11.1.1, и в частности оператор BooleanFunction. Основные результаты для булевых функций от четырех переменных представлены в таблице 1. Под полными понимаем функции, в записи которых присутствуют все 4 переменные.

Таблица 1. Свойства функций от 4 переменных

Номер	Порядки	Рефлексивные		Негативно-рефлексивные	
		Все	Полные	Все	Полные
1	1	6293	5860	6225	5792
2	2	5826	5397	5884	5455
3	3	5946	5513	5946	5513
4	4	5946	5513	5946	5513
5	1,2	1276	1088	1264	1076
6	1,3	1238	1047	1240	1054
7	1,4	1276	1088	1272	1084
8	2,3	1166	982	1202	1018
9	2,4	1232	1046	1254	1068
10	3,4	1285	1097	1285	1097
11	1,2,3	396	302	396	302
12	1,2,4	396	302	396	302
13	1,3,4	398	304	398	304
14	2,3,4	386	292	400	306
15	1,2,3,4	168	114	168	114

Время определения рефлексивных и негативно-рефлексивных булевых функций из общего числа 65536 всех булевых функций от четырех переменных заняло 50 минут счета на процессоре Intel Core i3-4170 с частотой 3.7 ГГц.

Расчеты для функций от пяти и тем более 6, 7... переменных являются в настоящее время практически невозможными.

Рефлексивные представления и описания целей и возможностей конкурентов

Обозначим через рефлексивные представления случаи, когда система S_i может иметь модель представлений о ситуации других систем $S_j (j \neq i)$ и будем записывать их как $J_i (J_j(\alpha))$.

Такие модели могут использоваться для более точной оценки ситуации. Эти модели являются отражением системы S_i о степени осведомленности системы S_j .

Цели и возможности A_i и M_i для системы S_i обычно не совпадают с собственным представлением своих целей и возможностей A_i^i и M_i^i . Система S_i может определить свои возможности M_i^i по воздействию на внешнюю среду для достижения определенных состояний систем и объектов.

На базе модели $J_i(J_j(\alpha))$ система S_i может оценить возможности системы S_j , которое обозначим через M_i^j .

Цель для целеустремленной системы можно определить как некоторые желанные ситуации. Для системы S_i цель есть некоторая совокупность предложений, которые описывают желаемое свойство ситуации.

$Aim_{J_m}^{J_m-1, J_m-2, \dots, J, I}$ – информационная модель совокупности предложений, которые описывают представление i -ой системы о целевых установках системы S_{J_m} . Рефлексивные представления о целях и возможностях оппонентов представлены на рис.3-рис.5.

Знания системы S_1

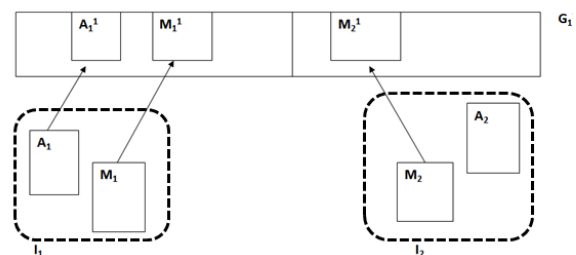


Рисунок 3 – Рефлексия 0-ранга

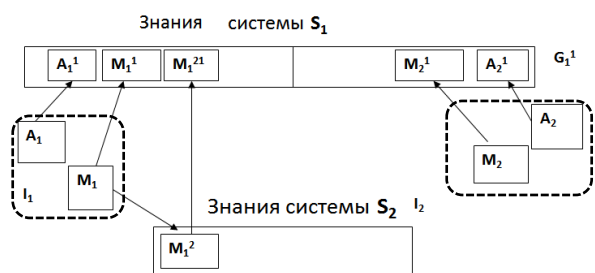


Рисунок 4 – Рефлексия 1-ранга

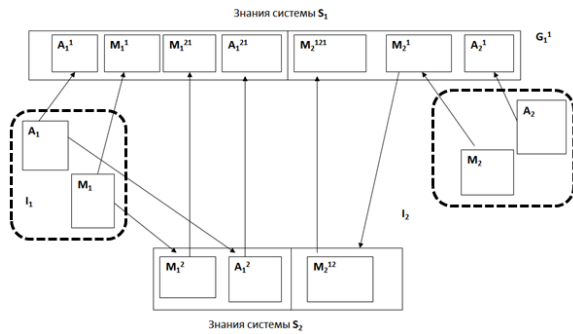


Рисунок 5 – Рефлексия 2-ранга

Рефлексивный вывод. Парадоксы и вероятностная интерпретация.

Рассмотрим простой пример, который иллюстрирует основные моменты и свойства рефлексивного обобщающего вывода и его вероятностного аналога.

Имеем два несовместных булевых уравнения $X1 \wedge X2 = 1$, $\neg(X1 \vee X2) = 1$ (система этих булевых уравнений не имеет решений).

Их левые части обозначим следующим образом $\psi1(X1, X2) = X1 \wedge X2$, $\psi2(X1, X2) = \neg(X1 \vee X2)$.

Рассмотрим булеву функцию $f(X1, X2, X3) = X3 \wedge \psi1(X1, X2) \vee \neg X3 \wedge \psi2(X1, X2)$ равную True, т.е. используем известную формулу разложения булевой функции более высокой размерности по дополнительным переменным.

Имеем, что при $X3=1$ истинна $\psi1(X1, X2)$, а при $X3=0$ истинна $\psi2(X1, X2)$.

Замечаем, что $f(X1, X2, X3) = f(f(X1, X2, X3), X2, X3)$.

Следовательно, она является рефлексивной функцией 1-го порядка.

Таким образом мы можем получить обобщающие представления и для булевых функций более высоких размерностей. Ясно, что их будет $m!$ для n уравнений вида $f_i(x_1, x_2, \dots, x_n) = \text{True}, i=1, n$. Обычно рассматриваем минимальное $m = \lceil \log_2 n \rceil + 1$, хотя можем использовать и большие значения.

Если обобщение мы всегда можем получить, используя известную формулу разложения булевой функции по нескольким переменным, то ответ на вопрос о существовании рефлексивного обобщающего представления для произвольной несовместной системы остается неясным.

Интерпретация правил использования вероятностных формул представлено в таб.2-3.

Основными элементами вероятностной логики являются логические связки-операции ($\neg, \wedge, \vee, \leftrightarrow, \rightarrow$) с индексом p , с помощью которого мы указываем вероятностную оценку истинности определенной формулы [12-13]. Пример

интерпретация вероятностного отрицания $\neg_p$ представлен в табл. 2. Общие сведения о свойствах и операциях ВЛ представлены в табл.3.

Таблица 2 – Вероятностное отрицание

Вход X	Выход $\neg_p X$	
	0	1
0	1-p	p
1	p	1-p

Таблица 3 –Свойства и операции ВЛ

1	Коммутативность	$x \vee_p y \leftrightarrow y \vee_p x$ $x \wedge_p y \leftrightarrow y \wedge_p x$
2	Двойное отрицание	$\neg_q(\neg_p x) \leftrightarrow \neg_p(\neg_q x)$ $\neg_p 0 \leftrightarrow \neg_1(\neg_p 1)$ $\neg_p 1 \leftrightarrow \neg_1(\neg_p 0)$
3	Операции с 1 и 0	$(0 \wedge_p y) \leftrightarrow \neg_p 1$ $(1 \wedge_p y) \leftrightarrow \neg_1(\neg_p y)$ $(0 \vee_p y) \leftrightarrow \neg_1(\neg_p y)$ $(1 \vee_p y) \leftrightarrow (\neg_p 0)$
4	Эквивалентность	$(y \wedge_p y) \leftrightarrow \neg_1(\neg_p y)$ $(y \vee_p y) \leftrightarrow \neg_1(\neg_p y)$
5	Вероятностная тавтология	$(y \wedge_p(\neg_1 y)) \leftrightarrow \neg_p 1$ $(y \vee_p(\neg_1 y)) \leftrightarrow \neg_p 0$
6	Вероятностная формула де Моргана	$\neg_q(x \vee_p y) \leftrightarrow \neg_1 y \wedge \neg_1 x$ $\neg_q(x \wedge_p y) \leftrightarrow \neg_1 y \vee \neg_1 x$

Подчеркнем, что в этой таблице мы используем соотношение $g = pq + (1-p)(1-q)$, которое играет важную роль в ВЛ. Нужно отметить, что ВЛ не является дистрибутивной и ассоциативной

Для обработки рефлексивных предложений, используем вероятностный преобразователь на рис.6. Здесь P_r -вероятность правильной оценки воздействия X и соответствующего ответа Y .

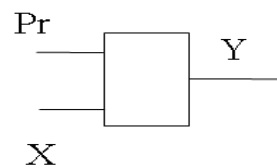


Рисунок 6– Вероятностный преобразователь

Мы интерпретируем X, Y как вероятности. Изображенный узел является простейшим вероятностным преобразователем, который при P_r близком к 1 выполняет операцию $T(X, P_r)$ при $X=0$ или 1, где функция T определена в таб.4.

Это означает, что если P_r булева переменная, а X –действительное число (вероятность), мы продолжаем функции $T(X, 0)$, $T(X, 1)$ на вещественный отрезок $(0, 1)$ функциями $1-X$ и X соответственно.

Таблица 4 – Таблица истинности T(P1,P2)

P1	P2	T(P1,P2)
0	0	1
0	1	0
1	0	0
1	1	1

Подчеркнем, что $T(p,q) = p \vee q$ и это выражение есть базовым для рефлексивной вероятностной логики.

Заметим, что предлагаемый преобразователь явно ассоциируется с “предложение есть речение, в котором что-нибудь утверждается или отрицается относительно чего-нибудь другого;” [15,с.73].

1	1	2	! x1 && ! x2	! x1 && x2
2	1	4	! x1 && ! x2	x1 && ! x2
3	1	6	! x1 && ! x2	(x1 && ! x2) (! x1 && x2)
4	1	8	! x1 && ! x2	x1 && x2
5	1	10	! x1 && ! x2	x2
6	1	12	! x1 && ! x2	x1
7	1	14	! x1 && ! x2	x1 x2
8	2	4	! x1 && x2	x1 && ! x2
9	2	5	! x1 && x2	! x2
10	2	8	! x1 && x2	x1 && x2
11	2	9	! x1 && x2	(x1 && x2) (! x1 && ! x2)
12	2	12	! x1 && x2	x1
13	2	13	! x1 && x2	x1 ! x2
14	3	4	! x1	x1 && ! x2
15	3	8	! x1	x1 && x2
16	3	12	! x1	x1
17	4	8	x1 && ! x2	x1 && x2
18	4	9	x1 && ! x2	(x1 && x2) (! x1 && ! x2)
19	4	10	x1 && ! x2	x2
20	4	11	x1 && ! x2	! x1 x2
21	5	8	! x2	x1 && x2
22	5	10	! x2	x2
23	6	8	(x1 && ! x2) (! x1 && x2)	x1 && x2
24	6	9	(x1 && ! x2) (! x1 && x2)	(x1 && x2) (! x1 && ! x2)
25	7	8	! x1 ! x2	x1 && x2

Рисунок 7 - Список систем от двух несовместных булевых уравнений от двух булевых переменных

Список систем от трех несовместных булевых уравнений от трех булевых переменных

Он был построен согласно программе в Wolfram Mathematica 11.2

Основной фрагмент программы представлен ниже.

Список систем от двух несовместных булевых уравнений от двух булевых переменных

Этот список был из 25 систем построен согласно программе в Wolfram Mathematica 11.2. Булевы функции представлены минимальными формами.

Список представлен на рис.7. Второе (третье) числа являются номерами булевых уравнений в нотации пакета Математика (функция BooleanFunction), которые представлены далее в строке.

```
For[k11 = 1, k11 ≤ N16 - 2, k11++,
For[k12 = k11 + 1, k12 ≤ N16 - 1, k12++, For[k13 = k12 + 1, k13 ≤ N16, k13++,
If[
Length[
RES = FindInstance[{BooleanFunction[k11, {x1, x2, x3}] == True &&
BooleanFunction[k12, {x1, x2, x3}] == True && BooleanFunction[k13, {x1, x2, x3}] == True},
{x1, x2, x3}, Booleans]]] == 0,
NW = NW + 1, 0] ]]]
```

Количество таких систем равно 922110 и естественно их нельзя представить полностью в работе. Примеры некоторых из этих систем представлены далее.

253 1 3 4 !x1&&!x2&&!x3 !x1&&!x2 !x1&&x2&&!x3
254 1 3 6 !x1&&!x2&&!x3 !x1&&!x2 (!x1&&x2&&!x3) || (!x1&&!x2&&x3)
255 1 3 8 !x1&&!x2&&!x3 !x1&&!x2 !x1&&x2&&x3
256 1 3 10 !x1&&!x2&&!x3 !x1&&!x2 !x1&&x3
257 1 3 12 !x1&&!x2&&!x3 !x1&&!x2 !x1&&x2

Простейшие несовместные системы булевых уравнений и их рефлексивные представления

Основой дальнейших обобщений является наличие несовместной пары булевых уравнений

$$x1=f, \neg x1=f,$$

1	11	1	3	(!x1&&!x2) (!x1&&x3)	!x1&&!x2	!x1
2	14	2	3	(!x1&&x2) (!x1&&x3)	!x1&&x2	!x1
3	35	1	5	(!x1&&!x2) (!x2&&x3)	!x1&&!x2	!x2
4	43	1	7	(!x1&&!x2) (!x1&&x3) (!x2&&x3)	!x1&&!x2	!x1 !x2
5	44	2	6	(x1&&!x2&&x3) (!x1&&x2)	!x1&&x2	(x1&&!x2) (!x1&&x2)
6	46	2	7	(!x1&&x2) (!x2&&x3)	!x1&&x2	!x1 !x2
7	47	3	7	!x1 (!x2&&x3)	!x1	!x1 !x2
8	50	4	5	(x1&&!x2) (!x2&&x3)	x1&&!x2	!x2
9	56	4	6	(x1&&!x2) (!x1&&x2&&x3)	x1&&!x2	(x1&&!x2) (!x1&&x2)
10	58	4	7	(x1&&!x2) (!x1&&x3)	x1&&!x2	!x1 !x2
11	59	5	7	(!x1&&x3) !x2	!x2	!x1 !x2
12	131	1	9	(x1&&x2&&x3) (!x1&&!x2)	!x1&&!x2	(x1&&x2) (!x1&&!x2)
13	139	1	11	(!x1&&!x2) (x2&&x3)	!x1&&!x2	!x1 x2
14	140	2	10	(!x1&&x2) (x2&&x3)	!x1&&x2	x2
15	142	2	11	(!x1&&x2) (!x1&&x3) (x2&&x3)	!x1&&x2	!x1 x2
16	143	3	11	!x1 (x2&&x3)	!x1	!x1 x2
17	163	1	13	(x1&&x3) (!x1&&!x2)	!x1&&!x2	x1 !x2
18	172	2	14	(x1&&x3) (!x1&&x2)	!x1&&x2	x1 x2
19	176	4	12	(x1&&!x2) (x1&&x3)	x1&&!x2	x1
20	178	4	13	(x1&&!x2) (x1&&x3) (!x2&&x3)	x1&&!x2	x1 !x2
21	179	5	13	(x1&&x3) !x2	!x2	x1 !x2
22	184	4	14	(x1&&!x2) (x2&&x3)	x1&&!x2	x1 x2
23	194	8	9	(x1&&x2) (!x1&&!x2&&x3)	x1&&x2	(x1&&x2) (!x1&&!x2)
24	200	8	10	(x1&&x2) (x2&&x3)	x1&&x2	x2
25	202	8	11	(x1&&x2) (!x1&&x3)	x1&&x2	!x1 x2
26	206	10	11	(!x1&&x3) x2	x2	!x1 x2
27	224	8	12	(x1&&x2) (x1&&x3)	x1&&x2	x1
28	226	8	13	(x1&&x2) (!x2&&x3)	x1&&x2	x1 !x2
29	232	8	14	(x1&&x2) (x1&&x3) (x2&&x3)	x1&&x2	x1 x2
30	236	10	14	(x1&&x3) x2	x2	x1 x2
31	242	12	13	x1 (!x2&&x3)	x1	x1 !x2
32	248	12	14	x1 (x2&&x3)	x1	x1 x2

где f принимает булевы значения True,False, как было упомянуто было ранее.

Определим

$$G(x1,x2)=G(x1,1)\wedge x2\vee G(x1,0)\wedge \neg x2.$$

Если взять первую нетривиальную функцию $G(x1,1)=x1$, получаем $G(x1,x2)=x1\wedge x2\vee \neg x1\wedge \neg x2$. Если $G(x1,1)=\neg x1$, получаем $G(x1,x2)=\neg x1\wedge x2\vee x1\wedge \neg x2$.

Остальные 2 функции $G(x1,1)=\text{True}$ или False мы рассматривать не будем.

Список рефлексивных функций, которые описывают систему несовместных булевых уравнений от двух переменных

Список представлен на рис.8

Рисунок 8 - Рефлексивные функции для систем несовместных булевых уравнений от двух переменных

Второе (третье) числа являются номерами булевых уравнений в нотации пакета Математика (функция BooleanFunction), которые представлены далее в строке.

Второе (третье, четвертое) числа являются номерами булевых уравнений в нотации пакета Математика (функция BooleanFunction), которые представлены далее в строке.

Графическое представление вероятностных логических операций

На рис.9 представлена вероятностная рефлексивная функция для несовместной системы $\neg x_1 \wedge x_2 = 1(\text{True})$, $x_1 \wedge \neg x_2 = 1(\text{True})$
 $F(x_1, x_2) = (1-x_1) \cdot x_2 + x_1 \cdot (1-x_2) - (1-x_1) \cdot x_2 \cdot x_1 \cdot (1-x_2)$

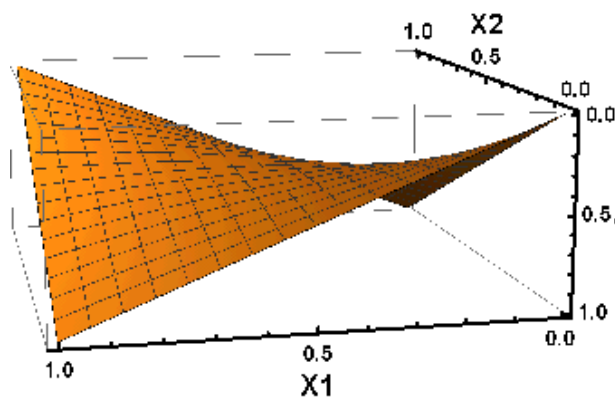


Рисунок 9 – Вероятностная рефлексивная функция

Выводы

В работе представлена одна из возможных модельных концепций логико-рефлексивного аппарата работы с символами интеллектуальной системы.

Функционирование логико-рефлексивного блока может быть описано следующими частями:

1. Представление текущей ситуации представлено символьными отношениями между элементами описания (в число которых входят как свойства, так и отношения между элементами наблюдений сознания). Ясно, что здесь необходимо прозрачное понимание и решение проблемы символа.

2. Обобщающее рефлексивное описание (если возможно) и следуемый из него рефлексивный вывод для «парадоксальных» или несовместных логических фактов (отношений).

3. Переход от булевой логики к вероятностной логике применением известных

формул и использованием таблицы для интерпретации наблюдаемых фактов.

Это направление перспективно для дальнейших исследований

Литература

1. Локк Джон. Сочинения в 3-х т.: Т. I. - М.: Мысль, 1985. — 621 с.
2. Янгутов Л.Е. Философское учение школы хуаянь.- Новосибирск: Наука, 1982.
3. Аристотель. Категории. ГСЭИ. Москва, 1939.
4. Секст Эмпирик. Сочинения в двух томах. Т.1. - М., Мысль, 1975. - 399 с.
5. Р.Шенк. Обработка концептуальной информации. - М.: Радио и связь, 1980.
6. Фреге Г. Логика и логическая семантика: Сборник трудов. - М.: Аспект Пресс, 2000. - 512 с.
7. Андриюхин А.И. Вероятностные оценки рефлексивных логических связей // Наукові праці Донецького національного технічного університету. Серія : Інформатика, кібернетика та обчислювальна техніка. - 2015. - Вип. 1. - С. 147-154. - Режим доступа: http://nbuv.gov.ua/j-pdf/Npdntu_inf_2015_1_25.pdf
8. Тейяр де Шарден П. Феномен человека. М., 2001.
9. Лепский В.Е., Зорина Г.И. Рефлексивное предприятие XXI века // Рефлексивные процессы и управление, № 2, 2005, том 5, с.21-40
10. Лефевр В.А. Конфликтующие структуры. - М.: Наука, 1973. - 188 с.
11. Lefebvre, V. A. A Psychological Theory of Bipolarity and Reflexivity. Lewiston, N.Y.: The Edwin Mellen Press. 1992.
12. Андриюхин А.И. Рефлексивные булевы функции // Информатика и кибернетика, № 4, 2016. - С.4-13.
13. Андриюхин А.И. Сознание, обратная связь и негативные рефлексивные булевы функции // Информатика и кибернетика, № 1, 2017, с.10-24.
14. Андриюхин А.И., Подтынний С.Д. КМОП-реализации рефлексивных систем вероятностной логики. // Информатика и кибернетика, № 1, 2015. С. 25-34.
15. Милль Д. С. Система логики силлогической и индуктивной: Изложение принципов доказательства в связи с методами научного исследования. — М.: ЛЕНАНД, 2011. — 832 с.

Andryukhin A.I. Reflective representations. The aim of the research is to use reflexive models of incompatible facts, described by Boolean equations from a finite number of variables. All incompatible systems of Boolean equations of two variables are presented. All incompatible systems of Boolean equations from three variables are constructed. All possible reflexive representations of inconsistent systems of Boolean equations from two variables are presented. All possible reflexive representations of incompatible systems of Boolean equations from three variables are constructed. The results of computer calculations are presented. The calculations used the Mathematica package.

Keywords: reflexion, rank of reflection, paradoxes, model, Boolean functions

Андрюхин А.И. Рефлексивные представления. Целью исследования является применение рефлексивных моделей несовместных фактов, описываемых булевыми уравнениями от конечного числа переменных. Представлены все несовместные системы булевых уравнений от двух переменных. Построены все несовместные системы булевых уравнений от трех переменных. Приведены все возможные рефлексивные представления несовместных систем булевых уравнений от двух переменных. Построены все возможные рефлексивные представления несовместных систем булевых уравнений от трех переменных. Приведены результаты компьютерных расчетов. В расчетах использовался пакет Mathematica.

Ключевые слова: рефлексия, ранг рефлексии, парадоксы, модель, булевы функции

Статья поступила в редакцию 20.10.2017

Рекомендована к публикации д-ром физ.-мат. наук А.С. Миненко

Компьютерное решение проблем. Ассоциативный поиск и рефлексивный вывод в когнитивных системах на основе SQL и NoSQL СУБД.І.

Андрюхин А.И., Новиков Е.Т.
Донецкий национальный технический университет, г. Донецк,
alexandruckin@rambler.ru

Андрюхин А.И., Новиков Е.Т. Компьютерное решение проблем. Ассоциативный поиск и рефлексивный вывод в когнитивных системах на основе SQL и NoSQL СУБД.І. В работе рассматривается решение проблем как один из основных человеческих когнитивных процессов. Представлено взаимодействие процесса решения проблем с такими когнитивными процессами, как абстракция, поиск, вывод, анализ и синтез на базе представления внутренних знаний известной моделью Вещь(объект)-атрибута-отношения (ВОС). После идентификации объекта проблемы, решение рассматривается как процесс поиска в пространстве памяти для нахождения взаимосвязи между набором целей решения и набором альтернативных путей. Исследуются комплекс когнитивных вычислений для описания и моделирования основных механизмов и процессов мозга. Выполнена попытка использования этой методики для проектирования новых технических решений с применением ассоциативного поиска и рефлексивного вывода в SQL и NoSQL СУБД.

Ключевые слова: Когнитивные вычисления, модель мозга, когнитивные процессы, решение проблем, концептуальная алгебра, ассоциативный поиск, рефлексивный вывод, NoSQL БД.

Введение

Одним из основных человеческих когнитивных процессов является решение проблем. Как когнитивный процесс более высокого уровня, решение проблем взаимодействует со многими другими когнитивными процессами, такими как абстракция, поиск, обучение, принятие решений, вывод, анализ и синтез и др. В работе основой представления внутренних знаний является известная инфологическая модель вещь(объект)-отношения-свойство (ВОС). Решение проблем - это когнитивный процесс мозг, который ищет решение для данной проблемы или находит путь к достижению заданной цели. Когда объект проблемы идентифицирован, решение проблемы можем рассматривать как процесс поиска в пространстве памяти для нахождения взаимосвязи между набором целей решения и набором альтернативных путей.

В настоящей статье представлены как когнитивная модель, так и математическая модель процесса решения проблем.

В работе также представлены когнитивные структуры мозга и механизмы представления внутренних знаний для когнитивного процесса решения проблем.

Сам когнитивный процесс формально описывается с использованием формализмов, которые представлены алгеброй понятий, алгеброй процессов реального времени и др.

Эта работа относится к области разработки методологий компьютерных когнитивных вычислений и новых когнитивных компьютеров, способных воспринимать, мыслить, и самообучаться.

Более конкретно, в данной работе рассматриваются задачи реализации механизмов ассоциативного поиска и рефлексивного логического вывода, моделирующих элементы когнитивных процессов сознания человека с использованием существующих SQL и NoSQL СУБД и SQL-запросов, т.е. является подсистемой системы компьютерного решения проблем.

Эта задача решается уже достаточно длительное время (за время развития компьютерных технологий и ранее). Упомянем только такие направления, как известные поисковые системы в Internet, онтологическая база знаний Сус, система-платформа Protégé для построения баз знаний, система Rules Wizard для поиска ассоциативных связей, аппаратно-программный комплекс STARAN и др.

Структура проблемы

В настоящее время известны следующие взгляды на общую структуру решения проблем.

Так в [1,2] считают, что проблема состоит из трех компонентов, известные как данная входная информация (начальные состояния и т.п.), цели и операции. Входная информация представляется в виде части проблемы. Цели определяются как желаемое окончательное состояние решения проблемы. Операции являются потенциальными действиями, которые могут быть выполнены для достижения цели решения.

Для любой заданной задачи существует связанное проблемное пространство [3,4], которое является всеми возможными целями и путями и потенциально связано с проблемой GPS[5].

Полагается, что проблемное пространство или пространство решений H является декартовым произведением непустого множества объектов задачи X , непустое множество путей Π и непустое множество целей C .

Решение определенной проблемы может не существовать в текущем пространстве решений решателя. Это может быть вызвано многими факторами, такими как, проблема поставлена нечетко, ожидаемые цели являются неоднозначными, и/или нет метода (пути), который связывает представления объекта проблемы с целями (задачами).

Аналогично представлено проблемное пространство в [6,7], в которых определены два элемента решения проблем:

(а) описание всех возможных состояний задачи и проблемы решателя (представления);

(б) список способов перемещение между этими состояниями (поиск). Первый элемент необходим решателю для понимания проблемы посредством абстракции и идентификации. Второй элемент позволяет использовать решатель проблемы для поиска возможного решения в памяти.

Типизация решений проблем

В настоящее время принята следующая классификация типов решения задач:

1.Прямое решение - поиск прямого пути решения на основе известных решений

2.Эвристика - принятие эмпирического правила или наиболее возможного решения.

3.Аналогия - сведение новой проблемы до существующей или аналогичной той, для которой решение уже известно.

4.Последовательное приближение - приближение к цели задачи шаг за шагом.

5.Алгоритмический вывод - применение известного и хорошо определенного решения для задачи.

6. Искрывающий поиск - с помощью систематического поиска всех возможных решений.

7 Разделяй-и-власть - решение всей проблеме декомпозицией, т.е. разлагая ее на набор подзадач.

8. Анализ и синтез - сведение данной проблемы до известных компонентов, а затем нахождение частных решений.

Принятие вышеупомянутых подходов к решению проблем естественно не может гарантировать достижение поставленной задачи, особенно когда решение не находится в пределах пространства решений решателя проблемы.

Существует ряд факторов, которые могут препятствуют процессу решения проблем, такие как:

(а) неудачный выбор направления работы, при котором неправильный метод принят для решения новой проблемы, в то время как более простые решения могли быть использованы;

(б) необходим метапознавательный подход, в котором для решения проблемы может потребоваться поддержка других метакогнитивных процессов для решения проблемы;

(с) отсутствие знаний, вследствие чего проблема или цель не могут быть хорошо представлены или промоделированы, и ни один метод не может быть применен к проблеме[8,9].

Модели когнитивных общих понятий

1.Аристотель

В 4 главе известных «Категорий» Аристотеля представлены 10 самых общих понятий у человека

[Десять категорий]

Из сказанного без какой-либо связи каждое означает или сущность, или «сколько», или «какое», или «по отношению к чему-то», или «где», или «когда», или «находиться в каком-то положении», или «обладать», или «действовать», или «претерпевать».

Необходимо подчеркнуть дальнейшее разъяснение Аристотеля:

Каждое из перечисленного само по себе не содержит никакого утверждения; утверждение или отрицание получается сочетанием их: ведь всякое утверждение или отрицание, надо полагать, или истинно, или ложно, а из сказанного без какой-либо связи ничто не истинно и не ложно, например «человек», «белое», «бежит», «побеждает»[10].

В настоящее время многие специалисты считают, что « критерии, которые использует Аристотель для того, чтобы определить принадлежность термина к той или иной категории, в высшей степени неясны, если их вообще можно назвать критериями. Но для его целей это не имело особого значения. Все что ему требовалось — это уметь отличать (а) качества от отношений, (b) то и другое — от субстанций и (с) все это от родов и видов. Пусть грубо и неточно, но он умел это делать.» [11,12]

Однако следует отметить некоторые другие недостатки в его схеме. Ведь самые простые с грамматической точки зрения предложения зачастую выражают не простые высказывания и здесь необходимо четко эксплицировать слова типа «все, некоторые, любой, этот, какой-либо, не, если..., то, или, и, чем и т. п.». Поэтому нам нужен метод, с помощью которого можно было бы представить и, что является особой задачей, установить однородность и разнородность категорий» [11,12].

2. Фреге

В настоящее время многие специалисты обращаются к гносеологическим воззрениям Г.Фреге, который занимался логико-математическим описанием, а точнее исчислением понятий (дословное название его одного из основных произведений)[13]. Обычно в логике рассматривались и рассматриваются суждения, как структурные знаковые системы.

Простейшими элементами являются понятия и логические связи. Исторически исследование таких систем развивалось как обобщение форм Аристотеля (логика предикатов, модальные логики и т.п.), алгебраические представления и т.п.

Фреге использовал только две сущности, *предмет* и *функция* и рассматривал в основном их отношение к *истине* и *знаку*. Его базовым положением является утверждение «При любом истолковании суждения нет места различия различию субъекта и предиката». На рис.1 и 2 представлены логические тезисы Фреге при его логическом исследовании естественного языка (немецкого).



Рисунок 1– Семантический треугольник

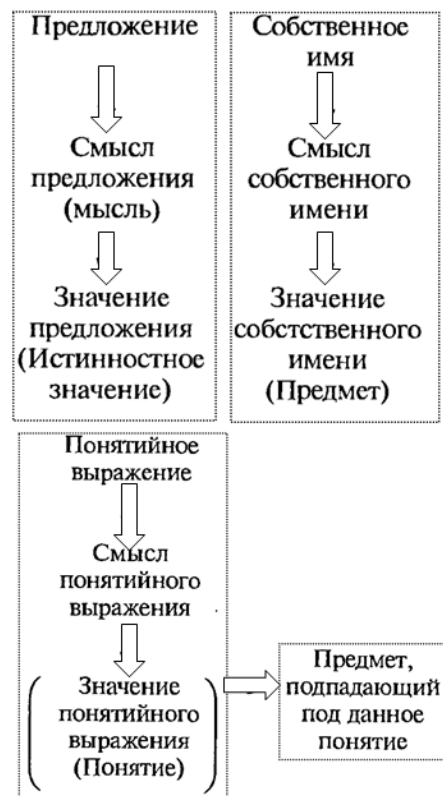


Рисунок 2 –Базовые положения Фреге

Работы Фреге заставляют по-иному посмотреть на строение наблюдаемого мира и его отражение в понятиях. К примеру, рассмотрим известный пример из его статьи «Логическое исследование мысли» в нашей интерпретации:

1. Люди общаются и у людей есть нечто общее, что называется смыслом.

2. Человек имеет дело непосредственно со своим представлением о мире, а не с внешним миром. (Буддийская притча-Что движется: флаг или ветер? Ответ: движется ваше сознание.)

3. Внутренний мир, мир представлений у людей не может быть общими и у каждого человека свой.

4. Согласно Фреге, суть общения, смысл или мысль "не относится ни к представлениям из моего внутреннего мира, ни к внешнему миру..."

Следовательно, мир троичен или по крайней мере имеет три различные части (для нас!).

3. Система GPS

Рекурсивный метод GPS является формальным методом, который выполняет декомпозицию проблемы на подпроблемы[5]. Он использует три метода для перевода проблемы из состояния А в состояние В, которые представлены на рис.3.

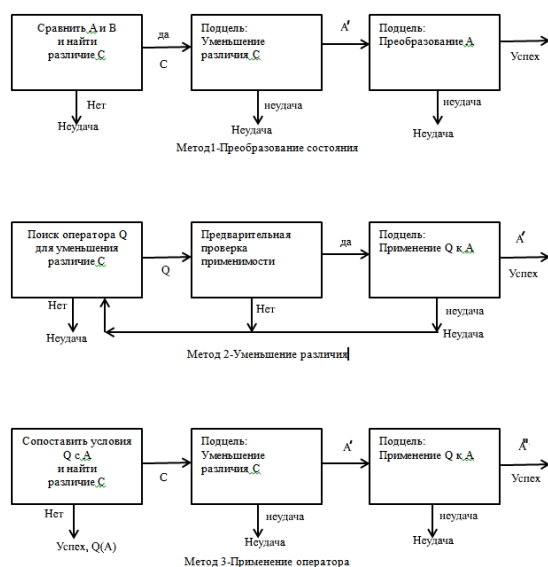


Рисунок 3 – Методы GPS

4. Система MARGIE и ее концептуальный синтаксис

Создатели этой системы (группа психологов, филологов, программистов) считали справедливым идею Хомского:

«существует некоторый язык, язык посредник, который является врожденным для представителя человеческого рода»[14]

Система MARGIE оказывает большое влияние на исследователей в области проблемы понимания текста.

Обработка естественного языка(ЕЯ) в ней происходит в 3-х направлениях:

А) отражение предложений естественного языка в их смысловое представление;

Б) хранение смыслового представления в памяти компьютера и осуществление умозаключений относительно полученного смысла;

В) перевод смыслового представления на естественный язык.

Основным элементом в системе является концептуализация – базовая единица понятийного уровня, из которой конструируется мысль. Были введены следующие концептуализации:

ДЕЯТЕЛЬ – понятие исполнителя АКТА

АКТ – действие, производимое по отношению к ОБЪЕКТУ

ОБЪЕКТ – вещь, над которой производится действие

РЕЦЕПИЕНТ – получатель объекта в результате АКТА

НАПРАВЛЕНИЕ – местоположение, к которому направлен АКТ

СОСТОЯНИЕ – состояние ОБЪЕКТА

Это базовый набор, из которого строится смысловое представление предложения.

Предложениям разговорного англ. языка соответствуют двумерные диаграммы, отражающие их смысл.

Были построены множества типов этих двумерных диаграмм (шаблонов, фреймов), к которым сводилось любое предложение естественного языка.

Система была написана на диалекте Лиспа.

В концептуализацию помещаются концепты (понятия) точно так же, как в предложение естественного языка помещаются слова.

Состоят из деятеля и действия. Иногда – объект и состояние.

ДЕЯТЕЛЯМИ могут физ. объекты, способные выполнять **АКТЫ**: компьютер, человек, ветер.

АКТЫ или действия разбиты на 2 класса:

- физические акты;
- психические акты.

Первые 5 физических актов описывают физические действия, производимые людьми:

- 1)ПРИКЛАДЫВАТЬ СИЛУ
- 2)ДВИГАТЬ ЧАСТЬЮ ТЕЛА
- 3)ПРИНЯТЬ ЧТО-ЛИБО ВНУТРИ ОДУШЕВЛЕННОГО ОБЪЕКТА
- 4)ВЗЯТЬ ЧТО-ЛИБО ИЗНУТРИ ОДУШЕВЛЕННОГО ОБЪЕКТА
- 5)ФИЗИЧЕСКИ ЗАХВАТИТЬ

Следующие 2 акта, необходимы для акцентирования результатов действия:

- 6)ИЗМЕНИТЬ МЕСТОПОЛОЖЕНИЕ ЧЕГО-ЛИБО
- 7)ИЗМЕНИТЬ ОТНОШЕНИЕ ДЛЯ ОБЪЕКТА

Инструментальные акты

- 8)ПРОИЗВЕСТИ ЗВУК
- 9)НАПРАВИТЬ ОРГАН ЧУВСТВ К ОПРЕДЕЛЕННОМУ СТИМУЛУ.

Для описания психических актов необходимо описать строение памяти человека, принятое создателями системы.

В речевых оборотах ЕЯ наблюдается разделение памяти на следующие части:

- 1) концептуальный процессор

В нем происходят все сознательные мыслительные процессы.

Концептуальный процессор пропускает только одну концептуализацию в единицу времени.

- 2)долговременная память

Вся память, присущая данному субъекту, вся информация, принимаемая человеком.

- 3) промежуточная память, где сохраняются элементы сознания, не используемые в данный

момент. Отражается в конструкциях: «быть на уме», «предположим».

Для представления концептуализации широко используется понятие состояние объекта, которое описывается с помощью определенных шкал и пример показан на рис.4.

Параметр состояния	шкала	Словесная интерпретация
Здоровье	[-10, 10]	-10 мертвый 0 нормально 10 абсолютно здоров
Психическое состояние	[-10, 10]	-5 подавлен 0 нормально 2 доволен 8 счастлив 10 экстаз
Отрицание	[-10, 0]	-8 омерзительно -6 отвратительно -4 противный -2 надоевший

Рисунок 4 - Шкалы состояния объектов

Конечной целью представления предложений в концептуальных единицах является использование этих концептуализаций для совершения умозаключений.

Для некоторых актов существуют наборы умозаключений, которые являются верными с некоторой степенью достоверности.

Умозаключение – новая концептуализация, которая выводится с некоторой вероятностью (достоверностью) из других концептуализаций.

Примеры некоторых из 16 типов умозаключений, рассматриваемых в Margie:

1)каузативное умозаключение- «Петя ударил Васю палкой», умозаключение «Петя был сердит на Васю»

2)умозаключение об условиях- «Свеженцев отправился на Канары» умозаключение: «откуда у него деньги»

Система из предложения «Андрей съел печенье» может на основании своей базы построить и ассоциативные умозаключения (последнее):

1)печенье начало находится во рту у Андрея;

2)печенье перестало находиться во рту у Андрея;

3)Андрей наелся;

4)Андрею понравилось печенье;

5) Андрею нужно сходить к врачу, так как печенье сладкое.

Сравнение умозаключений в MARGIE и традиционных дедуктивных системах

Концептуальное умозаключение – это новый кусок информации, порождаемый из других концептуализаций с некоторой степенью достоверности.

Цель концептуального умозаключения: достройка и восстановление ситуаций, которые подразумеваются высказыванием.

Концептуальное умозаключение – это попытка достроить пропущенную информацию и определить следующие цели:

1) возможности, не выраженные явно в высказываниях;

2) обусловленности, намерения деятельности в момент высказывания.

Денотационная математика Ванга для компьютерного решения проблем

Одно из современных известных направлений компьютерного решения проблем представлено в [15]. Основой для рассматриваемых там процессов решения проблем является парадигмальная схема структурных слоев человеческого мозга представленного на рис.5.



Рисунок 5 - Модель структурных слоев мозга LRMB [16]

Базируясь на представлении LRMB(The layered reference model of the brain) вводятся следующие основные концепции, такие как RTPA(Real-Time Process Algebra) и алгебра понятий, которые представляют компоненты денотационной математики [15,16]

Согласно последним работам, Ванг понимает под денотационной математикой - категории выразительных математических структур, которые имеют дело с высоким уровнем

сложных математических сущностей (обычно нечислового характера, т.е. нечисловые множества), таких как абстрактные объекты, сложные отношения, информация о поведении, концепции, знания, процессы и системы.

РТПА используется для формального моделирование поведения человека и представляет триплет $((T, B, R))$.

Здесь T - набор из 17 примитивных типов базовых объектов данных представленных на рис.6.

Primitive types	(Базовые типы)	T
N	Natural number (Натуральное число)	
Z	Integer (Целое число)	
R	Real (Действительное число)	
S	String (Строка)	
BL	Boolean (Булева переменная и	
T, F	Boolean constants константы)	
B	Byte Байт	
H	Hexadecimal 16-ричное число	
P	Pointer Указатель	
TI = hh:mm:ss:ms	Time Время	
D = yy:MM:dd	Date Дата	
DT = yyyy-MM-dd: hh:mm:ss:ms	Date/Time (Дата/Время)	
RT	Run-time determinable type	
	(Определяемый при выполнении тип)	
ST	System architectural type	
	(Тип системной архитектуры)	
@eS	Event Событие	
@TM	Timing Длительность	
@int	Interrupt Прерывание	
@sBL	Status Состояние	
System Algebra (Алгебра систем)		
$\supset / \sqsubset$	Super/sub relation	
	(Отношение супер/подсистема)	
	(Связанный независимый)	
$\leftrightarrow / \nleftrightarrow$	Related/independent	
$=$	Equivalent Эквивалентность	
Π	Overlapped (Перекрытая)	
$\sqcup$	Conjunction (Конъюнкция)	
$\ominus$	Difference (разница)	
$\Rightarrow$	Inheritance (Наследование)	
$\supset +$	Extension (Расширение)	
$\Rightarrow +$	Tailoring (Сшивание)	
$\supset \rightarrow$	Substitute (замена)	
$\oplus$	Composition (Композиция)	
$\oslash$	Decomposition (разложение)	
$\Leftarrow$	Aggregation/generalization	
$\vdash$	Specification (Спецификация)	
$\dashv$	Instantiation Создание экземпляра	

Рисунок 6 - Базовые типы данных и алгебра систем

B - набор из 17 метапроцессов для моделирования фундаментального поведения систем представленных на рис.7.

R - набор из 17 операций реляционных процессов для построения описания поведения

(конструирования) сложных систем. Этот набор представлен на рис. 8.

Meta Processes	(Метапроцессы)
$:=$	Assignment (Назначение)
$\blacklozenge$	Evaluation (Оценка, определение)
$\Rightarrow$	Addressing (Адресация)
$\Leftarrow$	Memory allocation (Выделение памяти)
$\neq$	Memory release (Освобождение памяти)
$\triangleright$	Read Чтение
$\triangleleft$	Write (Запись)
$\triangleright$	Input (Ввод)
$\triangleleft$	Output (Вывод)
$@$	Timing (Синхронизация)
$\pm$	Duration (Длительность)
$\uparrow$	Increase (Возрастание)
$\downarrow$	Decrease (Убывание)
$!$	Exception detection (Обнаружение исключений)
$\otimes$	Skip (Пропуск)
$\boxtimes$	Stop (Останов)
$\S$	System (Система)

Рисунок 7 - Метапроцессы

Process Relations	
$\rightarrow$	Sequence (Последовательность)
$\sim$	Jump (Переход)
$ $	Branch (Ветка, ответвление)
$ \dots \dots$	Switch (Переключение)
R^*	While-loop (Цикл пока)
R^+	Repeat-loop (Цикл повторять)
R^i	For-loop (Цикл для)
$\circ$	Recursion (Рекурсия)
$\mapsto$	Function call (Вызов функции)
$\parallel$	Parallel (Параллельно)
$\S$	Concurrency (Согласовано)
$\parallel$	Interleave (Чередование)
$\gg$	Pipeline (Путь-источник движения информации)
$\curvearrowright$	Interrupt (Прерывание)
$\hookrightarrow$	Time-driven dispatch
$\hookrightarrow$	Event-driven dispatch
$\hookrightarrow$	Interrupt-driven dispatch

Рисунок 8 - Отношения процессов

РТПА обеспечивает согласованную систему обозначений и формальную инженерную методологию моделирования программного обеспечения и интеллектуальных систем.

РТПА может использоваться для описания логических и физических моделей систем. Укажем, что логические представления архитектуры программной системы и ее операционная платформа могут быть представлены с использованием одного и того же набора обозначений.

Когда модель системы формально промоделирована, статическое и динамическое поведение, как системная архитектурная модель, могут быть повторно изменены на трех уровнях, таких как система, класс и объект.

Решение проблем и алгебра понятий

1. Формальное представление внутренних знаний

Концепция - это когнитивная единица для идентификации и/или моделирования конкретного объекта в реальном мире и абстрактного объекта во воспринимаемом мире. Прежде чем будет определена абстрактная концепция, введем следующие определения.

Пусть O - конечное непустое множество объектов и A - конечное непустое множество атрибутов. Далее полагаем, что семантическое окружение среды или контекст H всех понятий является триплетом, то есть:

$H = \{O, A, R\}$, $R: O \rightarrow O | O \rightarrow A | A \rightarrow O | A \rightarrow A$, где R - набор отношений между O и A , а также их отражающие отношения, а $|$ обозначает альтернативные отношения.

Понятия в денотационной математике являются абстрактными структурами, которые несут определенный смысл почти во всех познавательных процессах, как решение проблем, обучение и рассуждения. Абстрактное понятие c в H является 5-кортежем,

$$c = \{O; A; R^c; R^i; R^o\},$$

где O - является непустым множеством объектов понятий, $O = \{o_1; o_2; \dots, o_n\}$;

A - непустой набор атрибутов, $A = \{a_1; a_2; \dots, a_m\}$;

$R^c: O \rightarrow A$ - совокупность внутренних отношений;

$R^i: A' \times A, A' \in C' \wedge A \in c$ - набор входных отношений, где C' - множество внешних понятий, $C' \in H$, и для упрощения $R^i = A' \times A$ можно просто обозначить, как $R^i = C' \times c$;

$R^o: c \times C'$ совокупность выходных отношений.

Концептуальная алгебра - абстрактная математическая структура для формального рассмотрения понятий и их алгебраических отношений, операций и ассоциативных правил для составления сложных концепций.

Алгебра понятия CA на заданной семантической среде H является триплетом, то есть:

$$CA = \{C; OP; H\} = \{O; A; R^c; R^i; R^o\}, \{ \bullet_r, \bullet_c \}, H\},$$

где $OP = \{ \bullet_r, \bullet_c \}$ - множество реляционных и композиционных операции над абстрактными понятиями.

Реляционные операции $\bullet_r$ в алгебра концепций охватывает восемь сравнительных операторов для манипулирования алгебраические отношения между понятиями и представлены на рис.9.

Композиционные операции $\bullet_c$ в концептуальной алгебра включает в себя девять ассоциативных операторов для манипулирования алгебраическими композициями для понятий и представлены последними строками на рис.9.

(Алгебра понятий)	Concept Algebra
$> / <$	Super/sub relation
$\leftrightarrow / \nleftrightarrow$	Related/independent
$=$	Equivalent (Эквивалент)
$\equiv$	Consistent
$+$	Conjunction (Конъюнкция)
$*$	Elicitation
$\sim$	Comparison (Сравнение)
$\triangleq$	Definition
$\Rightarrow$	Inheritance
$\Rightarrow^+$	Extension (Расширение)
$\Rightarrow^-$	Tailoring
$\Rightarrow^+$	Substitute (Подстановка)
$\in$	Composition (Композиция)
$\ni$	Decomposition (Дekomпозиция)
$\Leftarrow$	Aggregation/generalization
$\vdash$	Specification (Спецификация)
$\vdash^+$	Instantiation (Создание экземпляра, конкретизация)

Рисунок 9 - Алгебра понятий

Концептуальная алгебра обеспечивает мощные математические денотационные средства для алгебраических манипуляций с абстрактными понятиями.

Концептуальную алгебру можно использовать для моделирования, определения и манипулирования типичными проблемами типа «быть».

Перечислим основные области ее применения, такие как базы знаний, системная инженерия и системная архитектура, дизайн систем с детализацией, когнитивная информатика, вычислительная техника, разработка ПО, системы ИИ.

Формальная модель когнитивного процесса решения проблемы

На основе РТРА и концептуальной алгебры когнитивный процесс решения проблемы, как это показано на схеме на рис.10, может быть формально описан согласно рис.11.

Согласно модели ВОС, постулируется, что представления внутренних знаний в мозге, получаемые результаты при решении задач в сознании решателя проблем являются новыми моделями в представлении ВОС, которые будут использоваться для обновления глобальной модели знаний ВОС. Эти процессы реализуются в долгосрочной памяти решателя проблем согласно композиционным операциям $\bullet_c$ и частично реляционными операциями $\bullet_r$.

Центральное положение в формальной модели процесса решения проблемы заключается в том, что знания о проблеме и ее решении (s) представлены внутри мозга человека на основе ВОС согласно рис.12, где представлен внешний мир реальными сущностями (PO), а внутренний мир виртуальными (фантомными) объектами (FC) и объектами (O).

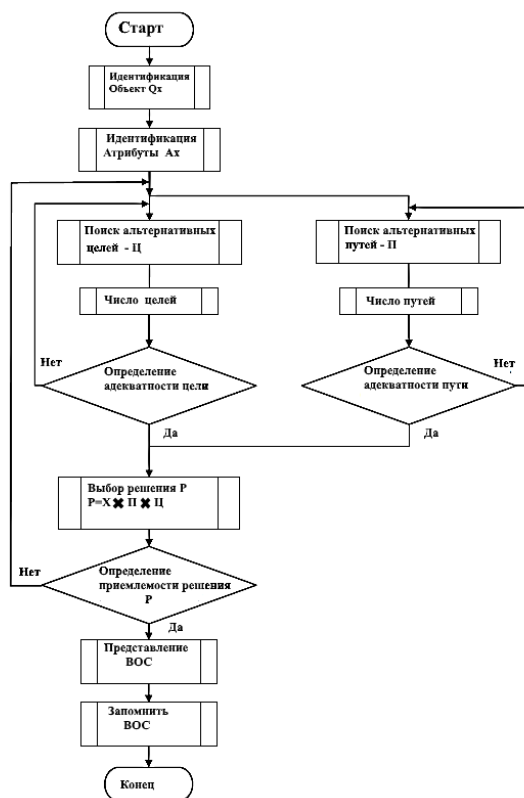


Рисунок 10 - Блок-схема решения проблемы с моделью BOC

```

PSP_Process(I:: OS, OARST; O:: OARST)ST
{
  // I. Идентификация и представление проблемы
  ↳ Идентификация (OS) // Представление объекта проблемы
  ↳ Идентификация (AST) // Характеристики проблемы
  ↳ Идентификация (R(OS, AST)ST) // Представление отношений между OS и AST
  ↳ XST := (OS, AST, RST)ST // Представление проблемы

  // II. Поиск целей и путей решения
  → {
    R (↳ Поиск (GST))
    ↳ Количество (GST)
    ↳ Оценка (GST)
  }

  // III. Генерация решений
  → SST := XS × TST × GST

  // IV. Выбор решений
  ↳ Оценка (SST)
  → ( ♦p(SST) ≥ k // k - порог удовлетворения
    → RST := S(XS, TST, GST)ST // сформировать новое отношение к SST
    → ∅ // Иначе
  )
  ↳ ♦ ~
  → ( ♦ GiveUpBL = F
    ↳ PSP_Process(I:: OS; O:: OAR(SST)ST)ST
    ↳ ♦ ~
    → ∅
  )

  // V. Представление результата решения проблемы
  → sOARST := (OS, AST, RST)ST // Формировать новую OARST модель для SST
  → OARST := OARST ⊕ sOARST // Обновить OARST
  ↳ Memorize (OARST)ST // Запомнить обновления OARST
}

```

Рисунок 11 - Псевдокод решателя проблем с

моделью BOC

Внутренний мир может быть разделен на два слоя: слой образов и абстрактный слой. Виртуальные объекты представляют собой прямые образы внешних реальных объектов, расположенными на уровне образов. Объекты являются абстрактными артефактами, расположенными на абстрактном слое. Абстрактный слой является основным определяющим свойством человеческого мозга.

Необходимо подчеркнуть, что в полной мере обыкновенный человек получает возможности применения абстрактного слоя или интеллекта в процессе его формирования. По Ж. Пиаже, человек по мере своего когнитивного развития закономерно проходит ряд стадий (Рис. 12):

1. Сенсомоторного интеллекта (от рождения до 2-х лет).
2. Дооперационального мышления (от 2 до 7 лет).
3. Стадию конкретных операций (от 7-8 до 11-12 лет).
4. Стадию формальных операций (от 11-12 до 14-15 лет).

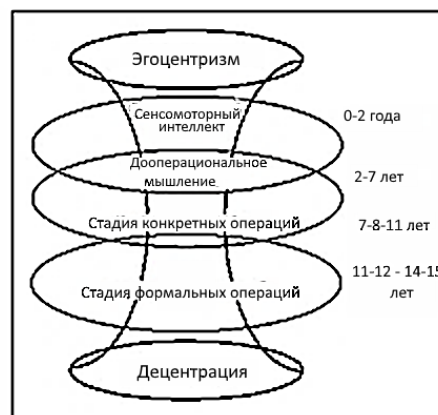


Рисунок 12 - Стадии развития интеллекта по Ж. Пиаже

Следует отметить, что виды животных не имеют такого абстрактного слоя в их мозгах. Поэтому у них нет косвенного или абстрактного мышления. Иными словами, абстрактное мышление - это уникальная сила человеческого мозга, рассматриваемая как качественное преимущество человека. Другим, но уже количественным преимуществом человеческого мозга является огромный объем долговременной памяти в коре мозга человека.

На основании этих двух основных преимуществ, человечество имеет наблюдаемую мощь множества человеческих существ.

Существуют метаобъекты (О) и производные объекты (О') в абстрактном слое в модели OAR. Первые являются объектами, непосредственно соответствующими виртуальным объектам и далее во внешний мир.

Последние являются абстрактными объектами, которые генерируются внутри и не имеют прямого соединения с виртуальными объектами или образами реальных объектов, таких как абстрактные понятия, идеи и состояния сенсорных (чувствственных) подсистем. Объекты на абстрактном слое мозга могут быть расширены в сеть объектов, атрибутов и отношений в соответствии с моделью ВОС, как показано на рисунке 13. Структурная модель когнитивной системы человека представлена на рис.14.

Когнитивное рефлексивное сознание и ТРИЗ

Процесс человеческого мышления можно представить как смену одного образа за другим.

Смена обуславливается органами чувств, а также самим процессом мышления. Мышление всегда сопоставляет образ с наиболее близкими образами из памяти. Каждый образ состоит из множества частей.

Важным примером когнитивного сознания может служить процесс решения изобретательских задач. Он представляет собой высокоуровневую абстракцию направляемого процесса мышления и многие другие абстракции процесса мышления могут быть сведены к ней.

Процесс решения изобретательской задачи – поиск решения в пространстве памяти для нахождения отношений между множествами целей решения и множеством путей их достижения, а также выбор оптимальных сочетаний.

Основа процесса решения изобретательских задач – поиск и установление ассоциативных связей между сущностями и логический вывод нового знания из уже имеющегося.

Результатом процесса решения изобретательских задач является технический объект (ТО). ТО – это созданное человеком или автоматом реально существующее (существовавшее) устройство, предназначенное для удовлетворения определенной потребности.

Любые устройства, выполняющие определенную функцию (операцию) по

преобразованию объектов живой и неживой природы, энергии или информационных сигналов можно отнести к ТО.

Каждый из них может быть представлен описаниями, имеющими иерархическую соподчиненность. Описания имеют следующие свойства:

а) каждое последующее описание является более детальным и более адекватно характеризует ТО по сравнению с предыдущим;

б) каждое последующее описание включает в себя предыдущее.

Описания имеют такие компоненты-свойства:

а) потребность или функция ТО;

б) техническая функция (ТФ);

в) функциональная структура (ФС);

г) физический принцип действия (ФПД);

д) техническое решение (ТР);

е) проект.

Вышесказанное можно представить на рис.15.

Техническая функция – это совокупность потребности, которую удовлетворяет ТО и физической операции (физическое превращение, преобразование). Поэтому $ТФ = (P, Q)$, где P – потребность, Q – физическая операция.

Описание физической операции (ФО) структурно состоит из трех компонент: А и С – входной и выходной потоки вещества, энергии или сигналов, Е – наименование операции Коллера по превращению А в С. Следовательно, $ФО = (A, E, C)$.

Подавляющее большинство ТО состоит из нескольких элементов. Каждый элемент, как самостоятельный ТО выполняет определенную функцию и реализует определенную физическую операцию. То есть между элементами имеют место два вида связей и соответственно два вида их структурной организации.

Первый вид структурной организации выражает, какие функциональные связи между собой имеют элементы.

Он называется конструктивной функциональной структурой. Конструктивная функциональная структура (ФС) – ориентированный граф, вершины которого – наименования элементов, а ребра – функции элементов.

Таблица 1 - Метакогнитивные модели мозга

Когнитивные модели	Описание	Математические абстракции и представления
Объект(Вещь, предмет)	Абстрактное представление внешней сущности или внутреннего понятия	Множество, кортеж, алгебра понятий
Свойства или атрибуты	Свойства и характеристики объекта	Множество, кортеж, алгебра понятий
Отношения	Связи и отношения между объект-объект, объект-атрибут или атрибут-атрибут	Алгебра понятий, реляционная алгебра, логика

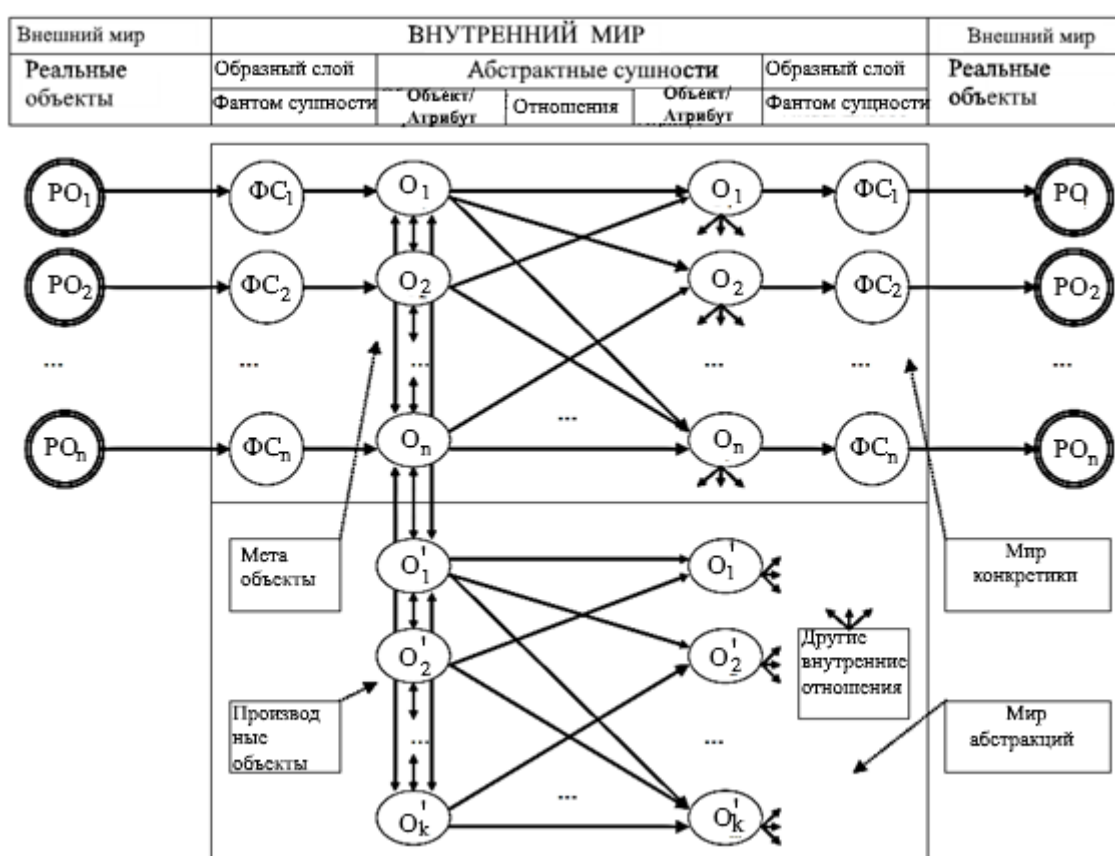


Рисунок 13 - Структурная модель мира в терминах ВОС

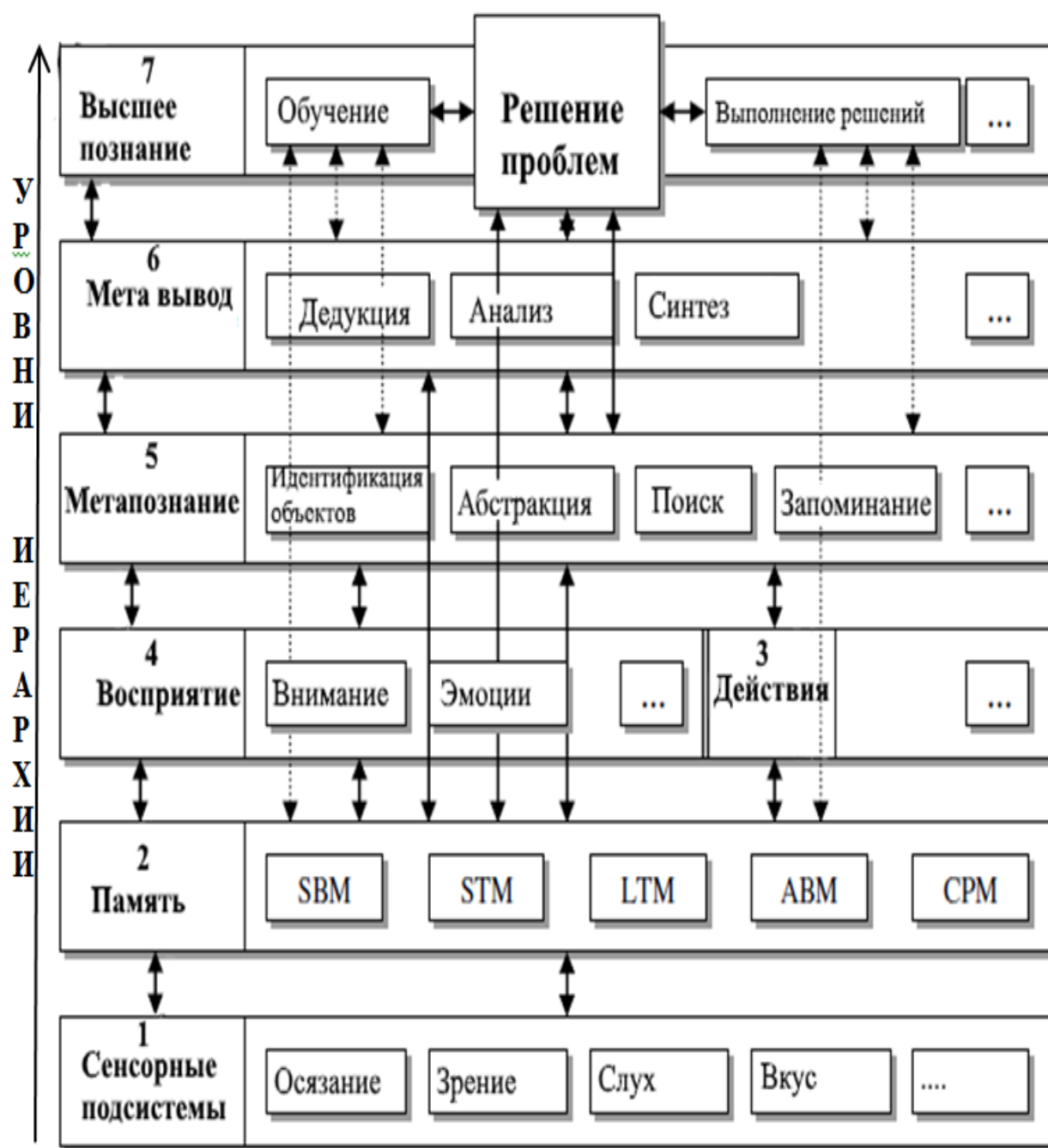


Рисунок 14 - Структурная модель когнитивной системы человека

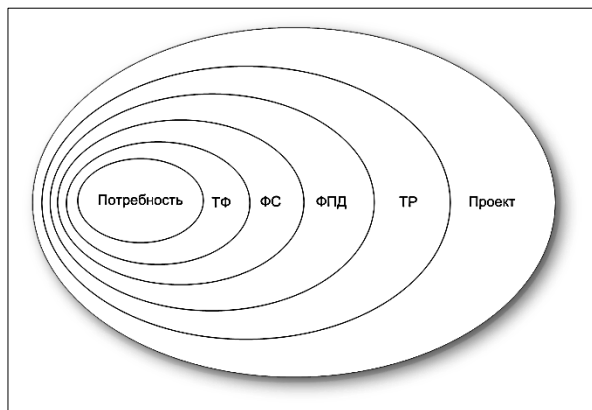


Рисунок 15 - Структурное описание объекта поиска[17]

Кроме функциональных связей, между элементами ТО имеются еще потоковые связи, то есть элементы, реализуя определенные физические операции, образуют поток преобразуемых или превращаемых веществ, энергии, сигналов или других факторов. Такие связи образуют потоковую функциональную структуру – ориентированный граф, дуги которого входные и выходные потоки вещества.

Под физико-техническими эффектами (ФТЭ) понимаются различные приложения физических законов, закономерностей и следствий из них, физические эффекты и явления, которые могут быть использованы в технических устройствах. Обобщенное качественное описание ФТЭ состоит из трех компонент: А – входной поток вещества, энергии или сигналов; С – выходной поток, В – физический объект обеспечивающий преобразование А в С. Схематично это можно представить (А->В->С) (см.рис.16).

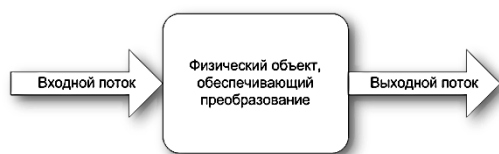


Рисунок 16 - Упрощено-идеализированное описание физико-технического эффекта

Физический принцип действия (ФПД) – ориентированный граф, вершины которого – наименования физических объектов В, ребра – входные (А) и выходные (С) потоки вещества. Описание ФПД, как правило, содержит изображение принципиальной схемы ТО, в которой в упрощено-идеализированной форме показаны основные конструктивные элементы, обеспечивающие реализацию ФПД, и указаны

направления потоков и основные физические величины, характеризующие используемые физико-технические эффекты.

Вышесказанное в основном опирается на [17,18] и конкретная задача ассоциативного поиска представляет собой программную реализацию алгоритма поиска цепочек ФПД, который представлена [17].

Конкретный пример цепочки ФПД представлен на рис.17.

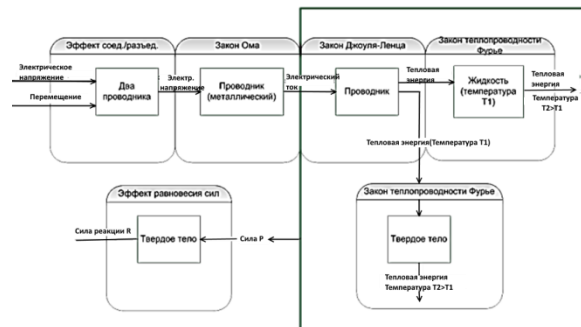


Рисунок 17 - Конкретное структурное описание ФПД

Базовые алгоритмы и модули БД

Первым базовым алгоритмом является алгоритм построения образа текстовой информации. Он позволяет поисковой системе принимать текстовое описание объекта поиска и основываясь на присутствующих в текстовом запросе словах, производить в базе знаний ассоциативный поиск релевантных компонентов.

Согласно базового алгоритма, технический объект, текстовое описание которого вводит пользователь – это объект поиска, набор физико-технических эффектов, информация о которых содержится в базе знаний – это компоненты. Результат поиска – это список потенциально возможных вариантов физического принципа действия объекта поиска. Элемент результата – это один из ФПД, содержащихся в результате поиска.

Алгоритм реализовывает механизм установления ассоциативных связей между описанием объекта поиска и компонентами в базе знаний, а также производить расчет мощностей этих связей и других показателей.

С точки зрения получения новых технических решений более интересен алгоритм логического вывода результатов поиска, который строит цепочки из компонентов, наиболее сильно ассоциирующихся с объектом поиска. Последний алгоритм в сущности есть модификация первого.

Основные модули системы представлены на рис.18 и 19.



Рисунок 18 - Основные части БЗ

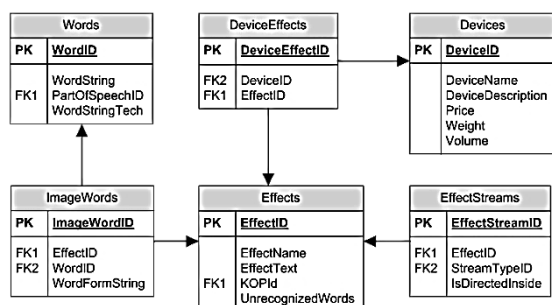


Рисунок 19 - Основные таблицы БЗ

Взаимодействие системы ассоциативного поиска с базой данных происходит посредством SQL-запросов, как наиболее эффективного средства взаимодействия с массивами структурированных данных. Базовые SQL-запросы представлены в таблице на рис.20.

Имя метода	Генерируемый SQL-запрос
GetEffectWords	"Select Words.WordID, WordString, PartOfSpeechName, Words.WordStringTech from (Words INNER JOIN ImageWords On Words.WordID = ImageWords.WordID) INNER JOIN PartsOfSpeech on Words.PartOfSpeechID = PartsOfSpeech.PartOfSpeechID WHERE ImageWords.EffectID = " + effectID
GetEffectImage	"Select Words.WordID, Words.WordString, Words.PartOfSpeechID, Words.WordStringTech FROM Words INNER JOIN ImageWords On Words.WordID = ImageWords.WordID WHERE ImageWords.EffectID = " + effectID
UpdateEffect	"Update Effects SET EffectName = " + effect.m_Name + ", EffectText=" + effect.m_Text + ", OperationID=" + effect.m_OperationID + " WHERE effectID=" + effect.m_ID
UpdateDevice	"Update Devices SET DeviceName = " + device.m_Name + ", DeviceVolume=" + device.m_Volume.ToString(CultureInfo.InvariantCulture) + ", DevicePrice=" + device.m_Price.ToString(CultureInfo.InvariantCulture) + " WHERE DeviceID=" + device.m_ID;

Рисунок 20-Основные SQL-запросы.

Опытные результаты и выводы

Первоначальные опытные результаты показали необходимость:

- 1) Сокращения перебора вариантов поиска даже для достаточно простых задач;
- 2) Полного восприятия и понимания результатов поиска, которые выдаются решателем.

Решение первой проблемы мы видим в использовании принципа «разделяй и властвуй».

Устранение второй проблемы потребует более активного использования с адаптацией понятий, которые представлены на рис.21.

	Формальная лингвистика
N	Noun (Существительное)
V	Verb (Глагол)
A	Adjective (Прилагательное)
Λ	Adverb (Наречие)
P	Preposition (Предлог)
τ	Determiner (определяющее слово)
δ	Degree word (Степень слова)
κ	Qualifier (спецификатор-слово)
α	Auxiliary (Вспомогательное слово)
γ	Conjunction word (совместное слово)
¬	Negative (отрицание)
NP	Noun phrase (Существительная фраза)
VP	Verb phrase (Глагольная фраза)
AP	Adjective phrase (Прилагательная фраза)
ΛP	Adverb phrase (Наречная фраза)
PP	Prepositional phrase (Предложная фраза)
CP	Complement phrase (Фраза дополнения)
⊢	To be (Быть)
⊢	To have (Иметь)
⊢	To do (Делать)
⊢ ... ⊢	Indirect to do (косвенно делать)

Рисунок 21 - Базовые понятия лингвистики

Литература

1. Ormrod, J. E. (1999). Human learning (3rd ed.). Prentice-Hall Inc..
2. Polya, G. (1954). Patterns of plausible inference. Princeton, NJ, USA: Princeton University Press.
3. Bender, E. A. (1996). Mathematical methods in artificial intelligence. Los Alamitos, CA, USA: IEEE CS Press.
4. Wilson, L. B., & Clark, R. G. (1988). Comparative programming language. Wokingham, UK: Addison-Wesley Publishing Co..
5. Newell, A., & Simon, H. A. (1972). Human problem solving. Prentice-Hall.
6. Tuma, D. T., & Reif, F. (1980). Problem solving and education. Hillsdale, NJ, USA: Lawrence Erlbaum Associates.
7. Payne, D. G., & Wenger, M. J. (1998). Cognitive psychology. Boston, NY, USA: Houghton Mifflin Company.
8. Matlin, M. W. (1998). Cognition (4th ed.). NY, USA: Harcourt Brace College Publishers.
9. Smith, M. U. (1991). Toward a unified theory of problem solving: Views from the content domains. Hillsdale, NJ, USA: Lawrence Erlbaum Associates.
10. Аристотель. Категории. ГСЭИ. Москва, 1939.

11. Райл Гилберт .Понятие сознания. Перевод с англ. — М.: Идея-Пресс, Дом интеллектуальной книги, 1999. — 408 с.
12. Прист С. Теории сознания. Перевод с англ. Грязнова А. Ф. - М.: Идея-Пресс, Дом интеллектуальной книги, 2000. - 286 с.
- 13.Фреге Г.Логика и логическая семантика: Сборник трудов-М.:Аспект Пресс,2000,512 с.
- 14.Р.Шенк. Обработка концептуальной информации.М.:Радио и связь,1980.
- 15.Wang, Yingxu. Software engineering foundations: a software science perspective. 2008 by Taylor & Francis Group, LLC.
- 16.Ning Zhong,Yingxu Wang *, Vincent Chiew. On the cognitive process of human problem solving// Cognitive Systems Research 11 (2010), 81–92.
17. Половинкин А.И. Основы инженерного творчества: Учеб. пособие для студентов втузов. — М.: Машиностроение, 1988. — 368 с.
18. Альтиуллер Г. С. Алгоритм изобретения. — М.: Московский рабочий. — 1969 (1-е изд.); 1973 (2-е изд.).
19. Веденов А.А. Моделирование элементов мышления. Москва: Издательство «Наука». Главная редакция физико-математической литературы, 1988.
20. http://www.triz-chance.ru/izobret_zad.php

Андрюхин А.И., Новиков Е.Т. Компьютерное решение проблем I.Ассоциативный поиск и рефлексивный вывод в когнитивных системах на основе SQL и NoSQL СУБД. В работе рассматривается решение проблем как один из основных человеческих когнитивных процессов. Представлено взаимодействие процесса решение проблем с такими когнитивными процессами, как абстракция, поиск, вывод, анализ и синтез на базе представления внутренних знаний известной моделью Вещь(объект)-атрибута-отношения (ВОС). После идентификации объекта проблемы, решение рассматривается как процесс поиска в пространстве памяти для нахождения взаимосвязи между набором целей решения и набором альтернативных путей. Исследуются комплекс когнитивных вычислений для описания и моделирования основных механизмов и процессов мозга. Выполнена попытка использования этой методики для проектирования новых технических решений с применением ассоциативного поиска и рефлексивного вывода в SQL и NoSQL СУБД.

Ключевые слова: Когнитивные вычисления, модель мозга, когнитивные процессы, решение проблем, концептуальная алгебра, ассоциативный поиск, рефлексивный вывод, NoSQL БД.

Andryukhin A.I., Novikov E.T. Computer Problem Solving. Associative search and reflexive inference in cognitive systems based on SQL and NoSQL DBMS.I. The paper considers the solution of problems as one of the main human cognitive processes. The interaction of the process of solving problems with such cognitive processes as abstraction, search, inference, analysis and synthesis based on the representation of internal knowledge by the known model of the Object-Attribute-Relationship (BOC) model is presented. After identifying the problem object, the solution is viewed as a process of searching in the memory space to find the relationship between the set of solution objectives and the set of alternative paths. The complex of cognitive calculations for the description and modeling of the basic mechanisms and processes of the brain is explored. An attempt was made to use this technique to design new technical solutions using associative searching and reflexive inference in SQL and NoSQL DBMS.

Keywords: Cognitive calculations, brain model, cognitive processes, problem solving, conceptual algebra, associative search, reflexive inference, NoSQL DB.

Статья поступила в редакцию 20.09.2016
Рекомендована к публикации д-ром физ.-мат. наук А.С. Миненко

УДК 004.04

Статистические компьютерные оценки характеристик многомерных операций и объектов. II.

Артеменко В.А., Андриюхин А.И., Филер З.Е.¹

Донецкий национальный технический университет

¹Кировоградский национальный педагогический университет,
alexandruckin@rambler.ru

Артеменко В.А., Андриюхин А.И., Филер З.Е. Компьютерные оценки характеристик многомерных операций и объектов. II. В работе рассматриваются известные классические многомерные объекты, такие как комплексные и гиперкомплексные числа, комплексные векторы и матрицы. Рассматриваются варианты построения скалярного отображения комплексных чисел. На базе этого отображения рассматриваются представления и свойства матриц. Приведены визуальные и стохастические оценки интервальных областей отображений для различных их параметров. Результаты работы, могут быть использованы при решении определенных вопросов построения устойчивых систем управления, теории интервального анализа и теории идентификация систем. При расчетах применялся пакет Wolfram Mathematica 11.2.

Введение

Настоящая работа является продолжением статьи [1], однако в ней основное внимание уделяется известным классическим многомерным объектам, таким как комплексные и гиперкомплексные числа, комплексным векторам и матрицам и определению операций сравнения их.

Основой многомерных операций является введение операций сравнения комплексных чисел. Напомним, что на ранних этапах изучения комплексных чисел применение обычных логических рассуждений приводило к неправильным выводам.

Так известно, что И.Бернулли, Даламбер и Лейбниц имели ошибочное мнение о значении логарифма отрицательного числа [2]. Только Л.Эйлер дал правильный ответ. Формулы Эйлера (см.рис.1) и Муавра позволяют уверенно решать любые вопросы связанные с комплексными числами и подчеркнем, что известные соотношения $e^{2\pi i}=1$ (или $e^{\pi i}=-1$), $i^i=0.20788$, $i^{\pi}=0.00719188$, $i^{ie}=0.0139836$ получены при работе с операцией возведения чисел в комплексную степень, которая в сущности представляет собой многомерную операцию.

Постановка задачи

Формулировка задачи проста: необходимо построить скалярное отображение комплексных чисел, что даст возможность решать уравнения и неравенства с ними.

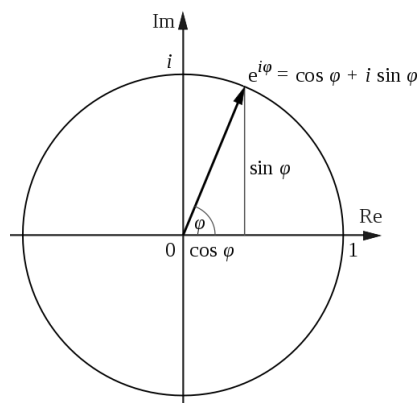


Рисунок 1 – Графическое представление комплексного нормированного числа

Необходимо подчеркнуть, что современные математические пакеты Wolfram Mathematica, Maple и другие, позволяют с помощью встроенных функций (модуль, аргумент, действительная и мнимая части комплексного числа) эффективно решать уравнения и неравенства с комплексными величинами.

К примеру, решение двух сотен уравнений для переменной M для заданных S, D при n=2,200, i=1, n с выдачей зависимости M(n)

$$\sum_{i=1}^n i^{S+Di} = \left(\sum_{i=1}^n i \right)^M$$

занимает время, меньшее 1 секунды для процессора I3 4170. График зависимости M(n) представлен на рис.2 при S=-1, D=2.

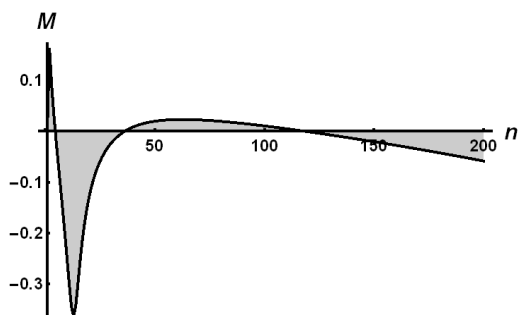


Рисунок 2 - Зависимость $M(n)$, $S=-1$, $D=2$

Несколько иной характер имеют результаты при $S=-1/2$ (гипотеза Римана), $D=2$ на рис.3.

Так как комплексное число является многомерной величиной (имеет две компоненты) возможно использовать методику решения известной в системном анализе проблемы оценки многокритериальных систем. Она состоит в выборе наилучшей системы из множества заданных систем $Q_1, Q_2, \dots, Q_n$, для которых заданы их оценки по критериям $K_1, K_2, \dots, K_m$.

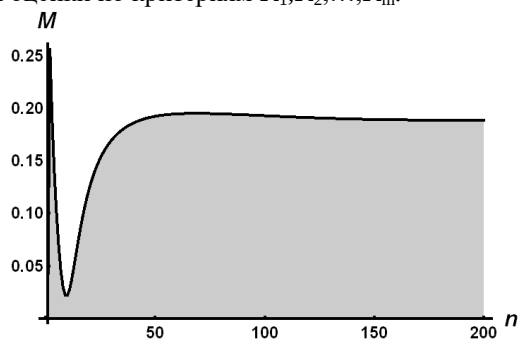


Рисунок 3 - Зависимость $M(n)$, $S=-1/2$, $D=2$

В таблице 1 элемент $X_{i,j}$ есть оценка объекта Q_i по критерию K_j .

Таблица 1 - Многокритериальные оценки систем

	K_1	...	K_i	...	K_m
Q_1					
...					
Q_i			$X_{i,i}$		
...					
Q_n					

Требуется определить «наилучший» объект. Согласно представлениям системного анализа мы должны построить интегральный критерий объекта (рассматриваемой системы).

Эти проблемы исследовались различными методами, важнейшими из которых являются:

1) методы, основанные на количественных измерениях, где:

а) методы, которые используют многокритериальную теорию полезности МАУТ [4-5];

б) методы, чьи результаты переводятся в количественный вид, использование метода аналитической иерархии (АНП) [6,7];

в) подходы использующие индикаторы при рассмотрении альтернатив в методах ELECTRE [8-9];

2) методы, основанные на качественных измерениях, при которых:

а) нет перехода к количественным переменным в вербальном анализе решений [10-13];

б) используют теорию нечетких множеств [14];

Обычно на практике используют мультиаддитивный интегральный критерий, и существует множество работ, в которых он описан [3].

Для применения мультиаддитивного интегрального критерия необходимо задавать приоритеты c_j для критериев K_j , $j=1, m$ и далее мы определяем «наилучший» объект согласно выражения $F = \max\{F_i, i=1, n\}$, $F_i = \{\sum c_j X_{i,j}, j=1, m\}$.

На рис.4-6 отображены представления выражения $(xi)^{yi}$ для действительных x, y для операции возведения чисел в комплексную степень.

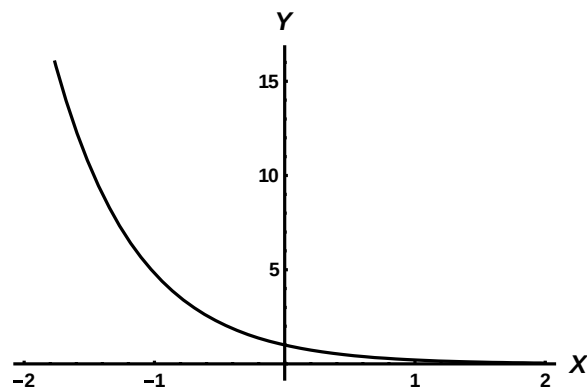


Рисунок 4 – График функции $Y=i^{xi}$

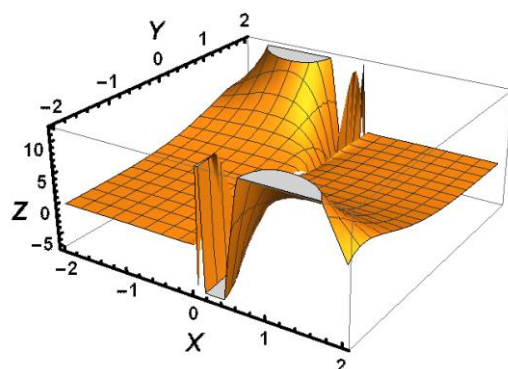


Рисунок 5 – График функции $Z=\text{Re}((xi)^{yi})$

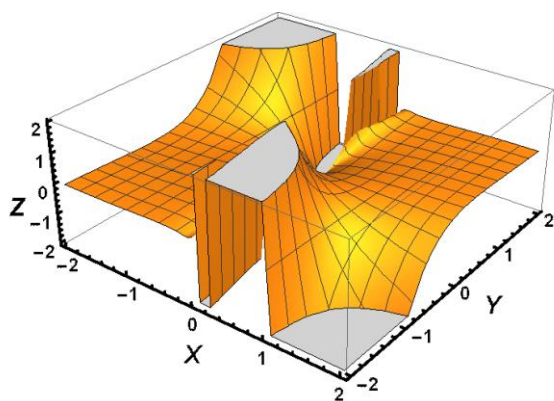


Рисунок 6 – График функции $Z = \text{Im}((xi)^{y_i})$

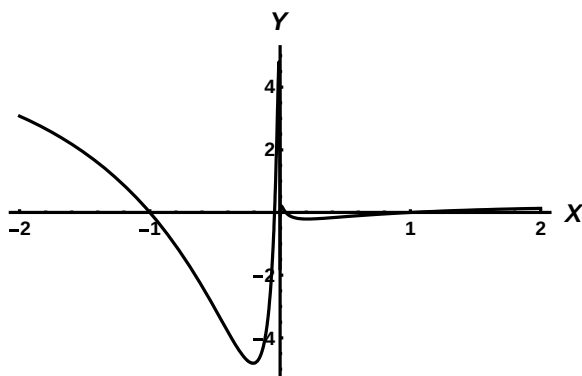


Рисунок 7 – График функции $Y = \text{Im}(xi)^i$

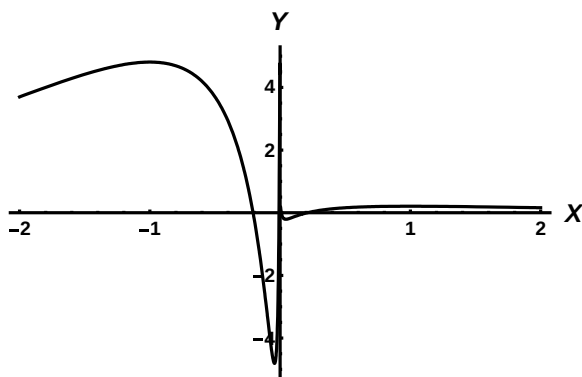


Рисунок 8 – График функции $Y = \text{Re}(xi)^i$

Многомерные операции для комплексных чисел

При построении многомерной операции авторы вначале пытались построить многомерную операцию сравнения двух комплексных чисел основываясь на операции возведения их в некоторую степень.

Определим многомерную операцию сравнения двух комплексных чисел $(r_1, \phi_1), (r_2, \phi_2)$, где r (ϕ), модуль(аргумент) соответственно.

Здесь мы ввели обозначение $(r, \phi) = re^{i\phi}$ и далее определим преобразование T , как

$$T((r_1, \phi_1), ((r_2, \phi_2)) = ((r_1, \phi_1))^P, ((r_2, \phi_2))^P \quad (1)$$

где показатель степени $P = 4\pi^2 / (\phi_1 \phi_2)$.

Компоненты вектора $((r_1, \phi_1))^P, ((r_2, \phi_2))^P$ являются действительными числами, т.к. $(r_1, \phi_1)^P = ((r_1, \phi_1)^{P_1})^{P_2}$, где $P_1 = 2\pi/\phi_1$, $P_2 = 2\pi/\phi_2$ и следовательно $P_1 \cdot P_2 = 4\pi^2 / (\phi_1 \phi_2)$.

Выражение $(r_1, \phi_1)^{P_1} = r_1^{P_1} e^{i\phi_1 P_1} = r_1^{P_1} e^{2\pi i} = r_1^{P_1}$ есть действительное число. Здесь мы использовали известное тождество $e^{2\pi i} = 1$.

Однако что практические расчеты при использовании этих преобразований показали, что это отображение обладает множеством недостатков:

1) имеем вычисления вида $2\pi/\phi$, т.е. особая точка при $\phi \rightarrow 0$;

2) комплексные числа, которые лежат на единичной комплексной окружности имеют одинаковую меру 1;

3) графические представления при использовании этого отображения имеют слишком сложный вид, не согласующийся с интуитивными представлениями.

Последнее обусловлено сложным поведением функций, представленных на рис.5-8.

Поиски привели к следующей формуле для отображения комплексного числа $(r, \phi) = re^{i\phi}$.

$$T(r, \phi) = r \sum_{i=1}^n \beta_i e^{g_i(\phi)} \quad (2)$$

Функция g должна удовлетворять основным условиям:

а) численные меры векторов комплексных чисел с одинаковым аргументом пропорциональны их модулям;

б) численные меры векторов комплексных чисел с одинаковым модулем, т.е. лежащих на комплексных окружностях одинакового радиуса, должны быть одинакового порядка.

Были построены численные меры для функций:

- 1) $g_1(r, \phi) = r \cdot \text{Exp}(\phi / (2\pi))$;
- 2) $g_2(r, \phi) = r \cdot \text{Exp}(-\phi / (2\pi))$;
- 3) $g_3(r, \phi) = r \cdot \text{Exp}(\cos(\phi) / (2\pi))$;
- 4) $g_4(r, \phi) = r \cdot \text{Exp}((-1/4 \cos(-\phi) - 3/4 \cos(\phi)) / (2\pi))$;
- 5) $g_5(r, \phi) = r \cdot \text{Exp}((-3/4 \cos(-\phi/2) - 1/4 \cos(\phi/2)) / (2\pi))$;
- 6) $g_6(r, \phi) = r \cdot \text{Exp}((-1/4 \sin(-\phi) - 3/4 \sin(\phi)) / (2\pi))$;
- 7) $g_7(r, \phi) = r \cdot \text{Exp}((-1/4 \sin(-\phi) - 3/4 \sin(\phi)) / (2\pi))$.

В табл.2 приведены основные характеристики приближений для этих зависимостей.

На рис.9-15 приведены графики функций $g_i(\phi)$, $i=1,7$ для $r=1$,

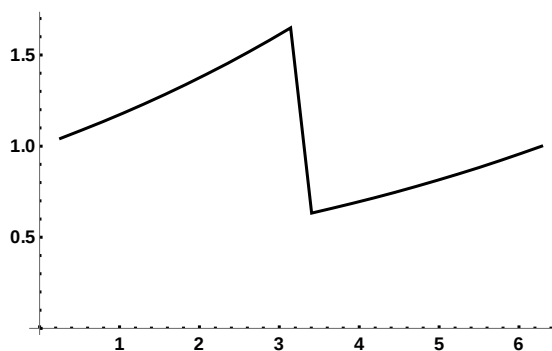


Рисунок 9 – График функции $g1(\varphi)$

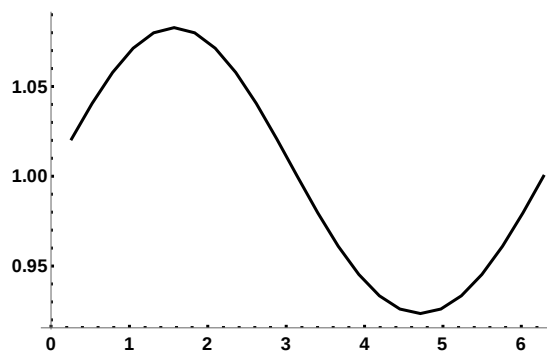


Рисунок 13 – График функции $g5(\varphi)$

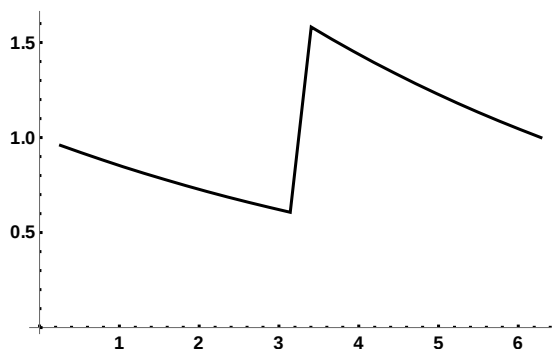


Рисунок 10 – График функции $g2(\varphi)$

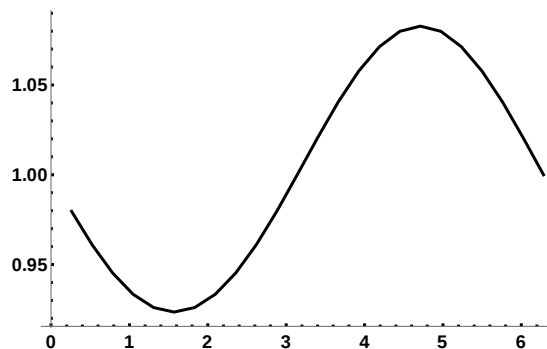


Рисунок 14 – График функции $g6(\varphi)$

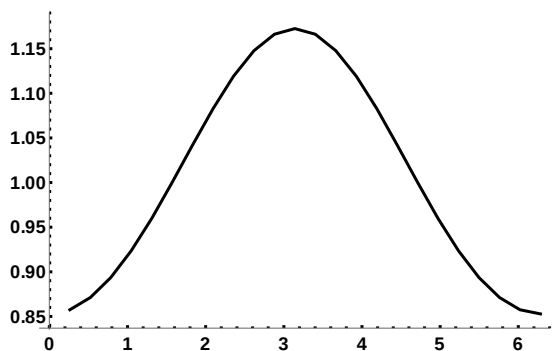


Рисунок 11 – График функции $g3(\varphi)$

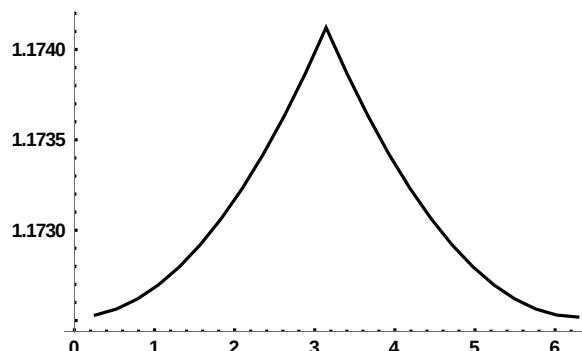


Рисунок 15 – График функции $g7(\varphi)$

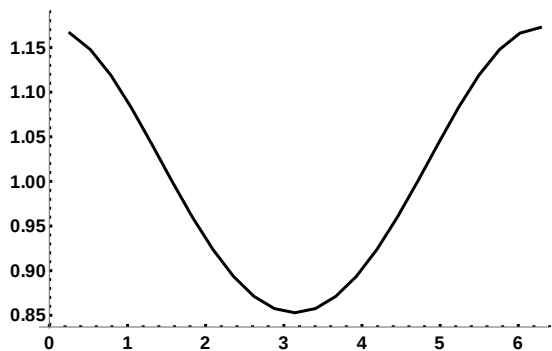


Рисунок 12 – График функции $g4(\varphi)$

Таблица 2 - Характеристики приближений

№	Функция	Среднее	Сред.кв.отклонение
1	$g1$	1.06405	0.2623
2	$g2$	1.02063	0.25169
3	$g3$	1.00634	0.10155
4	$g4$	1.00634	0.10155
5	$g5$	0.904881	0.0390529
6	$g6$	1.00158	0.0505385
7	$g7$	1.00158	0.0505385

Приведем примеры решения комплексных неравенств с использованием функций отображения $G1(r, \varphi) = r \cdot \text{Exp}(\varphi / (2\pi))$, $G2(r, \varphi) = r \cdot \text{Exp}(-\varphi / (2\pi))$, $G3(r, \varphi) = r \cdot \text{Exp}(-\varphi / (4\pi))$. Введем обозначения $Z1 = -1 + i$, $Z2 = 3 + 2i$. На рис.16-29 имеем результаты сравнения.

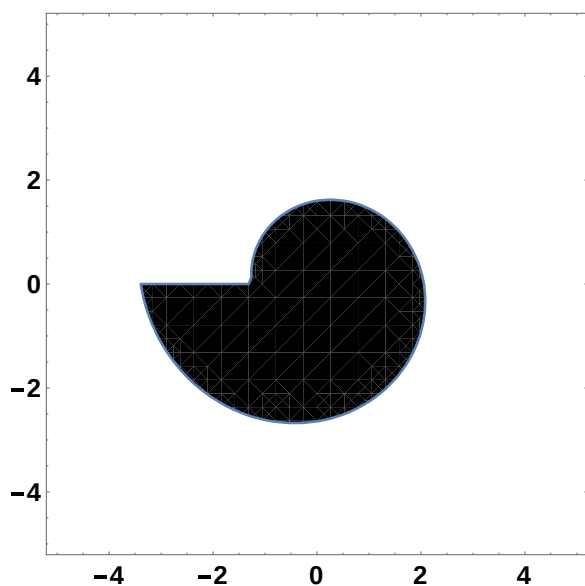


Рисунок 16 – Комплексные числа меньшие числа Z_1 для $G_1(r, \phi)$

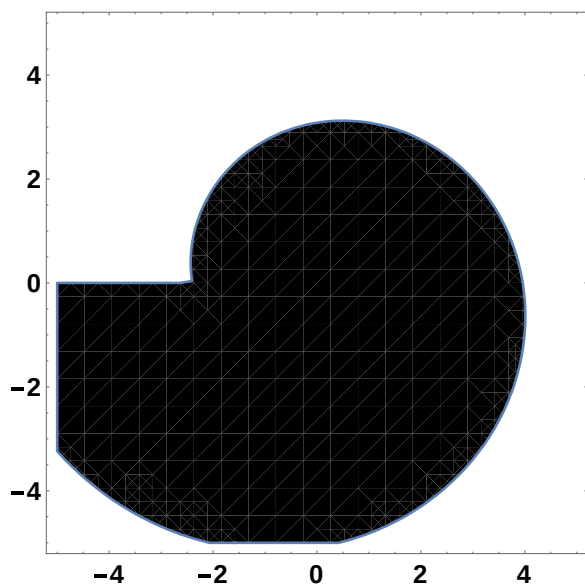


Рисунок 17 – Комплексные числа меньшие числа Z_2 для $G_1(r, \phi)$

Далее мы представим результаты сравнения комплексные числа в их отношениях по сравнению с числами Z_1 и Z_2 для $G_1(r, \phi)$.

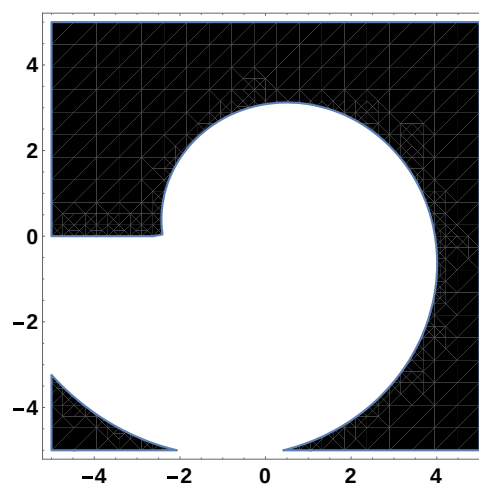


Рисунок 18 – Комплексные числа больше чем числа Z_2 и числа Z_1 для $G_1(\phi)$

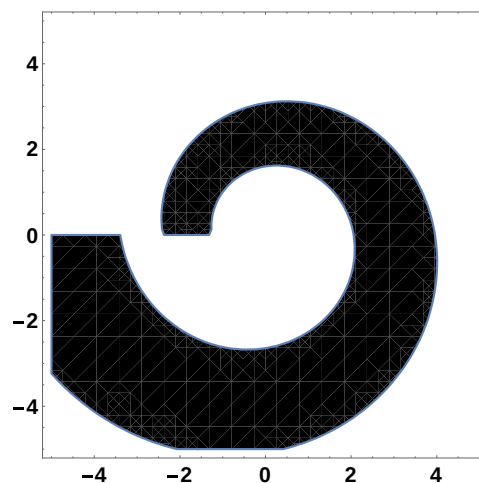


Рисунок 19 – Комплексные числа больше чем числа Z_1 и меньше, чем число Z_2 для $G_1(r, \phi)$

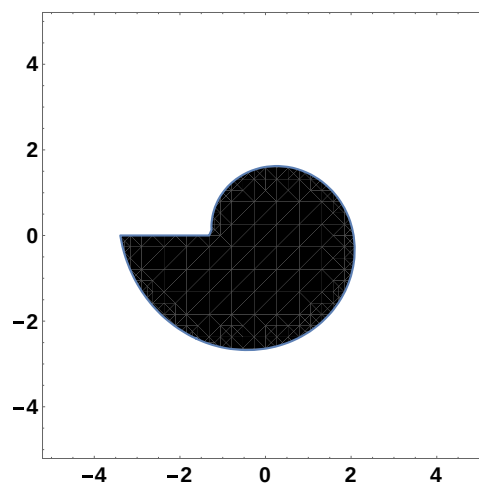


Рисунок 20 – Комплексные числа меньше чем числа Z_1 и меньше, чем число Z_2 для $G_1(r, \phi)$

Ясно, что комплексных числа больше чем число Z_2 и меньше, чем число Z_1 для $G_1(\phi)$ не существует и в целях экономии текста не помещаем пустой рисунок для этого варианта.. Для критерия $G_2(\phi)$ соответствующие области имеют следующий вид.

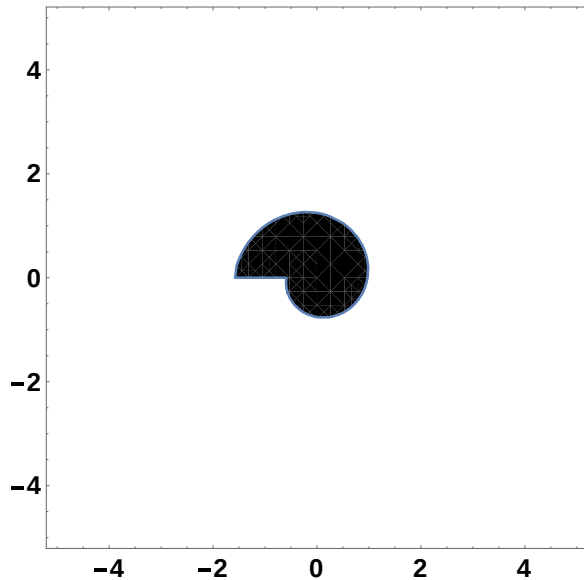


Рисунок 21 – Комплексные числа меньше числа Z_1 для $G_2(r, \phi)$

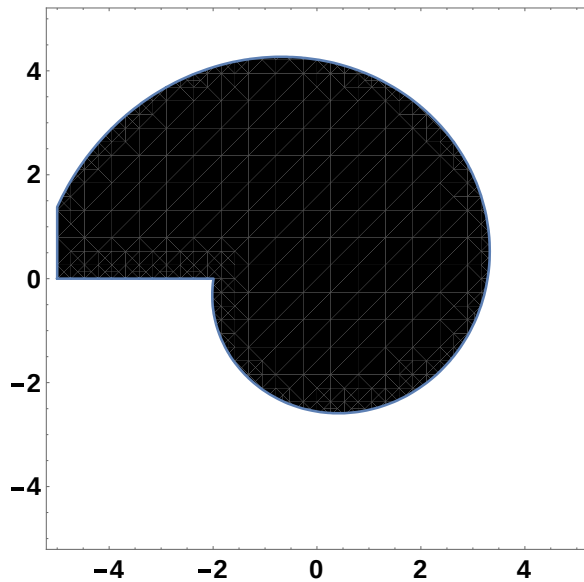


Рисунок 22 – Комплексные числа меньше числа Z_2 для $G_2(r, \phi)$

Аналогично в случае с критерием $G_1(\phi)$, мы представим комплексные числа в их отношениях по сравнению с числами Z_1 и Z_2 для $G_2(\phi)$.

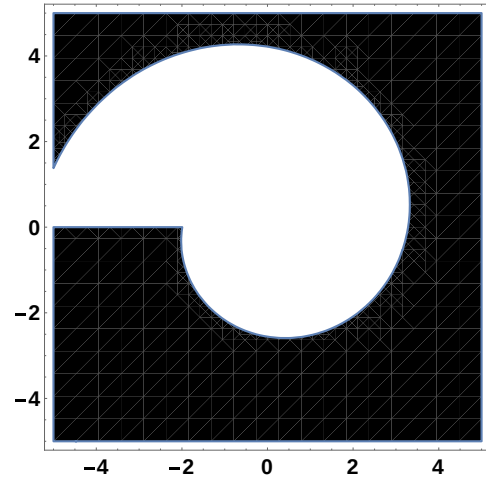


Рисунок 23 – Комплексные числа больше чем числа Z_2 и число Z_1 для $G_2(r, \phi)$

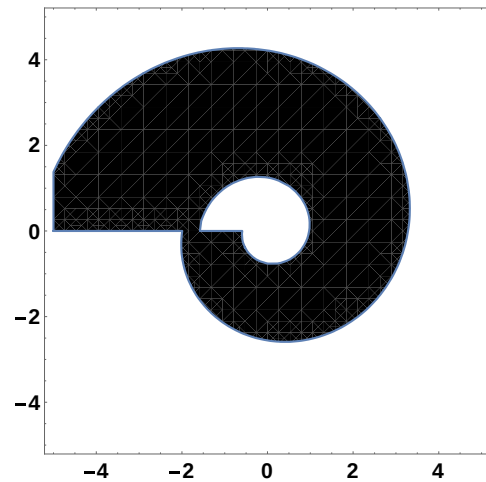


Рисунок 24 – Комплексные числа больше чем числа Z_1 и меньше, чем число Z_2 для $G_2(r, \phi)$

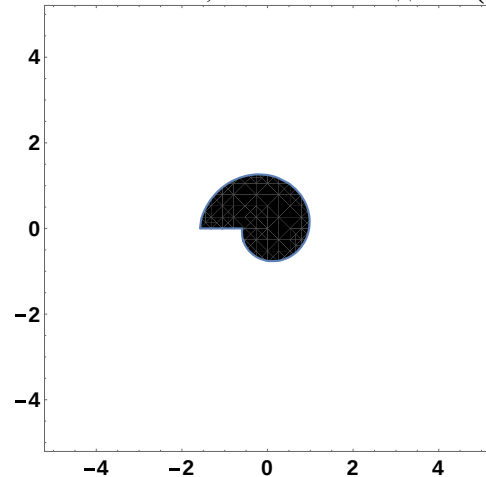


Рисунок 25 – Комплексные числа меньше чем числа Z_1 и меньше, чем число Z_2 для $G_2(r, \phi)$

Как и ранее пропускаем случай, когда комплексные числа больше чем числа Z_2 и меньше, чем число Z_1 для $G_2(\phi)$.

Ниже представлены области сравнения для $G3(\phi)$.

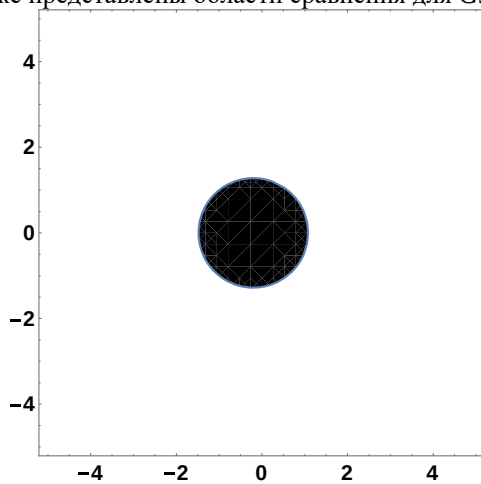


Рисунок 26 – Комплексные числа меньше числа $Z1$ для $G3(r, \phi)$

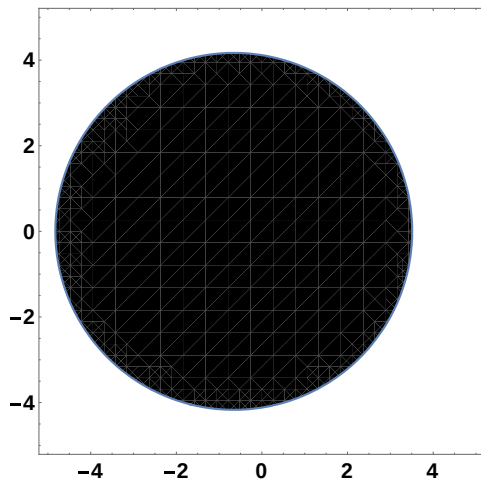


Рисунок 27 – Комплексные числа меньше числа $Z2$ для $G3(r, \phi)$

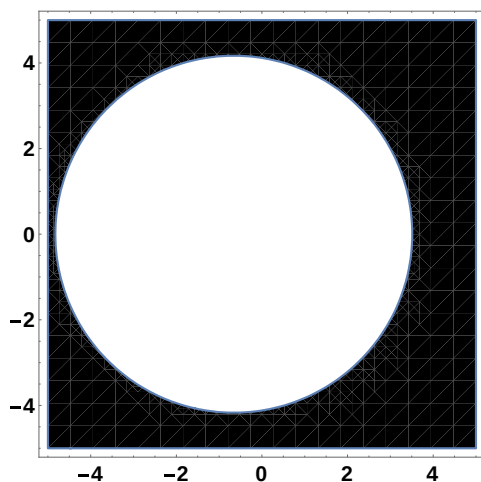


Рисунок 28 – Комплексные числа больше чем числа $Z2$ и число $Z1$ для $G3(r, \phi)$

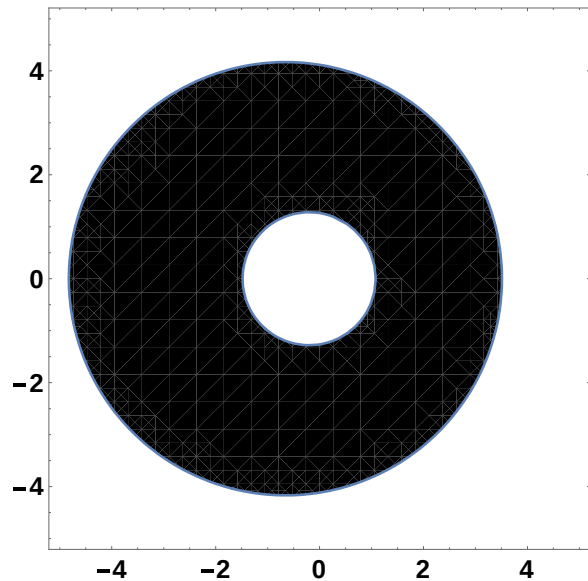


Рисунок 29 – Комплексные числа больше чем числа $Z1$ и меньше , чем число $Z2$ для $G3(r, \phi)$

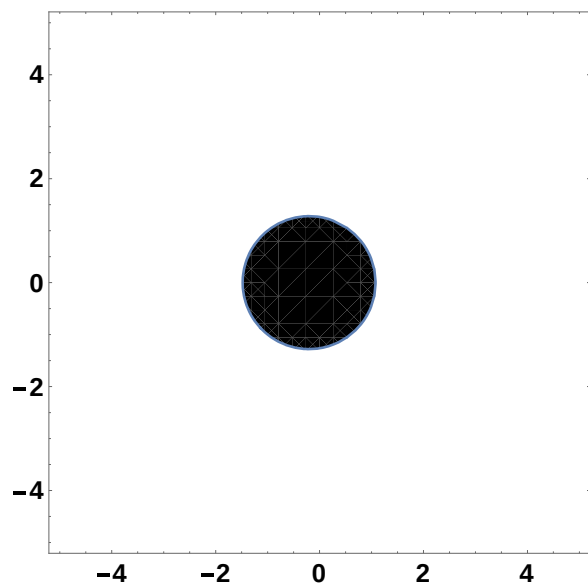


Рисунок 30 – Комплексные числа меньше чем числа $Z1$ и меньше , чем число $Z2$ для $G3(r, \phi)$

Опять пропускаем случай, когда комплексные числа больше чем числа $Z2$ и меньше, чем число $Z1$ для $G3(r, \phi)$.

Многомерные операции для комплексных матриц

Матрицы характеризуются своим спектром, т.е. множеством собственных чисел. Было предложено множество решений этой проблемы. Поэтому задача сравнения матриц сводится к задаче сравнения векторов.

Наличие множества переменных характеризующих комплексные значения

собственных чисел позволяет визуально представлять области сравнения только для квадратных матриц размера 2×2 , у которых одно собственное число является действительным. Так фиксируя матрицу $A = \{\{2 + I, -1 - 2 \cdot I\}, \{-0.5 - I, 2 + 0.3 \cdot I\}\}$, определяем ее собственные числа $\{2.69024 + 2.09877 I, 1.30976 - 0.798769 I\}$. Далее рассматриваем матрицы вида $\{\{X, 0\}, \{0, Y + I Z\}\}$ и можем визуально строить для них области $\{X, Y, Z\}$, для которых эти матрицы меньше заданной A по различным критериям $G1(r, \phi) = r \cdot \text{Exp}(\phi/(2\pi))$, $G2(r, \phi) = r \cdot \text{Exp}(-\phi/(2\pi))$, $G3(r, \phi) = r \cdot \text{Exp}(\cos(-\phi/(2\pi)))$.

На рис.29-31 представлены области сравнения для суммы собственных чисел.

На рис.32-34 представлены области сравнения для произведения собственных чисел. Таким образом мы сравниваем векторы с комплексными компонентами.

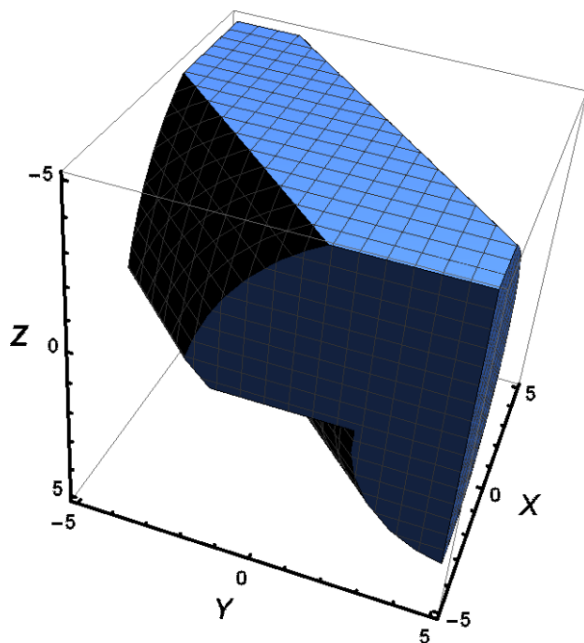


Рисунок 29 – Область определения для суммы собственных чисел для $G1$

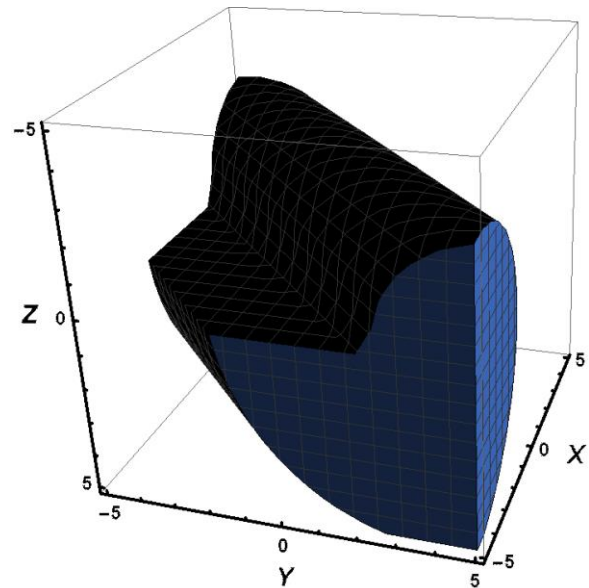


Рисунок 30 – Область определения для суммы собственных чисел для $G2$

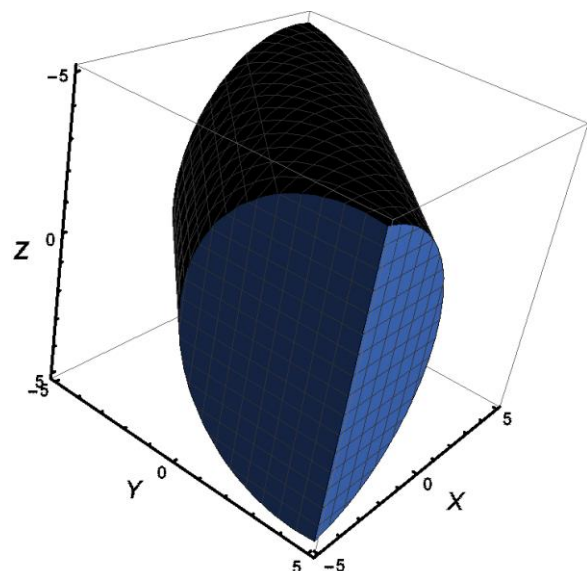


Рисунок 31 – Область определения для суммы собственных чисел для $G3$.

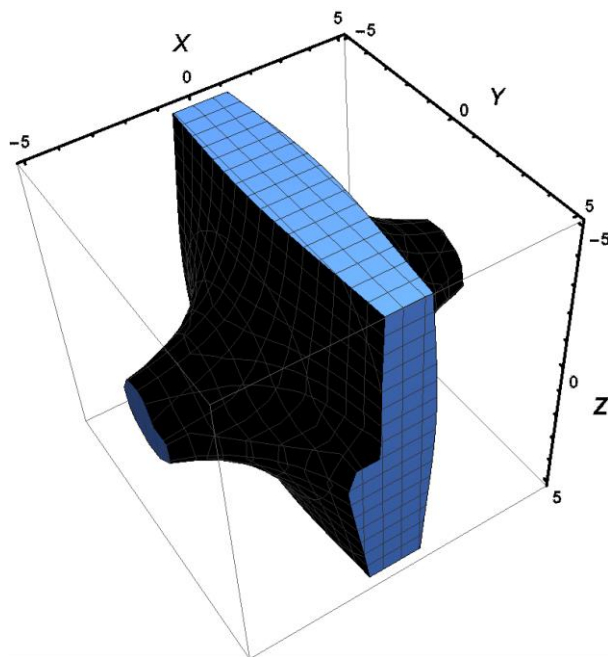


Рисунок 32 – Область определения для произведения собственных чисел для $G1$

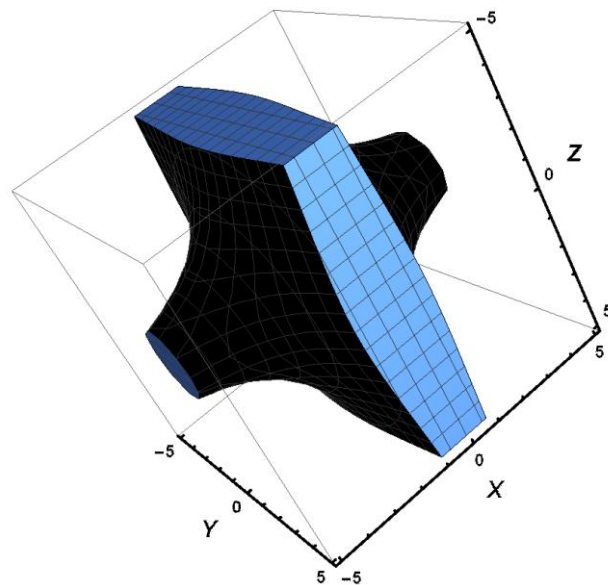


Рисунок 34 – Область определения для произведения собственных чисел для $G3$

Сравнения комплексных чисел для операции T

На рис.35-40 представлены результаты сравнения комплексных чисел при использовании преобразования T . Последнее определено в (1).

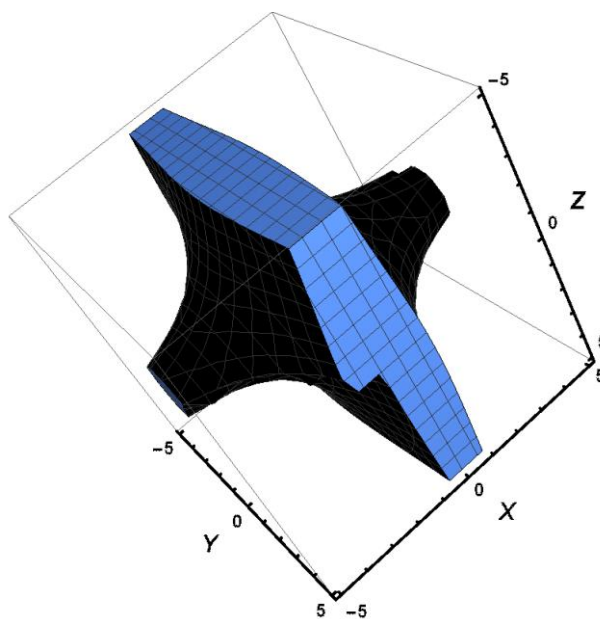


Рисунок 33 – Область определения для произведения собственных чисел для $G2$

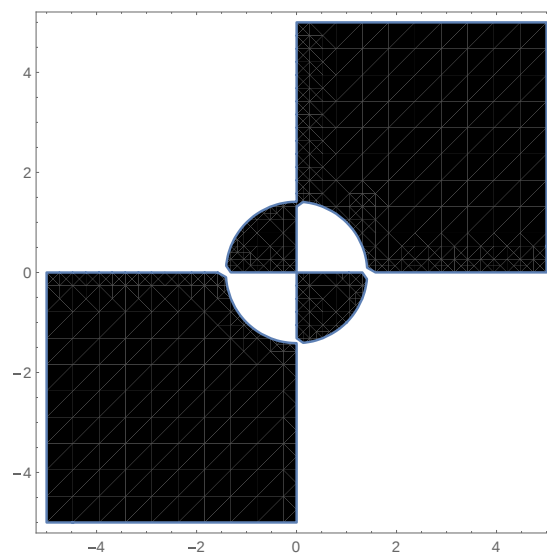


Рисунок 35 – Комплексные числа меньше числа $Z1$ для T

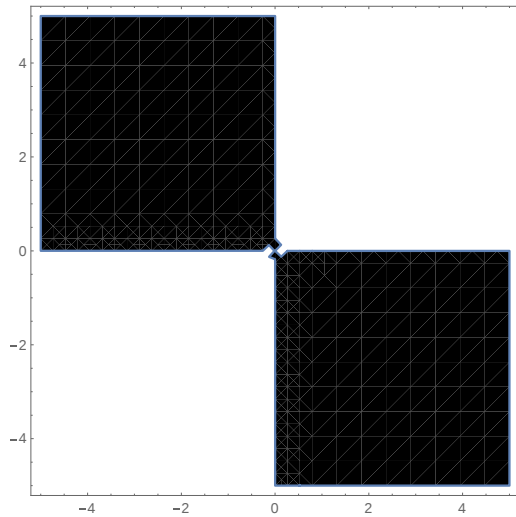


Рисунок 36 – Комплексные числа большие числа Z_2 для T

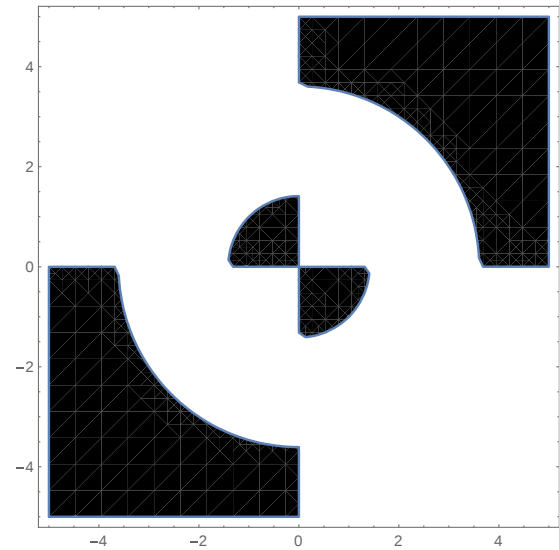


Рисунок 39 – Комплексные числа меньше чем числа Z_1 и больше, чем число Z_2 для T

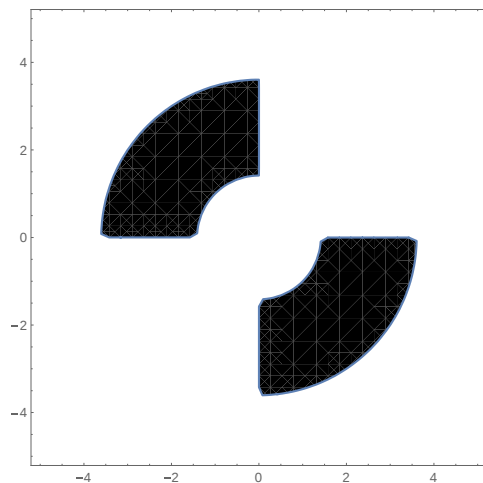


Рисунок 37 – Комплексные числа больше чем числа Z_2 и число Z_1 для T

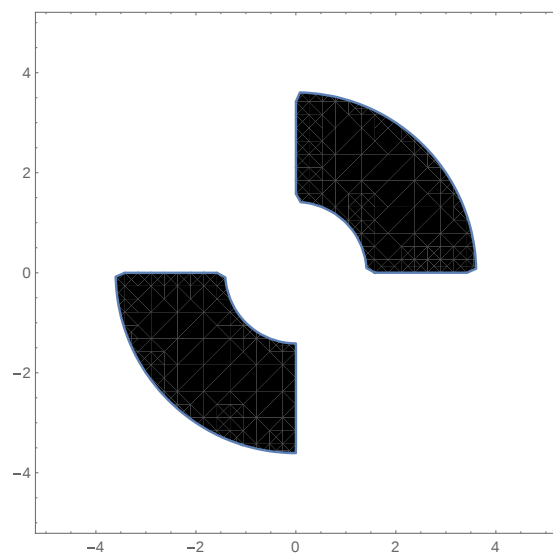


Рисунок 40 – Комплексные числа меньше чем числа Z_1 и меньше, чем число Z_2 для T

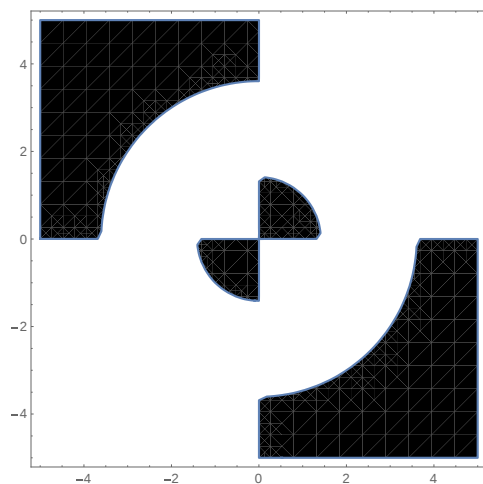


Рисунок 38 – Комплексные числа больше чем числа Z_1 и меньше, чем число Z_2 для T

Решение комплексных уравнений методом Монте-Карло

Использование различных скалярных отображений позволяет оценивать ненаблюдаемые возмущения. Однако интерпретация получаемых результатов затруднительна и сами результаты могут быть достаточно отличающимися друг от друга.

Для примера рассмотрим в общем случае решение уравнения $G(Z)=1$, где в первом случае $G(Z)$ есть класс отображений вида $G_1(Z)=|Z|(\text{Exp}((1-2A) \cdot \text{Arg}(Z)/2\pi))$, зависящий от параметра A . Во втором случае $G_2(Z)=|Z|(\text{Exp}((1-2A) \cdot \sin(\text{Arg}(Z))/2\pi))$. Будем считать, что $\text{Re}(Z) \in (-1, 1)$, $\text{Im}(Z) \in (-1, 1)$, $A \in (0, 1)$. Решения уравнения $G(Z)=1$ определяем неравенствами $|G(Z)-1| < \text{Eps}$.

На рис.41-42 (43-44) представлены области решений для G1 (G2), соответственно.

Значение P определялось с помощью метода Монте-Карло и отражает процентное отношение объема найденной области решений к объему всего пространства решений.

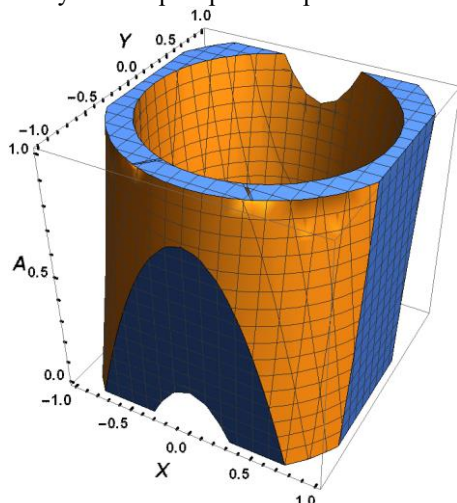


Рисунок 41 – Область решения: $Eps=0.1$, $P=23.9\%$

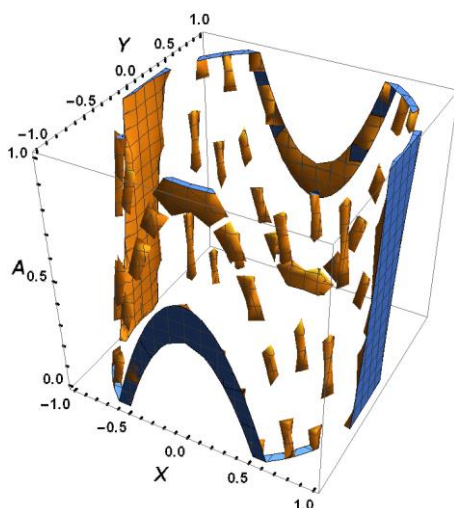


Рисунок 42 – Область решения: $Eps=0.025$, $P=6.5\%$

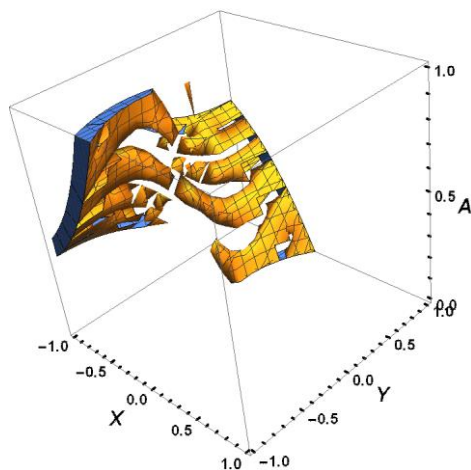


Рисунок 43 – Область решения: $Eps=0.1$, $P=3.9\%$

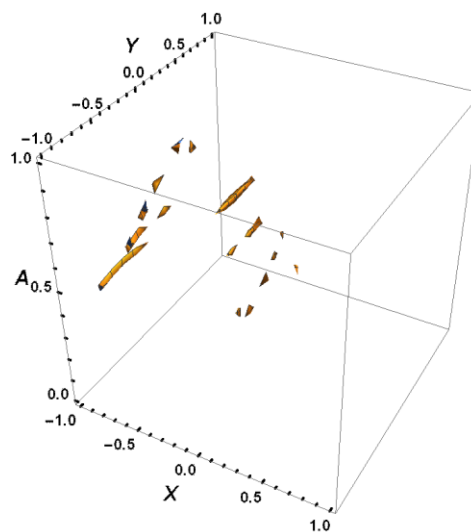


Рисунок 44 – Область решения: $Eps=0.025$, $P=1\%$

Последние результаты использовали подходы и идеи, изложенные в [15-18].

Выводы

Научная новизна работы заключается в том, что получены варианты построения скалярного отображения комплексных чисел. На базе этих отображения рассматриваются возможности скалярного представления матриц.

Заметим, что отображение T является более гибким, нежели скалярные интегральные функции, так как позволяют всегда находить решения для комплексных неравенств.

Приведены визуальные и стохастические оценки интервальные областей отображений для различных их параметров.

Практическая значимость результатов состоит в получении интервальных оценок параметров для некоторых операций, выполняемых с многомерными величинами.

Литература

1. Андрюхин А. И. Статистические компьютерные оценки характеристик многомерных операций и объектов. I. // Информатика и кибернетика, № 1, 2015, с.16-24
2. Балк М. Б. и др. Реальные применения мнимых чисел. К.: Рад. шк., 1988. — 255 с.
3. Complex Analysis for Mathematics and Engineering
4. Кини Р. Л. Размещение энергетических объектов: выбор решений. — М.: Энергоатомиздат, 1983. — 320 с.
5. Кини Р. Л., Райфа Х. Принятие решений при многих критериях: предпочтения и замещения. — М.: Радио и связь, 1981. — 560 с.

6. Саати Т. Принятие решений. Метод анализа иерархий. — М.: Радио и связь, 1993. — 278 с.
7. Саати Т. Л. Принятие решений при зависимостях и обратных связях: Аналитические сети. М.: Издательство ЛКИ, 2008. — 360 с.
8. Figueira J., Mousseau V., Roy B. ELECTRE methods // Multiple criteria decision analysis: state of the art surveys / Ed. by J. Figueira, S. Greco, M. Ehrgott. — International Series in Operations Research & Management Science. Boston: Springer, 2005. — Pp. 609–637.
9. Руа Б. Классификация и выбор при наличии нескольких критериев (метод Электра) // Вопросы анализа и процедуры принятия решений. Сборник переводов / Под ред. И. Ф. Шахнова. — М.: Мир, 1976. — С. 80–107.
10. Ларичев О. И. Вербальный анализ решений / Под ред. А. Б. Петровского. — М.: Наука, 2006. — 181 с.
11. Ларичев О. И. Теория и методы принятия решений, а также хроника событий в волшебных странах. — 2-е изд. — М.: Логос, 2002. — 392 с.
12. Петровский А. Б., Ройзензон Г. В. Многокритериальный подход к построению интегральных показателей // Таврический Вестник Информатики и Математики. — 2008. — № 2. — С. 143–150.
13. Петровский А. Б. Пространства множеств и мультимножеств. — М.: Едиториал УРСС, 2003. — 248 с.
14. Заде Л. Понятие лингвистической переменной и его применение к принятию приближенных решений / Под ред. А. Н. Колмогорова, С. П. Новикова. Математика. Новое в зарубежной науке № 3. — М.: Мир, 1976. — 168 с.
15. С.П. Шарый. Конечномерный интервальный анализ, 2013. — 613 с. — <http://www.nsc.ru/interval>
16. Челноков Ю.Н. Кватернионные и бикватернионные модели и методы механики твердого тела и их приложения. Геометрия и кинематика движения. — М.: Физматлит, 2006. — 512 с.
17. A.J.Hanson. Visualizing quaternions. Morgan Kaufmann is an imprint of Elsevier 2006 Elsevier Inc..
18. Todd A. E., N. Le Bihan, S. J. Sangwine. Quaternion Fourier Transforms for Signal and Image Processing. ISTE Ltd and John Wiley & Sons, Inc, 2014.

Artemenko VA, Andryukhin A.I., Filer Z.E. Computer estimates of the characteristics of multidimensional operations and objects. II. We consider well-known classical multidimensional objects, such as complex and hypercomplex numbers, complex vectors, and matrices, and the definition of their comparison operations. We consider variants of constructing a scalar mapping of complex numbers. Representations and properties of matrices are considered on the basis of this mapping. Visual mock-stochastic estimates of interval mapping regions for various parameters are presented. The results of the work can be used to solve certain problems of constructing stable control systems, the theory of interval analysis, and the theory of system identification. In calculations, Wolfram Mathematica 11.2 was used.

Keywords: scalar mapping, multidimensional operation, complex number, matrix, multidimensional quantities, interval analysis.

Артеменко В.А., Андриюхин А.И., Филер З.Е. Компьютерные оценки характеристик многомерных операций и объектов. II. В работе рассматриваются известные классические многомерные объекты, такие как комплексные и гиперкомплексные числа, комплексные векторы и матрицы. Рассматриваются варианты построения скалярного отображения комплексных чисел. На базе этого отображения рассматриваются представления и свойства матриц. Приведены визуальные и стохастические оценки интервальных областей отображений для различных их параметров. Результаты работы, могут быть использованы при решении определенных вопросов построения устойчивых систем управления, теории интервального анализа и теории идентификация систем. При расчетах применялся пакет Wolfram Mathematica 11.2.

Ключевые слова: скалярное отображение, многомерная операция, комплексное число, матрица, многомерные величины, интервальный анализ.

Статья поступила в редакцию 20.09.2016
Рекомендована к публикации д-ром физ.-мат. наук А.С. Миненко

Алгоритмы размещения файлов в компьютерных сетях

Д.В. Бельков

Донецкий национальный технический университет
belkov65@list.ru

Бельков Д.В. Алгоритмы размещения файлов в компьютерных сетях. В работе предлагаются новые алгоритмы для решения актуальной научной задачи оптимизации размещения файлов в компьютерных сетях. Они могут быть использованы для повышения эффективности компьютерной сети на этапах ее проектирования и эксплуатации. Критерием оптимальности размещения файлов является суммарный поток локальных запросов, инициированных в узлах сети в единицу времени. С увеличением потока локальных запросов эффективность функционирования сети увеличивается за счет уменьшения времени отклика на запросы к файлам. В работе приведены результаты вычислительного эксперимента.

Ключевые слова: Файлы, узлы компьютерной сети, методы размещения файлов.

Введение

В связи с интенсивным развитием средств вычислительной техники широкое распространение получили компьютерные сети с распределенной обработкой информации. Большой вклад в решение задач повышения эффективности компьютерных сетей внесли В.М. Глушков, А.А. Стогний, Е.Л. Ющенко, В.Н. Редько. Однако в связи с массовым распространением распределенных систем, работы в этом направлении не утрачивают актуальности и требуют дальнейшего развития. Одним из способов повышения эффективности функционирования компьютерных сетей является оптимизация размещения файлов. Поэтому задача оптимального размещения файлов по узлам сети имеет важное практическое значение. Эта задача относится к классу NP - трудных. Точные методы можно применять только для решения задач малой размерности. В случае большой размерности необходимо использовать приближенные методы. Поэтому актуальной является разработка новых методов размещения файлов [1-6]. Целью данной работы является повышение эффективности функционирования компьютерной сети за счет оптимизации размещения файлов. Задачи работы:

1. Разработать алгоритмы размещения файлов в компьютерной сети, которые позволяют уменьшить время решения этой задачи за счет исключения полного перебора вариантов решений.
2. Исследовать точность решений задачи распределения файлов, разработанными алгоритмами.

Анализ работ по размещению файлов в компьютерных сетях свидетельствует, что для

обеспечения эффективного функционирования распределенной системы необходимо рациональное размещение файлов по узлам компьютерной сети в зависимости от интенсивности запросов к ним. Наиболее рационально размещать файл в тот узел, который чаще других выдает запрос на его использование. Это позволяет существенно повысить эффективность функционирования сети за счет уменьшения нагрузки на каналы связи. Для статического размещения файлов по узлам компьютерной сети необходимо при фиксированных значениях интенсивностей запросов к файлам так распределить файлы по узлам компьютерной сети, чтобы время отклика сети было минимальным.

Задача размещения файлов

В данной работе для уменьшения времени отклика оптимизируется интенсивность информационного трафика. При функционировании сети в каждом узле образуются два типа запросов: сетевой запрос, для обработки которого необходим файл, не содержащийся в том узле, где возник запрос и локальный запрос, для обработки которого необходим файл, содержащийся в том узле, где возник запрос. Критерием оптимальности размещения файлов является суммарный поток локальных запросов, инициированных в узлах в единицу времени. Чем больше суммарный поток локальных запросов, тем меньше время отклика сети.

Обозначим: F_{ij} - количество запросов к файлу i из узла j в единицу времени; $X_{ij} = 1$, если файл i расположен в узле j , иначе $X_{ij} = 0$;

V_i - объем файла i ; B_j - объем узла j , $i=1...m$, $j=1...n$.

Для рационального использования памяти узлов целесообразно минимизировать объем их свободной памяти. Поэтому коэффициент заполнения узлов, равный отношению V_i / B_j необходимо максимизировать. Задача размещения файлов по узлам компьютерной сети имеет вид:

Целевая функция

$$L = \sum_{i=1}^m \sum_{j=1}^n F_{ij} V_i X_{ij} / B_j = \sum_{i=1}^m \sum_{j=1}^n L_{ij} X_{ij} \rightarrow \max \quad (1)$$

Ограничения:

$$X_{ij} \in \{0,1\}, \sum_{j=1}^n X_{ij} = 1, i=1...m \quad (2)$$

$$\sum_{i=1}^m V_i X_{ij} \leq B_j \quad (3)$$

В задаче (1)-(3) необходимо найти матрицу размещений файлов X . В задаче максимизируется суммарный поток локальных запросов [6]. Увеличение потока локальных запросов связано с увеличением эффективности функционирования сети следующим образом.

Пусть $\sum_{i=1}^m \sum_{j=1}^n F_{ij}$ - суммарная интенсивность всех

запросов, $\sum_{i=1}^m \sum_{j=1}^n F_{ij} X_{ij}$ - суммарная интенсивность

локальных запросов, $(\sum_{i=1}^m \sum_{j=1}^n F_{ij} - \sum_{i=1}^m \sum_{j=1}^n F_{ij} X_{ij})$ -

суммарная интенсивность сетевых запросов, R_0 - среднее время выполнения сетевого запроса, X_0 - максимальная пропускная способность сети, R - среднее время отклика сети. Среднее время отклика на один запрос совпадает со средним временем ожидания обслуживания сетевого запроса. Оно состоит из среднего времени доступа к каналу связи и среднего времени выполнения запроса. Значение R определяется по формуле

$$R = R_0 \cdot (\sum_{i=1}^m \sum_{j=1}^n F_{ij} - \sum_{i=1}^m \sum_{j=1}^n F_{ij} X_{ij}) / X_0. \quad \text{Критерий}$$

$$\sum_{i=1}^m \sum_{j=1}^n F_{ij} X_{ij} V_i / B_j \rightarrow \max \quad \text{задачи размещения}$$

файлов обеспечивает максимизацию значений $\sum_{i=1}^m \sum_{j=1}^n F_{ij} X_{ij}$ и, следовательно, минимизацию значения R .

Таким образом, максимизация интенсивности локальных запросов к файлам приводит к повышению эффективности функционирования сети за счет уменьшения среднего времени отклика на запросы к файлам.

Известно [7], что интенсивность запросов к файлам в Internet меняется в пределах от 0 до 1000 запросов в секунду и подчиняется вероятностному распределению Парето

$$F(x) = P(X \leq x) = \begin{cases} 0, & x < k \\ 1 - (k/x)^\alpha, & x \geq k, \alpha < 2 \end{cases}, \quad \text{где}$$

$P(X \leq x)$ - вероятность того, что значение случайной величины X не превысит заданное число x . Параметр $\alpha = \alpha_1 = 1,5$. Большинство файлов находится в диапазоне 100–100000 байт. Распределение объемов файлов подчиняется вероятностному распределению Парето с параметром $\alpha_2 = 1,1$.

Вычислительные эксперименты по размещению файлов среди узлов сети проведены для случая $E_1 = 500$ запросов в секунду, $k_1 = (\alpha_1 - 1) \cdot E_1 / \alpha_1 = 166,67$ запросов в секунду. Интенсивность запросов к i -му файлу из j -го узла вычислялась по формуле: $F_{ij} = k_1 / (1 - \gamma)^{1/\alpha_1}$ (запросов в секунду), где γ - случайная величина, равномерно распределенная на интервале $(0;1)$, $E_2 = 50000$ байт, $k_2 = (\alpha_2 - 1) \cdot E_2 / \alpha_2 = 4545,46$ (байт). Объем i -го файла вычислялся по формуле: $V_i = k_2 / (1 - \gamma)^{1/\alpha_2}$ (байт), $i=1,2,...,8$; $j=1,2,3$. В ходе вычислительных экспериментов предлагаемыми в работе методами и методом полного перебора решается задача распределения m файлов среди n узлов сети, $m=8$, $n=3$. Для полного перебора требуется $n^m = 3^8 = 6561$ итерация, предлагаемые методы выполняют 1000 итераций. На каждой итерации формируется матрица размещения файлов, и вычисляется целевая функция (ЦФ) по формуле (1). Программы разработаны в среде Delphi.

Алгоритм имитации отжига

В данном разделе для решения задачи размещения файлов предлагается алгоритм „Машина Коши” (МК), относящийся к классу методов имитации отжига. Алгоритм работает следующим образом. „Температура” T изменяется по формуле $T := T_0 / \tau^\alpha$, $T_0 = 1000$, $\alpha = 1,5$, τ - номер итерации. Вероятность выбора узла j для размещения i -го файла вычисляется по формуле $P_j := C_j / \sum_{j=1}^n C_j$, где

$$C_j = \frac{1}{1 + \exp\{-F_{ij} / (r \cdot T)\}}, \quad r - \text{максимальное}$$

текущее значение ЦФ на очередной итерации. Первоначально $r=1000$. Узел выбирается по правилу рулетки. Если найденное решение не

удовлетворяет условию (3) задачи, то оно отвергается. Результаты вычислительного эксперимента показаны на рис. 1. Получено строго оптимальное решение на 984 итерации.

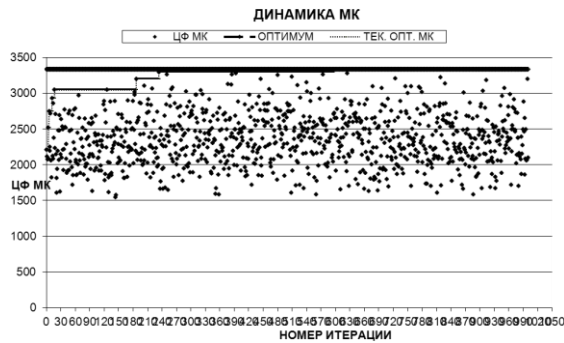


Рисунок 1 – Динамика поиска решений методом имитации отжига

Муравьиный алгоритм

В данном разделе для решения задачи размещения файлов предлагается алгоритм, относящийся к классу алгоритмов муравьиной колонии. Этот класс алгоритмов появился в результате исследований поведения живых муравьев. Муравей, двигаясь по определенному маршруту, оставляет за собой след пахнущего вещества (феромона). Такое вещество влияет на выбор маршрута: выбирается то направление движения, где уровень феромона больше. Самоорганизация муравьев обеспечивается взаимодействием следующих компонентов: случайность, многократность, положительная обратная связь, отрицательная обратная связь. Центральной идеей муравьиного алгоритма является накопление и использование статистических данных, собираемых искусственными муравьями [8].

Для решения задачи (1)-(3) в данной работе предлагается следующий алгоритм:

Шаг 1. Присвоить переменной record первоначальное (небольшое) значение;
Шаг 2. Присвоить переменной t значение 1;
Шаг 3. While $t \leq t_{\max}$ Do
Begin
Шаг 4. For i:=1 to m do
Begin
Шаг 5. Сформировать вектор P(n) вероятностей размещения i-го файла в j-й узел.
Шаг 6. Сгенерировать случайную величину g, распределенную по закону P(n);
Шаг 7. Если условие $j \leq g < j+1$ и условие (3) задачи выполняются, то назначить i-й файл в j-й узел, иначе сгенерировать новое значение случайной величины g.
End;
Шаг 8. Вычислить значение целевой функции по формуле (1) и присвоить его переменной c_f;
Шаг 9. If c_f > record then

Begin

Шаг 10. Присвоить переменной record значение переменной c_f и сохранить рекордное решение;
End;

Шаг 11. Присвоить переменной t значение t+1;
End;

Шаг 12. Вывести наилучшее решение и завершить алгоритм.

Пусть τ_j - количество феромона, накопленное очередным муравьем на маршруте j, ρ - коэффициент испарения феромона, $\rho = 0,01$, τ_0 - первоначальный уровень феромона, $\tau_0 = 0,01$, τ_j - уровень феромона на маршруте j. Вероятность размещения файла в узел j вычисляется по формуле (4):

$$P_j = \frac{\tau_j F_j}{\sum_{j=1}^n \tau_j F_j} \quad (4)$$

При выборе узла p для размещения файла происходит увеличение значения τ_p :

$\tau_p := (1 - \rho)\tau_p + \tau_p / ideal$, значение переменной ideal должно быть одного порядка с оптимальным значением целевой функции. При переходе к очередной итерации алгоритма уровень феромона накапливается:

$$\tau_j := (1 - \rho)\tau_j + \sum_{k=1}^n \tau_k / ideal.$$

Результаты эксперимента показаны на рис. 2. Получено строго оптимальное решение на 311 итерации.

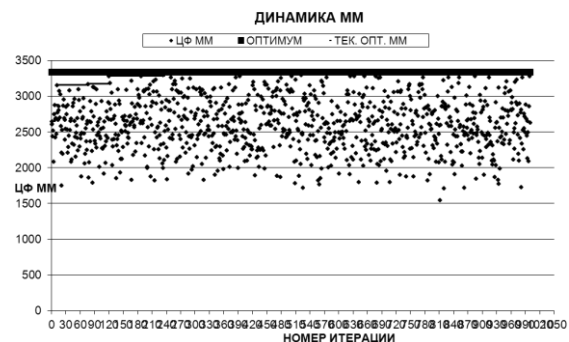


Рисунок 2 – Динамика поиска решений муравьиным методом

Генетический алгоритм

В данном разделе для решения задачи размещения файлов предлагается алгоритм, относящийся к классу генетических алгоритмов (ГА). Это стохастические эвристические оптимизационные методы. Они работают с совокупностью "особей" - популяцией, каждая из которых представляет возможное решение данной проблемы [9,10].

Для решения задачи (1)-(3) предлагается генетический алгоритм ГА, который выполняет поиск оптимального решения задачи с помощью постоянной эволюции предыдущего решения. При переходе от решения к решению применяются генетические операторы селекции, кроссинговера и мутации. В данной работе используется мультихромосомное представление решений задачи. Очередным решением является особь, мультихромосома которой показана в табл. 1. Она состоит из m хромосом и mn генов, где m – число файлов, n – число узлов, $X_{ij} \in \{0;1\}$

Таблица 1 - Мультихромосома особи

X11	X12	X13	X14	X15	...	X1n
X21	X22	X23	X24	X25	...	X2n
...	...	...	...	...	...	...
Xm1	Xm2	Xm3	Xm4	Xm5	...	Xmn

В ГА используются генетические операторы кроссинговера и мутации. Оператор кроссинговера производит обмен хромосомами между особями. Оператор мутации позволяет дать хромосомам-потомкам свойства, отсутствующие у родителей. Это позволяет алгоритму выходить из зоны локального экстремума. Выполнение кроссинговера показано в табл. 2а, 2б. Выполнение мутации показано в табл. 3.

Таблица 2а - Родительские особи (S и D)

S11	S12	S13	S14	S15	...	S1n
S21	S22	S23	S24	S25	...	S2n
S31	S32	S33	S34	S35	...	S3n
S41	S42	S43	S44	S45	...	S4n
...	...	...	...	...	...	...
Sm1	Sm2	Sm3	Sm4	Sm5	...	Smn
D11	D12	D13	D14	D15	...	D1n
D21	D22	D23	D24	D25	...	D2n
D31	D32	D33	D34	D35	...	D3n
D41	D42	D43	D44	D45	...	D4n
...	...	...	...	...	...	...
Dm1	Dm2	Dm3	Dm4	Dm5	...	Dmn

Таблица 2б – Результат кроссинговера

S11	S12	S13	S14	S15	...	S1n
D21	D22	D23	D24	D25	...	D2n
S31	S32	S33	S34	S35	...	S3n
S41	S42	S43	S44	S45	...	S4n
...	...	...	...	...	...	...
Sm1	Sm2	Sm3	Sm4	Sm5	...	Smn
D11	D12	D13	D14	D15	...	D1n
S21	S22	S23	S24	S25	...	S2n
D31	D32	D33	D34	D35	...	D3n
D41	D42	D43	D44	D45	...	D4n
...	...	...	...	...	...	...
Dm1	Dm2	Dm3	Dm4	Dm5	...	Dmn

Таблица 3 – Результат мутации

S11	S12	S13	S14	S15	...	S1n
S21	S22	S23	S24	S25	...	S2n
S41	S42	S43	S44	S45	...	S4n
S31	S32	S33	S34	S35	...	S3n
...	...	...	...	...	...	...
Sm1	Sm2	Sm3	Sm4	Sm5	...	Smn

В предлагаемом ГА оператор селекции имеет три разновидности: оператор селекции генов используется при формировании генов хромосом; оператор селекции хромосом используется при выборе хромосом для выполнения кроссинговера и мутации; оператор селекции особей используется при выборе родительских особей из популяции.

При формировании генов i -й хромосомы используется оператор селекции генов. Вероятность π_{ij} выбора j -го гена i -й хромосомы вычисляется, как отношение генной и хромосомной приспособленности по формуле (5):

$$\pi_{ij} = \frac{L_{ij}Z_{ij}}{\sum_{q=1}^n L_{iq}Z_{iq}} \quad (5)$$

При формировании каждого гена i -й хромосомы разыгрывается случайная величина, подчиняющаяся распределению π . Для этого используется следующая процедура:

Шаг 1. $\Psi_1 := \pi_1$
Шаг 2. For $q:=1$ to $n-1$ do
Шаг 3. Begin $\Psi_{q+1} := \pi_q + \pi_{q+1}$ End
Шаг 4. $\gamma := random$
Шаг 5. For $q:=1$ to $n-1$ do
Шаг 6. Begin If $\Psi_q \leq \gamma < \Psi_{q+1}$ then $X_{iq} := 1$ else $X_{iq} := 0$
Шаг 7. End

При выборе хромосом для выполнения кроссинговера и мутации используется оператор селекции хромосом. Вероятность Π_i выбора i -й хромосомы вычисляется, как отношение хромосомной и мультихромосомной приспособленности по формуле (6):

$$\Pi_i = \frac{\sum_{j=1}^n L_{ij}X_{ij}}{\sum_{r=1}^m \sum_{j=1}^n L_{rj}X_{rj}} \quad (6)$$

Пусть $Y_r = 1$, если выбрана хромосома r , иначе $Y_r = 0$. При формировании значений Y_r генерируются случайные величины, подчиняющиеся распределению Π . Для этого

используется процедура, аналогичная процедуре оператора селекции генов.

При выборе из популяции родительских особей используется оператор селекции особей. Вероятность P_k выбора особи на k -й итерации вычисляется, как отношение индивидуальной и наилучшей приспособленности. Пусть CF_k - значение целевой функции на k -й итерации, RCF - максимальное значение среди CF_s , где $s=1,2,...,k$. На каждой итерации целевая функция вычисляется по формуле: $CF = \sum_{i=1}^m \sum_{j=1}^n L_{ij} X_{ij}$.

Вероятность P_k вычисляется по формуле (7). Лучше приспособленные особи имеют большую вероятность стать родительскими.

$$P_k = CF_k / RCF_k \quad (7)$$

При выборе особи генерируется случайная величина, подчиняющаяся распределению P_k . Для этого используется такая же процедура, как и в операторе селекции генов.

Реализация ГА предусматривает выполнение следующих действий:

{Подготовительная часть}

{Формирование первоначальной особи}

Шаг 1. $k:=1$;

Шаг 2. Случайным образом сгенерировать мультихромосому особи $MH(k)$;

Шаг 3. Вычислить значение CF_k целевой функции особи $MH(k)$;

Шаг 4. $RCF := CF_k$;

{Основная часть}

{Цикл формирования популяции}

Шаг 5. While $k \leq k_{\max}$ Do

Begin

Шаг 6. Выполнить мутацию особи $MH(k)$ и получить особь $MH(k+1)$;

Шаг 7. Вычислить значение CF_{k+1} целевой функции особи $MH(k+1)$;

Шаг 8. If $CF_{k+1} > RCF$ then $RCF := CF_{k+1}$;

Шаг 9. $k:=k+1$;

Шаг 10. Выполнить кроссингвер особей $MH(k-1)$, $MH(k)$ и получить особи $MH(k+1)$ и $MH(k+2)$;

Шаг 11. Если условие (3) задачи выполняется, выполнить мутацию особей $MH(k+1)$ и $MH(k+2)$;

Шаг 12. Вычислить значение CF_{k+1} целевой функции особи $MH(k+1)$;

Шаг 13. If $CF_{k+1} > RCF$ then $RCF := CF_{k+1}$;

Шаг 14. Вычислить значение CF_{k+2} целевой функции особи $MH(k+2)$;

Шаг 15. If $CF_{k+2} > RCF$ then $RCF := CF_{k+2}$;

Шаг 16. $k:=k+2$;

Шаг 17. Выполнить оператор репродукции;

End;

Шаг 18. Вывести значения генов мультихромосомы с максимальной целевой функцией и завершить ГА.

Результаты эксперимента показаны на рис. 3. Получено строго оптимальное решение на 58 итерации.

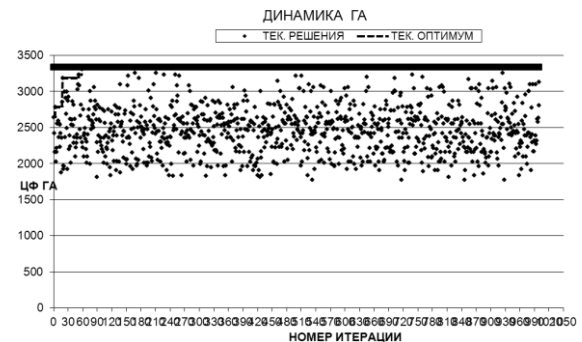


Рисунок 3 – Динамика поиска решений генетическим алгоритмом

Выводы

В работе предложены алгоритм имитации отжига, муравьиный и генетический алгоритмы решения задачи размещения файлов в компьютерных сетях. Динамика работы алгоритмов показана на рис. 4.



Рисунок 4 – Динамика поиска решений

Предложенные алгоритмы могут быть использованы для повышения эффективности компьютерной сети на этапах ее проектирования и эксплуатации. Критерием оптимальности размещения файлов является суммарный поток локальных запросов, инициированных в узлах сети в единицу времени. В ходе вычислительных экспериментов предлагаемыми в работе алгоритмами и методом полного перебора решена задача распределения m файлов среди n узлов сети, $m=8$, $n=3$. Для полного перебора требуется $n^m = 3^8 = 6561$ итерация, предложенными алгоритмами выполнено 1000

итераций. Получены следующие результаты. Строго оптимальное решение найдено алгоритмом имитации отжига на 984 итерации, муравьиным алгоритмом на 311 итерации, генетическим алгоритмом на 58 итерации.

Литература

1. Upadhyaya S., Lata S. Task allocation in Distributed computing VS distributed database systems: A Comparative study. // International Journal of Computer Science and Network Security, Vol. 8, N 3, 2008. – p. 338 – 346.
2. Singh A., Dutta K. Apply AHP for Resource Allocation Problem in Cloud. // International Journal of Computer Science and Network Security, Vol. 8, N 3, 2015. – p. 13 – 21.
3. Abdalla H.I. A New Data Re-Allocation Model for Distributed Database Systems. // International Journal of Database Theory and Application, Vol. 5, N 2, 2012. – p. 45 – 60.
4. Javed S., Bakhshi Z. H., Khalid M. M. Optimum allocation in Stratified Sampling with Random Costs. // International Review of Pure and Applied Mathematics, Vol. 5, N 2, 2009. – p. 363-370.
5. Khan M. F., Ali I., Ahmad Q.S. Chebyshev Approximate Solution to Allocation Problem in Multiple Objective Surveys with Random Costs.

- // American Journal of Computational Mathematics, N 1, 2011. – p. 247-251.
6. Kaur M., Singh S., Kaur R. Directory Structure and File Allocation Methods. // International Journal of Computer Science and Information Technologies, Vol. 7, N 2, 2016. – p. 577–582.
7. Зинкин С.А., Белецкий П.А. Оптимизация размещения данных по узлам информационно-вычислительной сети. В кн.: Современные тенденции технических наук: Материалы международной заочной научной конференции. – Уфа: Лето, 2013. – с. 29–31.
8. Курейчик В.В., Запорожец Д.Ю. Роевой алгоритм в задачах оптимизации. // Известия Южного федерального университета. Технические науки, Т. 108, № 7, 2010. – С. 28-32.
9. Abd El-Aziz M. H. Optimizing the Average Distributed Program Throughput (ADPT) by Using Genetic Algorithms. // International Journal of Intelligent Computing and Information Science, Vol. 10, N 1, 2010. –p.1-12.
10. Abd El-Aziz M. H., Younes A., Hassan M. R., Abdo H. Solving the file allocation problem in the distributed networks by using genetic algorithms. // International Journal of Computer Science and Network Security, Vol. 2, N 1, 2013. – p. 109 – 117.

Бельков Д.В. *Алгоритмы размещения файлов в компьютерных сетях.* В работе предлагаются новые методы для решения актуальной научной задачи оптимизации размещения файлов в компьютерных сетях (метод имитации отжига, муравьиный и генетический методы). Они могут быть использованы для повышения эффективности компьютерной сети на этапах ее проектирования и эксплуатации. Критерием оптимальности размещения файлов является суммарный поток локальных запросов, инициированных в узлах сети в единицу времени. С увеличением потока локальных запросов эффективность функционирования сети увеличивается за счет уменьшения времени отклика на запросы к файлам. В работе приведены результаты вычислительного эксперимента.

Ключевые слова: Файлы, узлы компьютерной сети, методы размещения файлов.

Belkov D.V. *File allocation algorithms in computer networks.* New algorithms for the decision of actual scientific task of optimization of file allocation in computer networks are offered in work. They can be used for the increase of efficiency of computer network on the stages of its design and exploitation. The total thread of the local queries initiated in the nodes of network in time unit is the criterion of optimum of file allocation. With the increase of thread of local queries is multiplied efficiency of functioning of network due to diminishment of response time upon requests to the files. The results of calculable experiment are presenting in work.

Keywords: Files, nodes of computer network, file allocation algorithms.

Статья поступила в редакцию 7.09.2017

Рекомендована к публикации доктором технических наук В.Н. Павлышом

Усовершенствование композиционного устройства управления с оптимальной адресацией микрокоманд

К.Н. Ефименко

Донецкий национальный технический университет
KN_Efimenko@mail.ru

Ефименко К.Н. Усовершенствование композиционного устройства управления с оптимальной адресацией микрокоманд. Предложен метод, позволяющий упростить схему адресации композиционного микропрограммного устройства управления с оптимальной адресацией микрокоманд, реализованную на FPGA. Метод основан на использовании процедуры элементаризации операторных линейных цепей с разделением кодов и оптимальным кодированием операторных линейных цепей. Применение данного метода обеспечивает уменьшение числа LUT-элементов в FPGA микросхеме. Приведен пример применения предложенного метода.

Ключевые слова: КМУУ с оптимальной адресацией микрокоманд, FPGA, оптимальное кодирование, логическая схема

Введение

Стремительное развитие компьютерной техники неуклонно ведет к ее внедрению практически во все сферы деятельности человека, что в свою очередь, предъявляет все более высокие требования к характеристикам проектируемых устройств. При этом основное внимание уделяется как увеличению быстродействия, так и снижению аппаратных затрат, что в конечном итоге влияет на снижение себестоимости изделий цифровой техники. Это относится как к универсальным, так и к специализированным вычислительным системам, при реализации которых широко используются ПЛИС типа «система-на-кристалле» (SoPC – system-on-a-programmable-chip) [1-4]. В состав SoPC входят средства для реализации произвольной логики (FPGA или CPLD), средства для реализации памяти, встроенные микропроцессоры, встроенные средства реконфигурации. Производительность таких ПЛИС определяется эффективностью взаимодействия всех встроенных компонентов [5].

Важнейшей составной частью любой цифровой системы является устройство управления [1,5], которое может быть реализовано как композиционное микропрограммное устройство управления (КМУУ) [6]. При реализации КМУУ в составе SoPC схема адресации микрокоманд строится на FPGA (field-programmable gate array) – программируемых пользователем матрицах вентилей, состоящих из миллионов элементов табличного типа (LUT-элементов) [7,8], а система микроопераций реализуются на встроенных блоках памяти (EMB, embedded

memory blocks). Ограниченное число входов LUT-элементов приводит к необходимости декомпозиции реализуемых булевых функций [8], что увеличивает число LUT-элементов (и их уровней) в схеме адресации КМУУ. В связи с этим актуальной остается задача разработки новых и совершенствования известных методов синтеза КМУУ.

В данной работе предлагается усовершенствование метода синтеза КМУУ с оптимальной адресацией микрокоманд [9], основанное на использовании процедуры элементаризации операторных линейных цепей (ОЛЦ) с разделением кодов.

Задачей исследования является разработка метода синтеза КМУУ с оптимальной адресацией микрокоманд, позволяющего уменьшить число LUT-элементов в комбинационной части УУ.

Общие теоретические положения

Пусть алгоритм управления цифровой системы задан в виде граф-схемы алгоритма (ГСА) Γ [6], содержащей начальную b_0 , конечную b_E , операторные и условные вершины. Операторные вершины образуют множество B_1 , имеющее M элементов. В вершинах $b_q \in B_1$ записываются микрокоманды $Y_q \in Y$, где $Y = \{y_1, \dots, y_N\}$ – множество микроопераций. В условных вершинах, образующих множество B_2 , записываются элементы множества логических условий $X = \{x_1, \dots, x_L\}$. Вершины ГСА образуют множество $B = B_1 \cup B_2 \cup \{b_0, b_E\}$, элементы которого связаны дугами из множества E .

Введем ряд определений [7], необходимых для дальнейшего изложения материала.

Определение 1. Операторной линейной цепью ГСА Γ называется конечная последовательность операторных вершин $\alpha_g = \langle b_{g1}, \dots, b_{gF_g} \rangle$, для любой пары соседних компонент которой существует дуга $\langle b_{gi}, b_{gi+1} \rangle \in E$, где $i=1, \dots, F_g-1$ – номер компоненты.

Определение 2. Вершина $b_q \in D^g$, где $D^g \subseteq B_1$ – множество вершин, входящих в ОЛЦ α_g , называется входом ОЛЦ α_g , если существует дуга $\langle b_t, b_q \rangle \in E$, где $b_t \notin D^g$.

Определение 3. Вершина $b_q \in D^g$, называется выходом ОЛЦ α_g , если существует дуга $\langle b_q, b_t \rangle \in E$, где $b_t \notin D^g$.

Определение 4. Вход ОЛЦ называется главным входом, если отсутствует связь этого входа с выходами операторных вершин.

Определение 5. Операторные линейные цепи $\alpha_i, \alpha_j \in C$ называются псевдоэквивалентными ОЛЦ, если их выходы связаны с входом одной и той

же вершины ГСА Γ , и образуют множество классов псевдоэквивалентных ОЛЦ $\Pi_C = \{B_1, \dots, B_I\}$. $I = |\Pi_C|$.

Пусть для ГСА Γ найдено разбиение $C = \{\alpha_1, \dots, \alpha_G\}$ множества B_1 на операторные линейные цепи и пусть для каждой пары соседних вершин ОЛЦ $\alpha_g \in C$ выполняется условие

$$A(b_{gi+1}) = A(b_{gi}) + 1 \quad (i=1, K, F_g - 1), \quad (1)$$

где $A(b_g)$ – адрес микрокоманды, соответствующей вершине $b_g \in B_1$.

В этом случае устройство управления цифровой системы может быть реализовано в виде КМУУ U_1 с оптимальной адресацией микрокоманд (рис. 1) [9], синтез которого основан на наличии в ГСА псевдоэквивалентных операторных линейных цепей (ПОЛЦ).

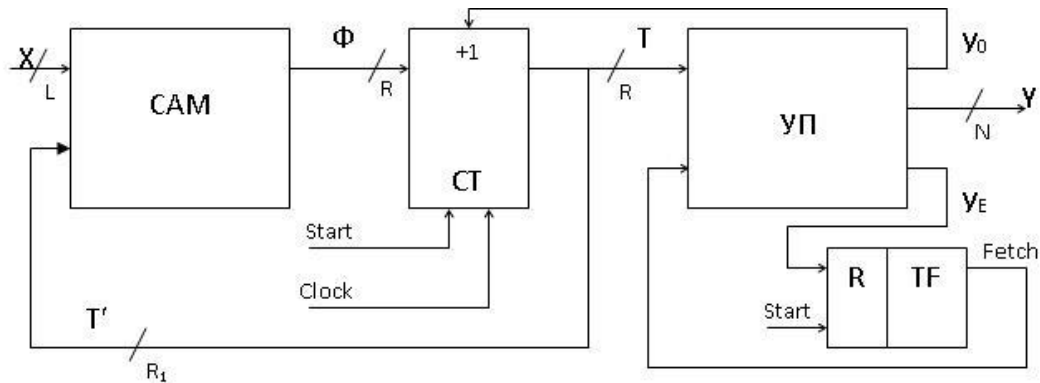


Рисунок 1 – Структурная схема КМУУ U_1

По сигналу Start в счетчик СТ заносится адрес первой микрокоманды интерпретируемого алгоритма, а триггер считывания TF устанавливается в единицу (Fetch=1), что разрешает выборку микрокоманд Y из управляющей памяти УП. Управляющая память УП хранит наборы микроопераций Y_q , где $Y_q \subseteq Y = \{y_1, \dots, y_N\}$ – микрооперации, записанные в вершине $b_q \in B_1$ ГСА Γ , и состоит из $2^R \cdot (N+2)$ бит. Первый дополнительный разряд используется для хранения сигнала y_0 , обеспечивающего естественную адресацию компонент ОЛЦ $\alpha_g \in C$. Второй – для организации режима останова КМУУ (сигнал y_E). Комбинационная схема адресации микрокоманд (CAM) благодаря оптимальной адресации микрокоманд, имеет $R_1 = \lceil \log_2 I \rceil \leq R$ сигналов обратной связи и реализует систему функций

$$\Phi = \Phi(T', X), \quad (2)$$

формирующих в счетчике СТ адрес $A(I_g^j)$ j -го входа ОЛЦ $\alpha_g \in C$.

Оптимальная адресация микрокоманд выполняется с помощью модифицированной

карты Карно. Модификация карты заключается в том, что по вертикали записываются двоичные наборы, следующие в естественном порядке. Карта Карно, содержащая адреса микрокоманд, предоставляет

$$\Delta = 2^R - M_1$$

клеток для адресации микрокоманд ОЛЦ $\alpha_g \in C'$, здесь $C' \subseteq C$ – множество ОЛЦ, выходы которых не связаны с входом вершины b_E ; M_1 – число компонент в ОЛЦ $\alpha_g \notin C'$. При выполнении условия

$$\Delta \geq \sum_{i=1}^I \Delta_i \quad (3)$$

адреса компонент любой ОЛЦ $\alpha_g \in C'$ могут быть расположены в соседних клетках карты Карно, причем компоненты всех ОЛЦ, принадлежащих i -му классу псевдоэквивалентных ОЛЦ B_i , будут расположены в одном кубе размерности Δ_i .

Предложенный метод оптимальной адресации микрокоманд [9] позволяет уменьшить число сигналов обратной связи в схеме адресации микрокоманд за счет уменьшения числа аргументов и числа термов в

системе функций (2). Однако его применение не всегда возможно. Для целесообразности использования метода необходимо выполнение ряда условий [9].

В настоящей работе предлагается для усовершенствования метода синтеза КМУУ с оптимальной адресацией микрокоманд использовать процедуру элементаризации ОЛЦ с разделением кодов.

Элементарной ОЛЦ (ЭОЛЦ) считается ОЛЦ, имеющая только один вход [6].

Основная идея метода

Структура КМУУ U_2 , интерпретирующего элементарные ОЛЦ с разделением кодов, представлена на рис. 2.

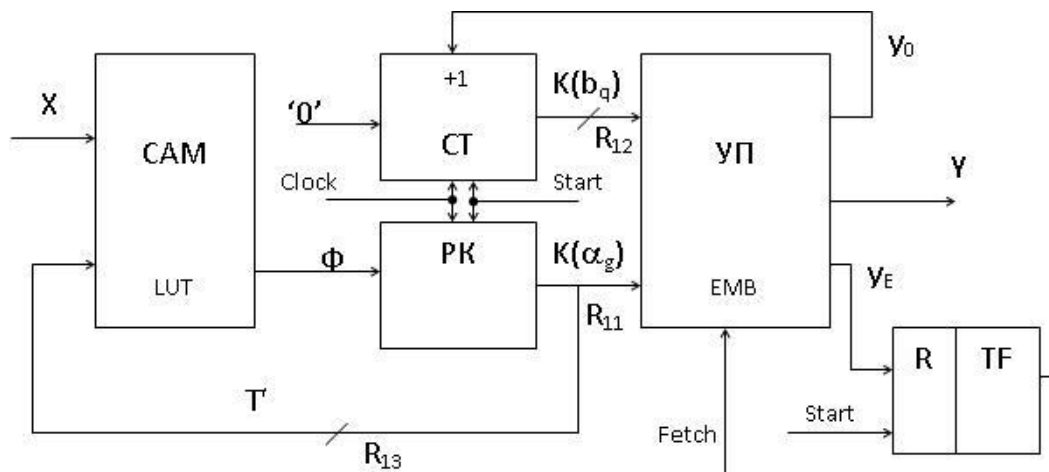


Рисунок 2 – Структурная схема КМУУ U_2

Здесь комбинационная схема САМ формирует систему функций

$$\Phi = \Phi(T', X), \quad (4)$$

задающих в регистре РК код $K(\alpha_g)$ текущей
ОЛЦ $\alpha_g \in C$.

Адрес микрокоманды представлен в виде конкатенации кодов [6]

$$A(Y_g) = K(\alpha_g) * K(b_g),$$

где $A(Y_q)$ – адрес микрокоманды Y_q , записанной в вершине b_q , входящей в ОЛЦ α_g и хранящейся в управляющей памяти УП;

$K(\alpha_g)$ – код ОЛЦ $\alpha_g \in C = \{\alpha_1, \dots, \alpha_G\}$
разрядности $R_{11} = \lceil \log_2 G \rceil$;

$K(b_q)$ – код вершины b_q кортежа $\alpha_g \in C$ разрядности

$$R_{12} = \lceil \log_2 F_{\max} \rceil, \quad (5)$$

где $F_{\max}$ – максимальное число компонент ОЛЦ в множестве S .

Разрядность адреса микрокоманды составляет $R = R_{11} + R_{12}$ и для его представления используется множество внутренних переменных

$$\mathbf{T} = \{T_1, \dots, T_{R_{11}}, T_{R_{11}+1}, \dots, T_{R_{11}+R_{12}}\}.$$

Использование элементарных ОЛЦ позволяет подавать на вход счетчика СТ нулевой код разрядности R_{12} , соответствующий коду первой компоненты текущей ОЛЦ $\alpha_g \in C$. При таком подходе коды ОЛЦ и коды их компонент являются взаимно независимыми и

на вход комбинационной схемы САМ подаются только R_{11} переменных обратной связи, образующих множество $T' = \{T_1, \dots, T_{R_{11}}\}$. Это позволяет уменьшить количество LUT-элементов, использующихся для реализации схемы САМ, по сравнению с КМУУ U_1 .

Функционирование КМУУ U_2 происходит следующим образом. По сигналу Start содержимое СТ и РК обнуляется, что является кодом первой компоненты первой ОЛЦ $\alpha_g \in C$, то есть адресу первой микрокоманды соответствующей ГСА Г. Триггер считывания ТФ устанавливается в единичное состояние и происходит считывание микрокоманды из УП.

Если адрес $A(Y_q)$ компоненты не является адресом выхода текущей ОЛЦ $\alpha_g \in C$, то одновременно с микрооперациями $Y_q \in Y$ формируется сигнал $y_0 = 1$. Таким образом, содержимое счетчика увеличивается на 1 и происходит переход к следующей компоненте Y_i текущей ОЛЦ $\alpha_g \in C$.

Если адрес $A(Y_q)$ компоненты является адресом выхода O_g текущей ОЛЦ, то сигнал $y_0=0$, счетчик обнуляется, и схема САМ формирует код $K(\alpha_q)$ следующей ОЛЦ $\alpha_q \in C$ в соответствии с системой (4). Для представления кода используются переменные $\varphi_i \in \Phi = \{\varphi_1, \dots, \varphi_{R_{11}}\}$. При формировании сигнала y_E триггер ТФ сбрасывается и функционирование КМУУ U_7 завершается.

В настоящей работе предлагается метод синтеза КМУУ U_2 .

Пример применения метода

Пусть алгоритм управления некоторой цифровой системы задан ГСА Γ_1 (рис. 3). Выполним синтез КМУУ U_2 по ГСА Γ_1 .

1. Формирование множества ОЛЦ C . Этот этап выполняется по известной методике [7] и в нашем случае имеем $C = \{\alpha_1, \dots, \alpha_5\}$, где $\alpha_1 = \langle b_1, b_2, b_3, b_4 \rangle$, $\alpha_2 = \langle b_5, b_6 \rangle$, $\alpha_3 = \langle b_7, b_8, b_9$,

$b_{10} \rangle$, $\alpha_4 = \langle b_{11}, b_{12} \rangle$, $\alpha_5 = \langle b_{13}, b_{14} \rangle$, $G = 5$.

2. Формирование множества элементарных ОЛЦ $C_E = \{\alpha_1, \dots, \alpha_{G_E}\}$.

Разбиение $\alpha_g \in C$ на элементарные ОЛЦ имеет смысл только в случае, если

$$|MI(\Gamma)| < |I(\Gamma)|, \quad (6)$$

где $MI(\Gamma)$ – множество главных входов ГСА Γ ; $I(\Gamma)$ – множество всех входов ГСА Γ . Для ГСА Γ_1 имеем $|MI(\Gamma_1)| = 5$ и $|I(\Gamma_1)| = 7$, следовательно, условие (6) выполняется.

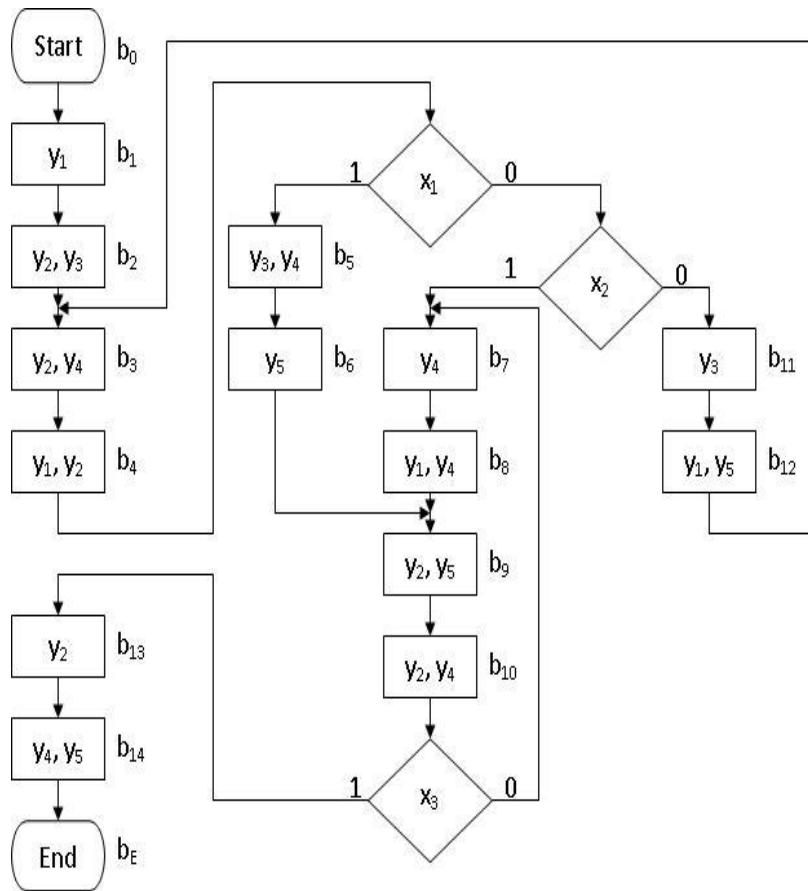


Рисунок 3 – Исходная граф-схема алгоритма Γ_1

Применение процедуры элементаризации [6] позволяет построить множество $C_E = \{\alpha_1, \dots, \alpha_7\}$ элементарных ОЛЦ, где $\alpha_1 = \langle b_1, b_2 \rangle$, $\alpha_2 = \langle b_3, b_4 \rangle$, $\alpha_3 = \langle b_5, b_6 \rangle$, $\alpha_4 = \langle b_7, b_8 \rangle$, $\alpha_5 = \langle b_9, b_{10} \rangle$, $\alpha_6 = \langle b_{11}, b_{12} \rangle$, $\alpha_7 = \langle b_{13}, b_{14} \rangle$, $G_E = 7$. Для кодирования элементарных ОЛЦ $\alpha_g \in C_E$ потребуется $R_{11}=3$ переменных $T_i \in T$.

В общем случае, число элементарных ОЛЦ больше числа исходных ОЛЦ, что увеличивает длину таблицы переходов КМУУ, но в свою очередь, позволяет уменьшить число переменных обратной связи на величину $\lceil \log_2(|I(\Gamma)| - |MI(\Gamma)|) \rceil$.

Формирование разбиения $\Pi_{C_E} = \{B_1, \dots, B_I\}$ множества ОЛЦ C_E на классы псевдоэквивалентных ОЛЦ.

Разбиение на классы псевдоэквивалентных ОЛЦ выполняется только для $\alpha_g \in C'_E$ таких, что выход этой ОЛЦ не связан с конечной вершиной ГСА. В нашем случае $C'_E = \{\alpha_1, \dots, \alpha_6\}$ и формирование разбиения множества ОЛЦ C'_E на классы псевдоэквивалентных ОЛЦ дает $\Pi'_{C'_E} = \{B_1, \dots, B_4\}$, где $B_1 = \{\alpha_1, \alpha_6\}$, $B_2 = \{\alpha_2\}$, $B_3 = \{\alpha_3, \alpha_4\}$, $B_4 = \{\alpha_5\}$, $I=4$.

4. Оптимальное кодирование

элементарных ОЛЦ. Оптимальное кодирование элементарных ОЛЦ $\alpha_g \in C'_E$ выполняется аналогично кодированию псевдоэквивалентных состояний автомата Мура [8]. При этом каждой элементарной ОЛЦ $\alpha_g \in C'_E$ ставится в соответствие R_{11} -разрядный код и все ОЛЦ $\alpha_g \in B_i$ должны располагаться в одном кубе булевого пространства. Вариант оптимального кодирования приведен на рис. 4, представляющем собой модифицированную карту Карно.

$T_2 T_3$	T_1	
	0	1
00	α_1	α_3
01	α_6	α_4
10	α_2	α_5
11	*	*

Рисунок 4 – Оптимальное кодирование ОЛЦ КМУУ $U_2(\Gamma_1)$

Как видно из рис. 4, полученные кубы булевого пространства дают коды, однозначно идентифицирующие классы $B_i \in \Pi_{C_E}$:

$K(B_1) = 00^*$,
 $K(B_2) = 01^*$,
 $K(B_3) = 10^*$,
 $K(B_4) = 11^*$.

Это приводит к уменьшению числа переменных обратной связи $T_r \in T'$ от R_{11} до $R_{13} = \lfloor \log_2 I \rfloor = 2$.

5. Формирование содержимого управляющей памяти УП. Результаты кодирования элементарных ОЛЦ приведены в табл. 1, коды ОЛЦ $\alpha_g \in C'_E$ взяты из карты Карно на рис. 4. Для кодирования ОЛЦ $\alpha_7 \notin C'_E$ взят один из несущественных кодов.

Таблица 1 - Кодирование элементарных ОЛЦ ГСА Γ_1

α_q	α_1	α_2	α_3	α_4	α_5	α_6	α_7
$K(\alpha_g)$	00	01	10	10	11	00	01
α_g	0	0	0	1	0	1	1

Согласно (5) имеем $F_{\max} = 2$, то есть для кодирования компонент ОЛЦ $\alpha_g \in C_E$ необходимо $R_{12} = 1$ переменных $T_r \in T$. При кодировании первой компоненты ОЛЦ $\alpha_g \in C_E$ переменная $T_4 = 0$, второй компоненты – $T_4 = 1$.

Таким образом, для адресации микрокоманд КМУУ $U_2(\Gamma_1)$ необходимо использовать $R = R_{11} + R_{12} = 3 + 1 = 4$ переменных, что не превышает

соответствующий показатель для КМУУ $U_1(\Gamma_1)$. Однако, число переменных обратной связи $R_{13} = 2$ в КМУУ $U_2(\Gamma_1)$, меньше, чем $R_1 = 4$ в КМУУ $U_1(\Gamma_1)$. Следовательно, применение КМУУ U_2 , интерпретирующего элементарные ОЛЦ с разделением кодов, целесообразно.

Фрагмент содержимого управляющей памяти УП для КМУУ $U_2(\Gamma_1)$ приведен в табл. 2. Здесь сигнал y_0 введен во все вершины, не являющиеся выходами ОЛЦ $\alpha_g \in C_E$, сигнал y_E введен в вершину b_{14} , связанную с конечной вершиной b_E .

Таблица 2 - Фрагмент содержимого управляющей памяти КМУУ $U_2(\Gamma_1)$

Адрес $A(b_q)$	Микрокоманда $Y(b_q)$							b_q
	y_0	y_1	y_2	y_3	y_4	y_5	y_E	
0000	1	1	0	0	0	0	0	b_1
0001	0	0	1	1	0	0	0	b_2
0100	1	0	1	0	1	0	0	b_3
...								
0111	0	0	0	1	0	1	1	b_{14}

6. Формирование таблицы переходов КМУУ. До построения таблицы переходов КМУУ $U_2(\Gamma_1)$ выполняется формирование системы формул перехода, в которых выходы ОЛЦ $\alpha_g \in B_i$ заменяются соответствующими классами $B_i \in \Pi_{C_E}$, а входы ОЛЦ заменяются ОЛЦ $\alpha_q \in C_E$, в которые происходит переход. В нашем случае имеем

$$\begin{aligned} B_1 &\rightarrow \alpha_2; \\ B_2 &\rightarrow x_1 \alpha_3 \vee \overline{x_1} x_2 \alpha_4 \vee \overline{x_1} \overline{x_2} \alpha_6; \\ B_3 &\rightarrow \alpha_5; \\ B_4 &\rightarrow x_3 \alpha_7 \vee \overline{x_3} \alpha_4. \end{aligned} \quad (7)$$

При этом не используется формула для класса $B_5 \in \Pi_{C_E}$, в который входит ОЛЦ $\alpha_7 \in C_E$, содержащая вершину b_{14} , связанную с конечной вершиной b_E ГСА Γ_1 . Таблица переходов КМУУ $U_2(\Gamma_1)$ строится по системе (7).

7. Формирование системы функций Ф. Формирование системы функций (4) выполняется по таблице переходов КМУУ $U_2(\Gamma_1)$ (табл. 3) в виде:

$$\varphi_r = \bigvee_{h=1}^N C_{rh} E_i^h X_h \quad (r=1, R_{13}),$$

где C_{rh} – булева переменная, равная единице, если и только если в h -й строке таблицы переходов записана функция $\varphi_r = 1$;

E_i^h – конъюнкция переменных $T_r \in T$, соответствующая коду $K(B_i)$ из h -й строки таблицы переходов $h=1, N$;

$$E_i^h = \bigwedge_{r=1}^{R_{13}} T_r^{h_r} \quad (i=1, I),$$

где $l_{ir} \in \{*, 0, 1\}$ – значение r -го разряда кода $K(B_i)$, $T_r^* = 1$, $T_r^0 = \overline{T_r}$, $T_r^1 = T_r$ ($r=1, R_{13}$).

Таблица 3 -Таблица переходов

КМУУ U_2 (Γ_1)						
B_i	$K(B_i)$	α_q	$K(\alpha_q)$	X_h	Φ_h	h
B_1	00^*	α_2	010	–	D_2	1
B_2	01^*	α_3	100	x_1	D_1	2
		α_4	101	$\overline{x_1} x_2$	$D_1 D_3$	3
		α_6	001	$\overline{x_1} \overline{x_2}$	D_3	4
B_3	10^*	α_5	110	–	$D_1 D_2$	5
B_4	11^*	α_7	011	x_3	$D_2 D_3$	6
		α_4	101	$\overline{x_3}$	$D_1 D_3$	7

Для КМУУ U_2 (Γ_1) имеем, например,

$$D_2 = \overline{T_1} T_2 \vee T_1 \overline{T_2} \vee T_1 T_2 x_3.$$

8. Синтез логической схемы КМУУ на FPGA. Синтез логической схемы выполняется на LUT-элементах (регистр РК реализуется на входящих в состав FPGA триггерах) и управляющей памяти УП на блоках ЕМВ [7,8].

Заключение

Предложенный в работе метод элементаризации ОЛЦ с разделением кодов и оптимальным кодированием ОЛЦ позволяет при выполнении условия

$$R_{13} < R_{11} \quad (8)$$

уменьшить число переменных обратной связи, необходимых для реализации системы функций (4), формирующих код следующей ОЛЦ ГСА. Применение метода не требует использования дополнительного регистра, так как для его реализации достаточно использовать имеющиеся в составе FPGA триггеры. В тоже время, происходит уменьшение разрядности счетчика СТ, который используется только для адресации компонент ОЛЦ (5). При этом время такта КМУУ U_1 и U_2 совпадают.

Проведенные авторами исследования на основе стандартных примеров МПА [10] показали, что использование КМУУ U_2 уменьшает число LUT-элементов FPGA, необходимых для реализации схемы адресации микрокоманд, до 20% по сравнению с КМУУ U_1 .

Научная новизна предложенного метода заключается в использовании особенностей базиса FPGA и применении элементаризации ОЛЦ с разделением кодов и оптимального кодирования ОЛЦ для уменьшения числа LUT элементов в схеме КМУУ.

С практической точки зрения метод дает возможность уменьшить площадь кристалла

FPGA, занимаемую схемой КМУУ с оптимальной адресацией микрокоманд, что позволяет получать схемы, обладающие меньшей стоимостью, чем известные из литературы аналоги.

Дальнейшие исследования могут быть направлены на разработку метода синтеза КМУУ, уменьшающего число LUT-элементов в схеме адресации при нарушении условия (8).

Литература

1. Baranov, S. Logic and System Design of Digital Systems/ S. Baranov. – Tallinn: TUT Press, 2008. – 266 pp.
2. Соловьев, В.В. Логическое проектирование цифровых систем на основе программируемых логических интегральных схем / В.В. Соловьев, А.С. Климович. – Москва: Горячая линия – Телеком, 2008. – 376 с.
3. Minks, P. FSM – based Digital Design using Verilog / P. Minks, I. Eliot. – London: John Wiley & Sons, 2008. – 351 pp.
4. Maxfield, S. The Design Warrior's Guide to FPGAs/ S. Maxfield. – Amsterdam: Elsevier, 2004. – 541 pp.
5. Палагин, А.В. Реконфигурируемые вычислительные системы: Основы и приложения / А.В. Палагин, В.Н. Опанасенко. – Киев: Просвита, 2006. – 280 с.
6. Barkalov, A. Logic synthesis for compositional microprogram control units/ A. Barkalov, L. Titarenko. – Berlin: Springer, 2008. – 272 pp.
7. Баркалов, А.А. Синтез композиционных микропрограммных устройств управления / А.А. Баркалов, Л.А. Титаренко. – Харьков: Коллегиум, 2007. – 302с.
8. Баркалов, А.А. Синтез микропрограммных автоматов на заказных и программируемых СБИС / А.А. Баркалов, Л.А. Титаренко. – Донецк: УНИТЕХ, 2009. – 336 с.
9. Баркалов, А.А. Оптимизация числа LUT-элементов в композиционном микропрограммном устройстве управления с общей памятью / А.А. Баркалов, С.А. Ковалев, К.Н. Ефименко // Управляющие системы и машины. – 2006. – №4. – С. 50-55.
10. Yang, S. Logic synthesis and optimization bench marks user guide. Technical report/ S. Yang. – Microelectronic Center of North Carolina, 1991. – 44 pp.

Efimenko K.N. Optimization of the compositional control unit with optimum addressing microinstructions. The method is proposed that leads to simplification of the circuit of addressing of compositional microprogram control unit with optimum addressing microinstructions implemented on FPGA. Method is based on usage of procedure elementarization of operational linear chains with sharing of codes and optimal coding of operational linear chains. The application of the method permits to decrease the amount of LUT-elements in the chip of FPGA. An example of proposed method application is given.

Keywords: CMCU with optimum addressing microinstructions, FPGA, optimal coding, logic circuit

Єфіменко К.М. Удосконалення композиційного пристрою керування з оптимальною адресацією мікрокоманд. Запропоновано метод, що дозволяє спростити схему адресації композиційного мікропрограмного пристрою керування з оптимальною адресацією мікрокоманд, яка реалізована на FPGA. Метод заснований на використанні процедури елементаризації операторних лінійних ланцюгів з розділенням кодів і оптимальним кодуванням операторних лінійних ланцюгів. Застосування даного методу забезпечує зменшення числа LUT-елементів в FPGA мікросхемі. Наведено приклад застосування запропонованого методу.

Ключові слова: КМПК з оптимальною адресацією мікрокоманд, FPGA, оптимальне кодування, логічна схема

Ефименко К.Н. Усовершенствование композиционного устройства управления с оптимальной адресацией микрокоманд. Предложен метод, позволяющий упростить схему адресации композиционного микропрограммного устройства управления с оптимальной адресацией микрокоманд, реализованную на FPGA. Метод основан на использовании процедуры элементаризации операторных линейных цепей с разделением кодов и оптимальным кодированием операторных линейных цепей. Применение данного метода обеспечивает уменьшение числа LUT-элементов в FPGA микросхеме. Приведен пример применения предложенного метода.

Ключевые слова: КМУУ с оптимальной адресацией микрокоманд, FPGA, оптимальное кодирование, логическая схема

Статья поступила в редакцию 17.10.2017
Рекомендована к публикации доктором технических наук В.Н. Павлышом

УДК 004.05

Модельный базис оценки риска взаимодействия в архитектуре партнерской сети

Казакова Е.И., Коломыцева А.О.

ГОУВПО «Донецкий национальный технический университет»

Казакова Е.И., Коломыцева А.О., Модельный базис оценки риска взаимодействия в архитектуре партнерской сети. В статье дана авторская трактовка понятия «риск и надежность взаимодействия», проведен критический анализ методов управления риском и всех форм его проявления в каналах взаимодействия партнерской предпринимательской сети, как сложной системы взаимодействия. Предложена стохастическая модель оценки риска взаимодействия и даны рекомендации по использованию результатов моделирования для проектирования архитектуры и каналов взаимодействия партнерской сети.

Ключевые слова: риск взаимодействия, архитектура сложной системы взаимодействия, партнерская сеть, экономика взаимодействий.

Сегодня формирование методологии сетевых форм организации бизнеса, как сложных форм организационных систем, и партнерских сетей в частности, развивается среди целого ряда известных научных школ и направлений [1-3]. Большинство работ рассматривают институциональные аспекты и посвящены формированию и развитию новых форм организации сетевых бизнес-систем, а так же вопросам классификации сетевых форм организации бизнеса. Выделены «чистые типы организации сетей», которые выступая эталоном для выработки последующих рекомендаций, позволяют учесть необходимость адаптации к условиям среды в которой они формируются и развиваются, а так же специфику правовых и отраслевых особенностей их последующего функционирования.

В мире стремительно происходит диффузия информации, технологий и знаний, и сегодня вместо жестко ограниченных друг от друга систем возникают конгломеративные формы предприятий исходя из необходимости их сближения и взаимодействия. Возникают системы нового типа, в которых успешная деятельность экономических агентов опирается не на задачу единоличного выживания в конкурентной борьбе, а на обладание более устойчивыми формами совместного существования в среде, где все взаимодействуют.

Так углубление специализации компании в условиях растущей взаимозависимости сфер деятельности и склонность скорее выбрать «альтернативу взаимного проигрыша» в оценке оптимального соотношения суммарных выгод и издержек в корне изменили традиционный подход к конкуренции и приводят к расширению стратегических опций, а так же к зарождению новых, более гибких форм взаимодействия.

В этой связи в течение последних десятилетий в мировой экономике наблюдается бурный рост сетевых компаний, принимающих

различные организационные формы и форматы сетевого взаимодействия. Появились и широко используются в практике бизнеса такие понятия, как ситуация взаимного выигрыша или конкурентного сотрудничества (coopetition — словообразование из англ. cooperation — «сотрудничество» и competition — «соревнование») [4]. Кооперативное поведение участников рынка стало проявляется в виде установления устойчивых вертикальных межфирменных взаимодействий (между членами цепочки создания ценности), а также развития горизонтальных связей (между конкурентами).

Проведенный анализ работ в данной области исследований показал [2-4], что наиболее неизученным остается вопрос анализа рисков, которые возникая на всех стадиях организации сложных сетевых форм организации бизнеса и партнерских отношений – от появления, становления до преобразования в новые организационные формы сетевого взаимодействия фактически влияют, как на эффективность и результативность взаимодействий, так и зачастую устанавливают приоритеты и правила реализации основных управленческих функций и задач каждой сетевой единицы. Результаты более ранних исследований [5-7], как правило ориентировались на изучение приоритетов взаимодействия независимых участников рынка, которые стремились снизить избыточную конкурентную активность и координировать стратегию на достижение согласованных совместных целей. Развитие информационной и сетевой экономик привело к трансформации большинства подходов управлению экономическими отношениями и сместило парадигму оптимального управления процессами в сторону оптимизации связей между элементами, так как объекты управления более стремительно могут менять свои формы и пространственное расположение. Это привело к появлению нового типа организации

экономических отношений – экономики взаимодействия. Для нее более актуальными становятся вопросы пересмотра существующих подходов и моделей анализа эффективности функционирования экономических объектов, особенно в части определения интенсивности и динамики процессов развивающихся одновременно в нескольких средах «Пространство-Время-Результативность».

Именно скорость обработки информации, пространственная глобализация рыночных взаимодействий, а так же смещение вектора эффективности от конкуренции к кооперации создают основу возникновения качественно нового критерия управления – обеспечение эффективности взаимодействия в архитектуре сетевой бизнес-системы, как в сложной иерархической системе функционирующей в условиях неопределенности и риска[8].

В таблице 1 приведена классификация для категории риска взаимодействия, которая учитывает принципиальные особенности и условия организации экономических отношений для условий экономики взаимодействий.

Таблица 1 - Параметры классификации риска взаимодействия

Классификационные признаки риска взаимодействия	Виды и формы проявления риска взаимодействия
По уровню оценки	Низкий; Значительный; Критический;
По степени проявления в динамике	Допустимый; Умеренный; Недопустимый;
По сфере возникновения	Внешний (не консолидированный по собственности); Внутренний (модульный, элементарный); Смешанный (трансформирующийся);
Для форм организации партнёрских каналов	В каналах вертикальной межфирменной интеграции; В каналах горизонтальной интеграции; В универсальных (конгломеративных) каналах;
По степени устойчивости в динамике	В устойчивых каналах; В неустойчивых каналах;
По характеру связей между элементами и уровнями в архитектуре сетевой бизнес-системы	Балансирующий связь; Дисбалансирующий связь;

Обоснование критерия управления взаимодействием необходимо начать с формального описания реального процесса возникновения риска в условиях взаимодействия, которое осуществляется между несколькими элементами партнерской сети и осуществляется в виде случайного процесса.

Математическая модель представляет собой оператор, преобразующий функцию оценки риска взаимодействия до вступления в сеть $n(D_{\text{вз}})$ в функцию $n(B_{\text{вз}})$ - риск взаимодействия после вступления в сеть. Параметры этого оператора зависят от остальных величин, которые являются управляющими параметрами в архитектуре сетевой бизнес-системы данного вида (общий эффект (прибыль) сети, вклад каждого элемента сетевой бизнес системы в общий эффект взаимодействия, коэффициент устойчивости основных показателей эффекта взаимодействия в динамике и т.д.).

Получение математической модели целесообразно осуществлять в два этапа: на первом этапе найти вид оператора, преобразующего функцию $n(D_{\text{вз}})$, описывающую архитектуру каналов взаимодействия и основные формы организации в сети, в функцию $n(B_{\text{вз}})$, которая характеризует результат-эффект взаимодействия в целом и для каждого элемента отдельно, а на втором этапе – найти зависимость параметров оператора Φ от остальных величин и в первую очередь риска взаимодействия, с учетом выбранной формы организации сетевого взаимодействия и управляющих воздействий в сетевой бизнес-системе.

Первому этапу соответствует структурная схема преобразования для оператора:

$$f(x) \rightarrow \Phi(x, y) \rightarrow f(y).$$

Оператор связи определяется по известным законам изменения входных и выходных функций, полученных в результате эксперимента. Связь между входом и выходом можно представить в виде уравнения

$$f(y) = \int_{x_{\min}}^{x_{\max}} \Phi(x, y) f(x) dx \quad (1)$$

Для обоснования этого выражения рассмотрим физическую сущность искомого оператора связи $\Phi(x, y)$. Пусть имеется некоторый однородный массив (портфель взаимодействий) объемом V_1 , для которого справедливо условие

$$V(x) = V(x_1) = \text{const}, \quad f(x_i) = 0 \text{ при } i = 1. \quad (2)$$

В результате выбора сетевой формы организации партнерских отношений распределение риска по каналам горизонтальной межфирменной интеграции будет описываться

некоторой функцией, для которой, в свою очередь, будет соблюдаться условие

$$\int_{D_{\min}}^{D_{\max}} V_1(D_{\text{вз}}) dD_{\text{вз}} = V(x_1). \quad (3)$$

Для архитектуры сети взаимодействия необходимость учитывать вероятность возникновения внешнего риска взаимодействия, также однородного, поэтому будет получена другая функция распределения риска по каналам вертикальной межфирменной интеграции $V_2(D)$

$$\int_{D_{\min}}^{D_{\max}} V_2(D_{\text{вз}}) dD_{\text{вз}} = V(x_2). \quad (4)$$

Продолжая рассмотрение всех возможных значений в портфеле взаимодействий, определим результирующий закон распределения вероятности риска взаимодействия как сумму функций $V_1(D), V_2(D), \dots, V_k(D)$:

$$\int_{D_{\min}}^{D_{\max}} V_1(D_{\text{вз}}) dD_{\text{вз}} = V(x_1), \quad V(D_{\text{вз}}) = \sum_{i=1}^k V_i(D_{\text{вз}}), \quad (5)$$

где $\sum_{i=1}^k V(x_i) = V$ - набор возможных форм организации взаимодействия в каналах сетевой бизнес-системы (портфель взаимодействий).

Разделив и умножив правую часть выражения (5) на $V(x_i)$, получим

$$V(D) = \sum_{i=1}^k \Phi(x_i, D_{\text{вз}}) V(x_i). \quad (6)$$

Переходя к пределу при $i \rightarrow \infty$, а $\Delta x = x_i - x_{i-1}$ к нулю, имеем:

$$V(D) = \int_{x_{\min}}^{x_{\max}} \Phi(x, D_{\text{вз}}) V(x) dx \quad (7)$$

что совпадает с выражением (1). Следовательно, нахождение математической модели в таком виде отвечает экономической сущности и вероятностной природе риска в оценке эффектов взаимодействия.

Установленные закономерности процессов взаимодействия, характера влияния возмущений внешней и внутренней среды, условия выбора форм и требований к организации обмена в каналах взаимодействия обнаруживают общую тенденцию в функционировании партнерских систем, как сложных вероятностных систем. Общность закономерностей процессов взаимодействия позволяет разработать стохастическую (вероятностную) модель их функционирования [9].

Функционирование сетевой бизнес-системы в динамике представляет собой последовательную смену его состояний во времени, обусловленную формой организации взаимодействия в каналах и вероятностью потери баланса взаимодействия, которые выражаются устойчивым снижением

запланированных эффектов взаимодействия. Источником таких возмущений, с одной стороны, является внешняя среда, с другой – внутренние условия и особенности построения отношений в каналах взаимодействия. Представим всю совокупность действующих на процесс факторов в качестве его входов. Тогда выходами будут его состояния из множества $\{E_n\}$. Выделим из множества всевозможных состояний, в которых может находиться процесс, систему s несовместимых классов $\{E_s\}, s = 1, 2, \dots, n$. Основу такого разбиения в каждом конкретном случае определяют требования к задачам.

Для непосредственного подтверждения приемлемости модели марковской цепи при описании поведения процесса взаимодействия в сетях воспользуемся экспериментальными данными, и покажем на основании этих данных, что процесс взаимодействия до и после вступления в партнерскую сеть E_i в E_j является стационарным и эргодическим.

С целью установления стационарности процесса взаимодействия в сети, найдем матрицу вероятностей переходов процесса взаимодействия в предлагаемой архитектуре сетевой бизнес-системы при шаге $\Delta t = 1$ мес. и определим ее неподвижный вектор $\|t\|$.

Подсчитав вероятность появления риска взаимодействия для пар состояния $P(E_i E_j)$, вычислим вероятности появления каждого состояния $P(E_i)$ и условные вероятности появления состояния (риск взаимодействия после вступления в партнерскую сеть) E_j , если предшествующим было состояние E_i (до вступления в сеть), по формулам:

$$P(E_i) = \sum_j P(E_i E_j) \quad (8)$$

$$P(E_j / E_i) = \frac{P(E_i E_j)}{P(E_i)} = P_{ij} \quad (9)$$

В результате вычислений получаем матрицу вероятностей переходов:

$$\|P\| = \begin{pmatrix} 0.979 & 0.016 & 0.003 & 0.002 \\ 0.050 & 0.919 & 0.026 & 0.005 \\ 0.030 & 0.070 & 0.959 & 0.004 \\ 0.007 & 0.002 & 0.003 & 0.988 \end{pmatrix} \quad (10)$$

Найдем неподвижный вектор $\|t\|$ для полученной матрицы вероятностей переходов из уравнения (8)

$$\|t\| \cdot \begin{pmatrix} 0.979 & 0.016 & 0.003 & 0.002 \\ 0.050 & 0.919 & 0.026 & 0.005 \\ 0.030 & 0.070 & 0.959 & 0.004 \\ 0.007 & 0.002 & 0.003 & 0.988 \end{pmatrix} = \|t\| \quad (11)$$

Учитывая, что $\sum_{i=1}^n t_i = 1$, получаем следующую

систему уравнений для компонент вектора $\|t\|$:

$$\begin{cases} 0.979t_1 + 0.016t_2 + 0.003t_3 + 0.002t_4 = t_1 \\ 0.050t_1 + 0.919t_2 + 0.026t_3 + 0.005t_4 = t_2 \\ 0.030t_1 + 0.070t_2 + 0.959t_3 + 0.004t_4 = t_3 \\ 0.007t_1 + 0.002t_2 + 0.003t_3 + 0.988t_4 = t_4 \\ t_1 + t_2 + t_3 + t_4 = 1 \end{cases} \quad (12)$$

которая имеет единственное решение:

$$t_1 = 0,552; \quad t_2 = 0,125; \quad t_3 = 0,134; \quad t_4 = 0,189,$$

т.е. в пределе

$$\lim_{n \rightarrow \infty} P^{(n)} = \begin{pmatrix} 0.552 & 0.125 & 0.134 & 0.189 \end{pmatrix} \quad (13)$$

Соответствующий матрице (10) граф состояния изображен на рис. 1, где стрелки указывают направления возможных переходов из состояния E_i в состояние E_j , а цифры соответствуют вероятностям этих переходов.

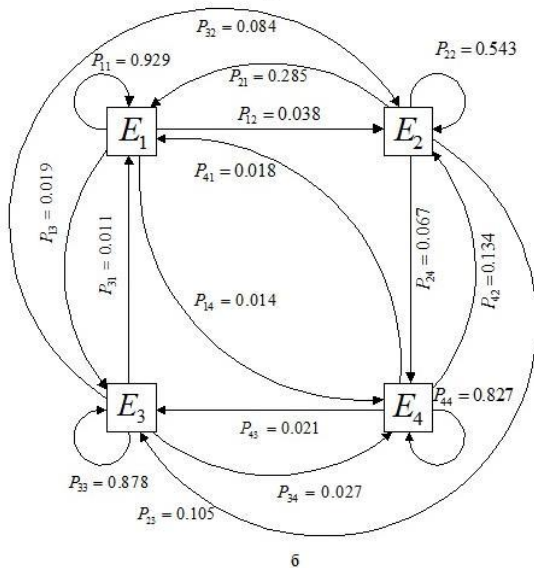


Рисунок 1 - Граф смены состояний процесса взаимодействия в сетевых бизнес-системах

Таким образом, рассматриваемые процессы являются процессами стационарными и разработанная математическая модель оценки рисков взаимодействия адекватна, имеет хорошую сходимость с экспериментальными данными, что позволит основательно подойти к выбору наиболее эффективной формы организации процессов взаимодействия в партнерских сетях, а так же и гарантировать минимальный уровень проявления риска взаимодействия в динамике. Так же для условий экономики взаимодействия в работе предлагается ввести понятие «риск взаимодействия», «надежность взаимодействия» и

«портфель взаимодействий» и изложить их следующей трактовке.

Риск взаимодействия определяется как вероятность потери баланса взаимодействия в архитектуре сетевой бизнес-системы, а также утрата возможности получения качественного и количественного эффекта от реализации равновесных партнерских отношений в динамике. Обратной величиной риска взаимодействия предложено считать *надежность взаимодействия*, которая определена как вероятность устойчивого достижения баланса взаимодействия, который обеспечивает эффект последовательной реализации партнерских отношений в динамике для данной архитектуры сетевой бизнес-системы. При этом *портфелем взаимодействий* считать набор возможных вариантов организации сетевых партнерских отношений для которых гарантированы как условия обеспечения сбалансированности связей между участниками взаимодействия так и устойчивость показателей суммарного эффекта взаимодействия в динамике.

Литература

1. Burt R. Models of Network Structure // Annual Review of Sociology. 1980. Vol. 6.
2. Camarinha-Matos L.M., Afsarmanesh F.F. Collaborative Networks: A New Scientific Discipline // Journal of Intelligent Manufacturing. 2005. \ Vol. 16.
3. Contractor F. Contractual and Cooperative Forms of International Business: Towards a Unified Theory of Modal Choice // Management International Review 1990. Vol. 30. No. 1.
4. Методология исследования сетевых форм организации бизнеса [Текст]: коллект. моногр. / М. А. Бек, Н. Н. Бек, Е. В. Бузулукова и др.; под науч. ред. М. Ю. Шерешевой; Нац. исслед. ун-т «Высшая школа экономики». — М.: Изд. дом Высшей школы экономики, 2014. — 446, [2] с.
5. Williamson O. Comparative Economic Organization: The Analysis of Discrete Structural Alternatives // Administrative Science Quarterly. 1991. Vol 36. No. 2.
6. Fernandez A., Arrunada B., Gonzalez M. Quasi-Integration in Less-Than-Truckload Trucking // Institutions, Contracts and Organizations. Perspectives from New Institutional Economics, /ed. by C. Menard. Cheltenham, UK: Edward Edgar, 2000. P. 294–303.
7. Forrester J. W. New Corporate Design // Sloan Management Review 1965. Vol. 7. No. 1.
8. Miles R.F., Snow C.C. Network Organizations: New Concepts for New Forms // California Management Review 1986. Vol. 28. No. 3.
9. Клейнер Г.Б., Тамбовцев В.Л., Качалов Р.М. Предприятие в нестабильной экономической среде: риски, стратегии, безопасность. М.: Экономика, 1997.

10. Teece D.J. Profiting from Technological Innovation: Implications for Integration, Collaboration, Licensing and Public Policy // Research Policy. 1986. Vol. 15. No. 6.

Казакова Е.И., Коломыцева А.О., Модельный базис оценки риска взаимодействия в архитектуре партнерской сети. В статье дана авторская трактовка понятия «риск и надежность взаимодействия», проведен критический анализ методов управления риском и всех форм его проявления в каналах взаимодействия партнерской предпринимательской сети, как сложной системы взаимодействия. Предложена стохастическая модель оценки риска взаимодействия и даны рекомендации по использованию результатов моделирования для проектирования архитектуры и каналов взаимодействия партнерской сети.

Ключевые слова: риск взаимодействия, архитектура сложной системы взаимодействия, партнерская сеть, экономика взаимодействий

Helen Kazakova, Anna Kolomytseva, **Model basis the assessment of interaction risk in architecture affiliate network.** The article presents the author's interpretation of the concept of "risk and reliability interaction interaction channels partner business networks as complex systems of interaction. The stochastic risk," a critical analysis of the methods of risk management and all forms of its manifestations in the assessment model of interaction and recommendations for the use of simulation results for the design of the architecture and interaction channels partner network.

Keywords: risk and reliability interaction, architecture complex systems of interaction, partner network, economy of interactions

Статья поступила в редакцию 11.09.2017
Рекомендована к публикации доктором технических наук В.Н. Павлышом

Программная система для оценки качества настройки струнного инструмента

Леснов Е.В., Андрюхин А.И.
Донецкий национальный технический университет
s4eniel@gmail.com

Леснов Е.В., Андрюхин А.И. Программная система для оценки качества настройки струнного инструмента. В данной работе рассмотрены различные вариации тюнеров, приведено описание проектирования и реализации собственной программной системы для оценки качества настройки струнных инструментов, реализующей алгоритм считывания звукового потока и его последующую обработку.

Введение

На сегодняшний день многие люди увлекаются игрой на гитаре. Кто-то более серьезно, кто-то менее, но ни один гитарист не обходится без тюнера. Современные технологии предоставляет возможность настроить музыкальный инструмент практически в любых условиях, имея при себе необходимый гаджет.

В настоящее время, платформа Android является одной из самых быстроразвивающихся мобильных операционных систем. Регулярно выпускаются новые устройства, причем каждое следующее поколение на порядок мощнее предыдущего. Следует отметить, что данная ОС является open source и распространяется по лицензии GNU. Эти факторы в совокупности и создают огромную популярность данной платформы.

Таким образом, с оглядкой на популярность платформы Android, решение создать приложение на базе данной ОС является очень актуальным.

Постановка задачи

Тюнер — отдельное устройство или программа, облегчающие настройку музыкальных инструментов. Использование тюнера облегчает задачу в разы: от музыканта требуется только следить за шкалой или другой индикацией, предусмотренной прибором, и смотреть, чтобы отклонение от вышеупомянутого эталона не было слишком велико[1].

В связи с развитием мобильных устройств не так давно появилась такая разновидность тюнеров, как мобильный тюнер, который представляет собой программное обеспечение для смартфонов. Для настройки инструмента требуется, собственно, инструмент и смартфон.

Далее, музыкант играет каждую струну по очереди и, смотря на экран гаджета, определяет в какую сторону нужно крутить колки. Были

проанализированы существующие аналоги и на основании этих данных были разработаны основные требования и поставлены основные задачи.

Основная цель — разработать приложение для начинающих музыкантов, помогающее им настроить инструмент, а также получить быстрый доступ к табулатурам основных типов аккордов. Кроме того, приложение должно иметь интуитивно понятный и максимально информативный интерфейс, иметь широкий выбор строев инструмента, охватывающий все стандарты исполнения.

Приложение должно иметь высокую скорость обработки входящего звукового потока, а также иметь расширение функционала в виде интерфейса взаимодействия с базой данных, содержащей в себе изображения табулатур основных типов аккордов.

Приложение должно работать на платформе Android актуальной версии.

Структура программной системы

Посмотрим на статическую структурную диаграмму, показывающую разбиение программной системы на структурные компоненты и связи между компонентами, изображенную на рисунке 1.

Имеется общий пакет com.saeniel.tuner, внутри которого находятся классы, отвечающие за функциональность, собственно, тюнера, а также подпакет, содержащий в себе классы, реализующие функциональность помощника по поиску аккордов в базе данных.

Как видно из диаграммы, после компиляции, на выходе получится .apk файл, готовый к установке на устройство Android.

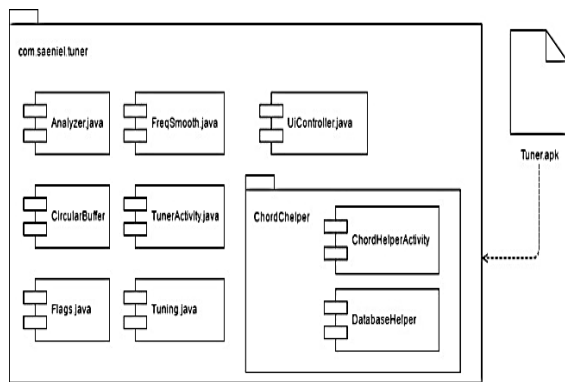


Рисунок 1 — Диаграмма развёртывания

Диаграмма вариантов использования системы представлена на рисунке 2.

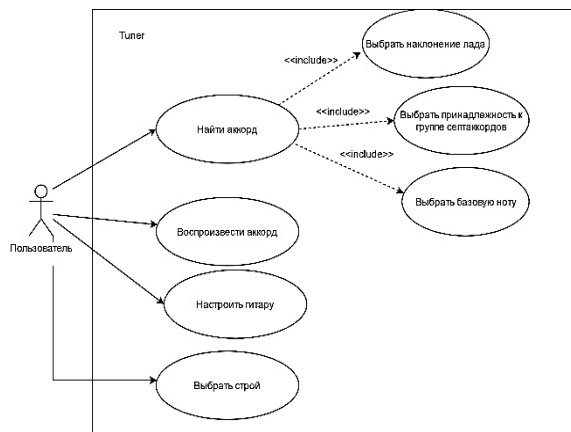


Рисунок 2 — Диаграмма вариантов использования

Пользователь имеет возможность совершить ряд действий: найти заданный аккорд, воспроизвести его звучание, выбрать строй, по которому будет настраиваться инструмент и, собственно, настроить гитару.

Данная программная система каждый момент времени находится в определенном состоянии. На рисунке 3 приведена диаграмма состояний.

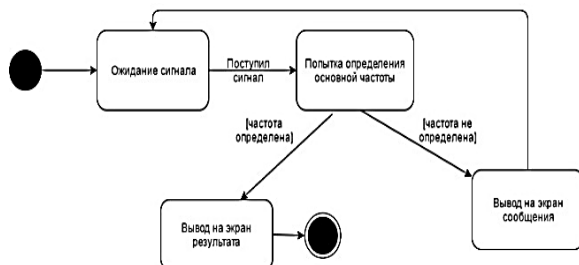


Рисунок 3 — Диаграмма состояний

Проектирование структур системы

При разработке приложения были использованы средства Android SDK для реализации логики приложения. Для хранения информации необходимой для работы помощника по поиску аккордов была спроектирована база данных. Для работы базы данных используется свободная, кроссплатформенная система управления базами данных SQLite.

Исходя из требований, выдвинутых к разрабатываемой системе, была спроектирована база данных. На рисунке 4 представлена схема базы данных, отвечающая за хранение аккордов

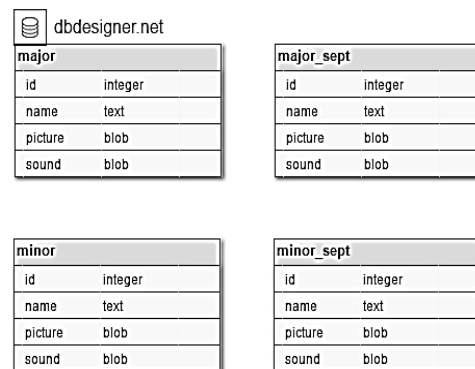


Рисунок 4 – Схема БД

В базе данных содержится четыре таблицы, которые отображают четыре основных типа аккордов: мажорные аккорды, мажорные септаккорды, минорные аккорды, минорные септаккорды. В таблицах хранится такая информация, как название аккорда, его графическое отображение в виде табулатуры и звуковой файл, позволяющий услышать звучание аккорда.

Как видно из представленной схемы базы данных, между таблицами отсутствуют связи. Это легко объясняется тем фактом, что все таблицы являются справочниками, что, по сути, означает их неизменяемость.

И хоть между таблицами нет связей, их все объединяет одинаковая структура. Во все таблицы входят только следующие поля:

- id — уникальный порядковый номер, идентификатор;
- name — название аккорда;
- picture — графическое изображение аккорда, его табулатура;
- sound — звучание аккорда.

Программная реализация

Для разработки системы был выбран высокоуровневый язык программирования Java. В качестве интегрированной среды разработки была выбрана Android Studio.

При создании данной программной системы был использован объектно-ориентированный подход.

Данное приложение использует шаблон «Наблюдатель». Этот поведенческий шаблон создает механизм у класса, который позволяет получать экземпляру объекта этого класса оповещения от других объектов об изменении их состояния, тем самым наблюдая за ними

На этапе проектирования отдельные сущности были выделены в классы, согласно самой концепции ООП. На рисунке 5 изображена диаграмма классов разработанного приложения.

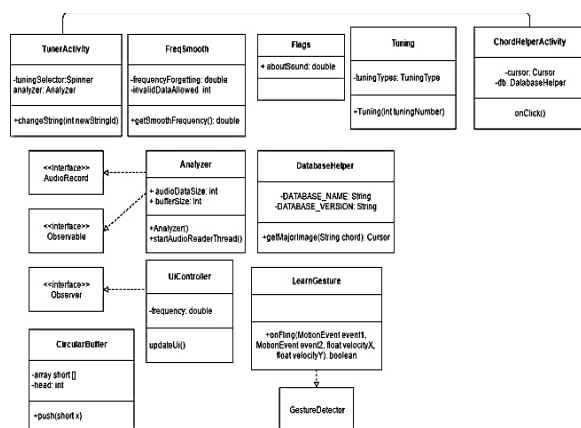


Рисунок 5 – Диаграмма классов

Рассмотрим более детально классы данного приложения.

Класс TunerActivity отвечает за функцию настройки инструмента. Он отвечает за принятие информации из классов, реализующих логику обработки входного звукового сигнала, а также за отображение результатов проанализированного сигнала на экране устройства.

Класс Flags содержит в себе флаги информирования пользователя.

Класс CircularBuffer реализует кольцевой буфер. Кольцевой буфер — это структура данных, использующая единственный буфер фиксированного размера, как будто бы после последнего элемента сразу же снова идет первый. Такая структура легко предоставляет возможность буферизации потоков данных.

Класс FreqSmooth позволяет «сгладить» частоту, тем самым отделив полезную частоту от ненужного сигнала.

Класс Analyzer отвечает за считывание анализ входного потока аудиоданных. Этот класс является «наблюдаемым» относительно класса UiController, согласно шаблону «наблюдатель». После проведенного анализа, результат передается в класс UiController, где происходит визуализация проанализированных данных.

Класс Tuning отвечает за графическую интерпретацию логики работы тюнера. Содержит

в себе механизм, позволяющий перейти от понятия «частота» к понятию «нота».

Класс UiController отвечает за отображение на экране смартфона обработанного звука. Является «наблюдателем» относительно класса Analyzer, согласно шаблону «наблюдатель».

Класс LearnGesture отвечает за распознавание жестов, позволяющих переключиться между тюнером и помощником аккордов.

Класс DatabaseHelper осуществляет взаимодействие между базой данных и приложением. Содержит функции получения информации из базы.

Класс ChordHelperActivity отвечает за логику взаимодействия пользователя с помощником аккордов. Основную логику содержит в себе обработчик кнопки «Поиск». Нажатие на данную кнопку запускает процесс парсинга аккорда. Определяется базовая нота аккорда, затем определяется наклонение лада (мажор или минор) и является ли аккорд септаккордом.

Алгоритм работы тюнера изображен на рисунке 6.

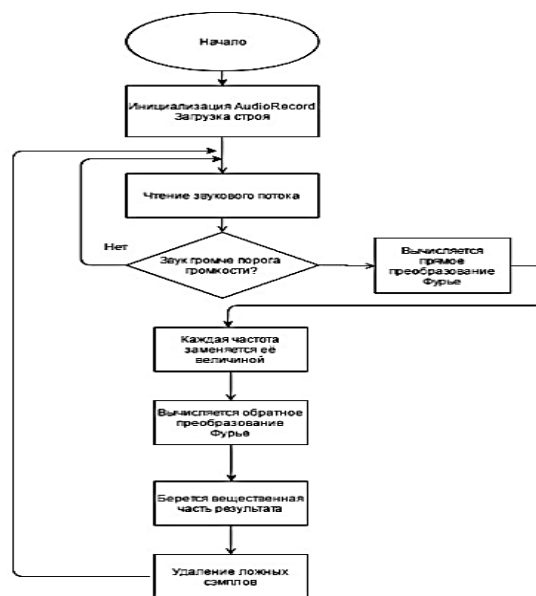


Рисунок 6 – Алгоритм работы тюнера

При запуске приложения создается объект класса, отвечающий за запись звука. Далее, происходит загрузка строя инструмента и начинается чтение звукового потока. Считывается порция сигнала и сразу определяется его громкость. Если значение громкости недостаточно велико, считывание сигнала происходит заново. При условии, что громкость сигнала удовлетворительная начинается вычисление автокорреляции первого порядка. Вычисляется прямое преобразование Фурье, после которого значение частоты заменяется

вычисленным значением. Затем, вычисляется обратное преобразование Фурье и от полученного результата берется вещественная часть. После проведенных вычислений начинается процесс удаления «ложных» сэмплов, после чего на экране отображается сыгранная пользователем нота.

Непрерывная функция и представление её рядом Фурье

Математической основой спектрального анализа сигналов является преобразование Фурье.

Математически, сигнал длительностью T секунд является некоторой функцией $f(x)$, заданной на отрезке $\{0, T\}$ (x в данном случае — время). Такую функцию всегда можно представить в виде суммы гармонических функций (синусоид или косинусоид) вида:

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{+\infty} A_k \cos\left(2\pi \frac{k}{T} x + \theta_k\right) \quad (1), \text{ где}$$

k – номер тригонометрической функции (номер гармонической составляющей, номер гармоники)
 T – отрезок, где функция определена (длительность сигнала)
 A_k – амплитуда k -ой гармонической составляющей,
 θ_k – начальная фаза k -ой гармонической составляющей.

Среднеквадратичное отклонение ряда от функции $f(x)$ будет стремиться к нулю, но несмотря на среднеквадратичную сходимость, ряд Фурье функции, вообще говоря, не обязан сходиться к ней поточечно[3].

Этот ряд может быть также записан в виде:

$$f(x) = \sum_{k=-\infty}^{+\infty} \hat{f}_k e^{i2\pi \frac{k}{T} x} \quad (2), \text{ где}$$

$\hat{f}_k$ – k -ая комплексная амплитуда.

Или так:

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{+\infty} [a_k \cos\left(2\pi \frac{k}{T} x\right) + b_k \sin\left(2\pi \frac{k}{T} x\right)] \quad (3)$$

Связь между коэффициентами (1) и (3) выражается следующими формулами:

$$A_k = \sqrt{a_k^2 + b_k^2}$$

и

$$\theta_k = \arctg \frac{b_k}{a_k}.$$

Пример использования системы

Разработанный программный продукт является приложением, предназначенным для начинающих музыкантов и совмещает в себе функционал тюнера и интерфейс по поиску табулатур аккордов. Проводя циклически анализ входного потока данных из микрофона

устройства, приложение определяет основную частоту, звучащую в реальном времени. Результаты сравнения фактической частоты с эталонной выводятся на экран, что позволяет увидеть отклонение от эталона.

Помощник поиска аккордов представляет из себя интерфейс взаимодействия с базой данных. Заполнив соответствующие поля, пользователь имеет возможность найти в базе данных изображение табулатуры искомого аккорда, а также его звуковой файл, содержащий звучание аккорда. Изображения и звуковые файлы хранятся в базе как массивы байт, которые в приложении приводятся в воспроизводимые форматы.

После запуска приложения, по умолчанию запускается тюнер. Скриншот тюнера приведен на рисунке 7.



Рисунок 7 – Экран тюнера

На данном этапе пользователю следует поднести смартфон к резонатору гитары (если речь идет о классической или акустической гитаре) или же динамику комбоусилителя, если настраиваемый инструмент — электрогитара. Затем, требуется играть открытые струны медленно, чтобы дать время приложению на анализ. Проанализировав звук, приложение выдаст результат в строку состояния и отобразит графически отклонение от эталона.

Изображение работы тюнера приведено на странице 8.



Рисунок 8 – Работа тюнера

По окончании настройки инструмента, пользователь имеет возможность просмотреть табулатуру того или иного аккорда, а также услышать, как он звучит. На выбор пользователя предоставлены четыре основных типа аккордов: минорный, мажорный, минорный септаккорд, мажорный септаккорд.

Для того, чтобы перейти из режима тюнера в режим помощника поиска аккордов, пользователь должен совершить действие «свайп влево» или «свайп 46 вправо». Загрузится помощник поиска аккордов, изображение экрана которого показано на рисунке 8.

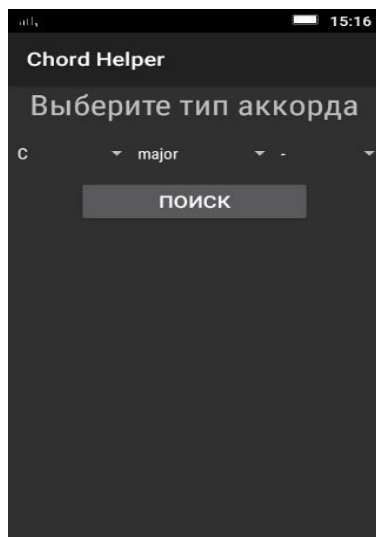


Рисунок 9 – Экран помощника поиска аккордов

Пользователь имеет возможность выбора базовой ноты аккорда, наклонение лада (минор или мажор), а также принадлежность аккорда к группе септаккордов.

После выбора соответствующих вариантов в списках, требуется нажать кнопку

«Поиск». После поиска в базе данных будет произведена загрузка изображения и звукового файла.

По нажатию на изображение табулатуры запустится воспроизведение данного аккорда встроенным плеером.

На рисунке 10 показан результат поиска в базе данных по запросу «до мажор септаккорд».



Рисунок 10 – Работа помощника по поиску аккордов

Тестирование

При разработке данного программного обеспечения было использовано тестирование юзабилити.

Юзабилити-тестирование — это исследование, выполняемое с целью определения, удобен ли некоторый искусственный объект (такой как веб-страница, пользовательский интерфейс или устройство) для его предполагаемого применения[2].

Для оценки качества интерфейса или пригодности использования продукции были использованы следующие юзабилити-метрики:

- результативность;
- эффективность;
- удовлетворенность.

В результате тестирования, фокус-группа определила, что приложение является интуитивно-понятным и эффективным. Последний респондент проводил тест с экстремально низким строем инструмента, и в следствие этого, приложение не всегда могло адекватно определить ноту.

Результаты тестирования приведены на рисунке 11.

Результативность	Эффективность	Удовлетворенность
90 %	30%	+
90%	30%	+
100%	30%	+
90%	30%	+
95%	30%	+
85%	45%	+/-

Рисунок 11 – Результаты юзабилити-тестирования

Выводы

В ходе выполнения данной работы было разработано программное обеспечение, предназначенное для начинающих музыкантов. Данный программный продукт представляет из себя тюнер и помощник поиска табулатур, написанный с использованием языка Java, СУБД MySQL для платформы Android. Приложение позволяет выполнить основные функции, связанные с настройкой струнных музыкальных инструментов.

Помимо этого, в рамках программного продукта реализован помощник поиска аккордов, реализующий поиск заданного аккорда в базе данных.

В дальнейшем планируется улучшать и дорабатывать систему оптимизируя основной алгоритм путем уменьшения времени обработки сигнала и повышении точности обработки, т.к.

именно эти два параметра являются самыми важными в тюнере.

В настоящее время основная функциональность разработки была расширена за счет добавления помощника по поиску аккордов, планируется еще больше улучшить представленный помощник, превратить его в полноценный справочник по теории музыки в мобильном приложении, содержащий в себе исчерпывающие описания и звуковые примеры.

Реализация вышеописанных планов позволит сделать данное приложение конкурентноспособным на рынке аналогичных программных продуктов, что поможет вывести распространение разработанного приложения на коммерческий уровень.

Литература

1. Тюнер [Электронный ресурс] // guitarlesson. – Режим доступа: <http://guitarlesson.ru/stati/tyuner-dlya-gitary.html>. – Загл. с экрана.
2. Тестирование юзабилити [Электронный ресурс] // artw. – Режим доступа: <https://artw.ru/blog/archives/3537/>. – Загл. с экрана
3. Ряд Фурье [Электронный ресурс] // Wikipedia. – Режим доступа: https://ru.wikipedia.org/wiki/Ряд_Фурье
4. Эдмунд Лэй. Цифровая обработка сигналов для инженеров и технических специалистов / Эдмунд Лэй. – Москва: ООО «Группа ИДТ», 2007

Леснов Е.В., Андрюхин А.И. Программная система для оценки качества настройки струнного инструмента. В данной работе рассмотрены различные вариации тюнеров, приведено описание проектирования и реализации собственной программной системы для оценки качества настройки струнных инструментов, реализующей алгоритм считывания звукового потока и его последующую обработку.

Ключевые слова: качество, настройка, струнный инструмент, тюнер, Java, Android

Lesnov E.V., Andryukhin A.I. A software system for evaluating the quality of tuning a string instrument. In this paper, various variations of tuners are considered. A description of the design and implementation of its own software system for evaluating the quality of tuning of stringed instruments is given. This system implements the algorithm of reading the audio stream and its subsequent processing.

Keywords: quality, tuning, string instrument, tuner, Java, Android

Статья поступила в редакцию 20.11.2017

Рекомендована к публикации доктором технических наук В.Н. Павлышом

Разработка программного обеспечения для формирования оптимальных маршрутов быстрой доставки товаров

С.А. Макогон, О.Д. Ситникова

Донецкий национальный технический университет
makogon_94@mail.ru, olga_d_s@mail.ru

Макогон С.А., Ситникова О.Д. Разработка программного обеспечения для формирования оптимальных маршрутов быстрой доставки товаров

В данной работе была рассмотрена задача построения оптимального маршрута быстрой доставки товаров, существующие программные комплексы, предоставляющие подобный функционал на коммерческой основе, приведено описание проектирования реализации собственной системы, предоставляющей функционал, необходимый для построения оптимального маршрута быстрой доставки товаров с использованием модифицированного алгоритма муравьиного поиска.

Введение

В современном мире создаются все условия, необходимые для комфортной и как можно менее энергозатратной деятельности человека. Создаются различные устройства, которые минимизируют участие человека практически во всех отраслях производства. Однако, очень большое количество разнообразных задач невозможно выполнить без прямого либо косвенного вмешательства человека. Данная проблема обусловлена недостаточным уровнем развития технологий на данном этапе существования человечества. Одной из задач, требующих непосредственно вмешательства человека, является доставка товаров.

Практически все задачи, связанные с доставкой и распределением товаров, являются задачами такого подраздела экономики, как логистика. Именно эта отрасль занимается изучением способов построения как можно более коротких маршрутов для оптимизации работы по распределению, перевозке и доставке товаров.

В математической интерпретации, данные задачи сводятся к решению NP-трудной задачи, которая называется «Задача коммивояжера».

Технологии не стоят на месте, и, поэтому, для построения кратчайших маршрутов доставки создаются различные приложения, обеспечивающие временной расчет своевременного выполнения задания по обеспечению потребителей необходимыми товарами.

Постановка задачи

Необходимо спроектировать и реализовать программное обеспечение, реализующее основные функции по поиску оптимального кратчайшего маршрута для быстрой доставки товара.

Торговая сеть большого города включает N магазинов (торговых центров), в каждом из которых есть кулинарный отдел, предоставляющий услугу доставки горячих обедов по заявкам с учетом меню. Услуга пользуется спросом, количество заявок очень большое. Особенность услуги в том, что она должна быть выполнена в кратчайший срок за время не более T минут.

В каждом торговом центре имеется K_i курьеров. Курьер выезжает на доставку сразу по нескольким адресам. Распределением заказов занимается один диспетчер сети. Нужно помочь ему сформировать маршруты для курьеров с минимизацией расходов на все маршруты и при ограничении времени доставки: каждый курьер должен быть в пути не более T минут. Также каждый курьер не может обслужить более Z заказов в одном маршруте. Все заказы должны быть обслужены в обязательном порядке. Известны адреса заявок. Все расстояния известны, также, известно ориентировочное время в пути между любыми точками. Загрузка всех курьеров не обязательна, критерием является минимизация суммарного расстояния (то есть затраты на бензин) при ограничении на время по маршруту.

Основная цель — спроектировать рабочую систему для построения оптимального маршрута быстрой доставки товаров с использованием технологий, предоставляемых картографическими

API. Изучить и использовать алгоритмы для поиска кратчайших путей, закрепить и улучшить навыки анализа, проектирования и конструирования программного обеспечения.

Освоить базовые понятия и функционал HTML, Javascript, CSS и развить практические навыки работы с данными языками. Получить практические навыки, связанные с использованием возможностей картографических API, в частности Google Maps API, Open Street Maps API и GraphHopper Route Optimization API.

Одним из главных требований к продукту выступает модульность. Приложение должно быть расширяемым – иметь возможность изменить или увеличить часть функционала системы без необходимости переписывания кода всей системы или замены архитектуры.

Актуальность работы обусловлена востребованностью подобных систем на рынке, потребностью в создании открытой системы, реализующей основные функции, с учетом преимуществ и недостатков существующих аналогов, а также личной необходимостью.

Структура программной системы

Система должна иметь модульную структуру (разделяться на пакеты/модули) для обеспечения расширяемости системы без необходимости редактирования существующего кода, а также с целью возможности повторного использования кода.

Для обеспечения должного уровня надежности функционирования программное изделие должно:

- минимизировать количество действий, необходимое для выполнения операции;
- уменьшить время ожидания отклика приложения, либо сообщать о проведении длительной операции;
- проводить проверку пользовательского ввода на клиентской и серверной части;
- обеспечить обработку исключительных ситуаций, вызванных ошибочными действиями пользователей с целью предотвращения некорректного завершения работы приложения.

Система должна иметь постоянный доступ к сети Интернет для полного и бесперебойного функционирования и выполнения поставленных задач.

На рисунке 1 приводится диаграмма, иллюстрирующая взаимосвязь модулей программного продукта.

Диаграмма вариантов использования приложения пользователем предоставлена на рисунке 2.

На рисунке 3 изображена диаграмма последовательности для запроса на сервер Google Maps API.

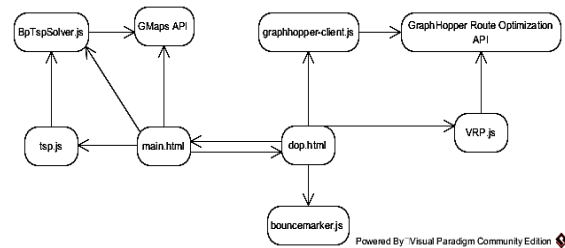


Рисунок 1 — Схема связи модулей приложения

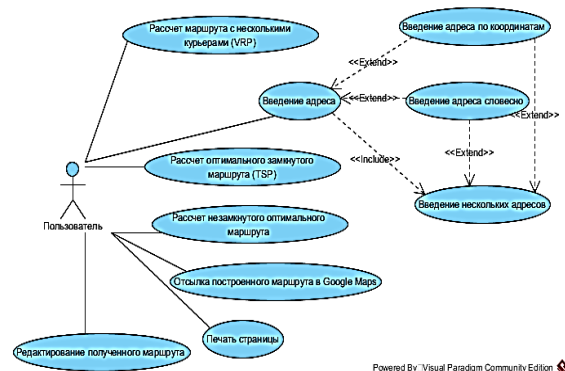


Рисунок 2 – Диаграмма вариантов использования

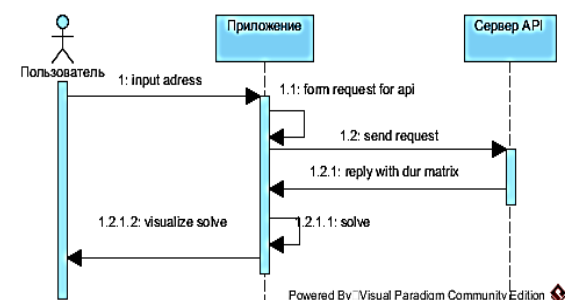


Рисунок 3 – Диаграмма последовательности для запроса на сервер GMaps API

Эта диаграмма наглядно иллюстрирует последовательность действий программного продукта под влиянием запроса со стороны пользователя.

Изначально пользователь производит ввод необходимых данных в форму, программа получает эти данные в «сыром» виде и преобразовывает их в формат, необходимый для корректного выполнения пользовательского запроса на сервер с API. Сервер обрабатывает полученный им запрос и выдает результат, необходимый программному комплексу для решения задачи. После получения ответа, приложение производит необходимые вычисления и предоставляет полученный результат пользователю.

Проектирование структур системы

Ввиду необходимости использования сторонних API для получения необходимых для вычисления данных, следует подготовить почву для формирования корректного формата запросов и создать локально необходимые типы структур. Это становится возможным благодаря наличию документации использованных в системе API.

Для осуществления работы с объектом карт необходимо изначально его создать в контексте программы. Происходит это путем создания какого-либо контейнера на заглавной странице и функции инициализации карты. На рисунке 4 изображен фрагмент кода, выполняющего данное действие.

```
function initMap(center, zoom, div) {  
    var myOptions = {  
        zoom: zoom,  
        center: center,  
        mapTypeId: google.maps.MapTypeId.ROADMAP  
    };  
    gebMap = new google.maps.Map(div, myOptions);  
    google.maps.event.addListener(gebMap, "click", function(event) {  
        tsp.addWaypoint(event.latLng, addWaypointSuccessCallback);  
    });  
}
```

Рисунок 4 – Создание объекта карт

В переменную myOptions заносятся данные о зуммировании карты, ее центровке и устанавливается id типа карты.

После того, как объект карты создан, можно перейти к созданию второго необходимого для работы элемента, а именно — метки. На рисунке 5 изображен фрагмент кода для инициализации пользовательского маркера.

```
var icon;  
icon = new google.maps.MarkerImage("newicons/red" + (num + 1) + ".png");  
var marker = new google.maps.Marker({  
    position: latlng,  
    icon: icon,  
    map: gebMap  
});
```

Рисунок 5 – Инициализация пользовательского маркера

Как видно, создается переменная с названием icon, в которой будет храниться иконка, необходимая для отображения маркера. Создается так же и переменная marker, в которой будет храниться информация о самом маркере в форме, доступной для интерпретации посредством Google Maps API. В переменную position заносятся данные о координатах положения маркера, в переменную icon заносятся данные об отображаемой на месте маркера картинке, в переменную map заносится информация на какую карту будет отображаться заданная метка.

Для осуществления запроса на построение маршрута необходимо создать структуру данных, приведенную на рисунке 6.

Для использования маршрутов создается объект типа DirectionsService и вызывается DirectionsService.route() для инициализации запроса в службу Directions.

```
var myGebDirections = new google.maps.DirectionsService();  
  
myGebDirections.route({  
    origin: origin,  
    destination: destination,  
    waypoints: wayArrChunk2,  
    avoidHighways: avoidHighways,  
    avoidTolls: avoidTolls,  
    unitSystem: directionunits,  
    travelMode: travelMode  
},
```

Рисунок 6 – Создание маршрута

Данный объект может содержать следующие поля:

- origin (обязательное) — указывает начальную точку маршрута. Может быть указан в виде строки, значения LatLng (координат) или объекта google.maps.Place.
- destination (обязательное) — указывает конечную точку маршрута. Параметры аналогичны параметрам поля origin.
- travelMode (обязательное) — указывает какой способ передвижения необходимо использовать при построении маршрута
- transitOptions (необязательное) — указывает значения, которые относятся только к тем запросам, где поле travelMode имеет значение TRANSIT.
- drivingOptions (необязательное) — указывает значения, относящиеся только к запросам, где поле travelMode имеет значение DRIVING.
- unitSystem (необязательное) — указывает, какую систему единиц измерения следует использовать для отображения результатов.
- waypoints [] (необязательное) — определяет массив промежуточных точек DirectionsWaypoint.
- optimizeWaypoints (необязательное) — указывает, что маршрут, использующий предоставленные значения waypoints, может быть оптимизирован путем расположения этих промежуточных точек в более эффективном порядке.
- provideRouteAlternatives (необязательное) — если это поле имеет значение true, служба Directions может указывать несколько альтернативных маршрутов в ответе.
- avoidHighways (необязательное) — если это поле имеет значение true, рассчитанный маршрут должен избегать крупных магистралей, если это возможно.
- avoidTolls (необязательное) — если это поле имеет значение true, рассчитанный маршрут должен избегать платных дорог, если это возможно.
- region (необязательное) — код региона, указываемый как значение ccTLD ("домен верхнего уровня") из двух символов[1].

Программная реализация

Для разработки функционала программной системы был выбран мультипарадигменный язык программирования Javascript в связке с HTML и CSS. В качестве интегрированной среды разработки была выбрана программа Notepad++ ввиду ее доступности и открытости для разработчиков.

Язык Javascript является простым, доступным для понимания, кроссплатформенным, мультипарадигмальным языком программирования. Обычно, данный язык используется в браузерах, как средство для придания интерактивности веб-страницам.

Однако, существуют примеры использования данного языка в браузерных операционных системах, в качестве средства для создания серверных приложений, в прикладном программном обеспечении даже в офисных приложениях. Ввиду перечисленных особенностей, в последнее время ведутся работы по созданию новых средств для выполнения задач разработки в более масштабном объеме. Основными архитектурными чертами языка являются:

- динамическая типизация;
- слабая типизация;
- автоматическое управление памятью;
- прототипное программирование;
- функции как объекты первого класса.

При разработке Javascript на него оказали влияние множество языков. Одним из главных критериев реализации была цель сделать язык как можно более похожим на Java, но более легким для использования даже рядовыми пользователями[2].

Для разработки программной системы был выбран именно этот язык программирования ввиду указанных выше преимуществ, возможности получить доступ к различным веб-ориентированным системам и личным интересом в изучении возможностей данного языка как для использования в веб-приложениях, так и в качестве работы разрабатываемой системы на стороне пользователя. Так же, приятным дополнением ко всему перечисленному является и то, что для работы с языком Javascript не требуется установка особых интегрированных сред разработки, так как код может выполняться в браузере, а для его написания и редактирования достаточно наличия простого текстового редактора.

Для реализации функционала программного обеспечения использовались следующие библиотеки и API:

- библиотека jQuery;
- Google Maps API;
- OpenStreetMaps API;
- GraphHopper API.

В ходе разработки были созданы собственные библиотеки, написанные с использованием языка Javascript, а именно такие:

- TSPSolver;
- TSP.

Данные средства были выбраны ввиду их доступности, открытости, наличию обширной документации, отзывчивому и приятному комьюнити разработчиков, которые активно поддерживаются создателями данных средств и предоставляют широкий спектр услуг и возможностей для использования и дальнейшего развития предлагаемых продуктов.

Однако, у выбранных средств имеются и недостатки, которые выясняются при углубленном их изучении. Так, Google Maps API предоставляют услуги по своей работе как на платной, так и на бесплатной основе. При использовании на бесплатной основе существуют весьма существенные ограничения, установленные корпорацией Google на использование картографических API. Данные ограничения заключаются в максимальном количестве ежедневных загрузок карты, установленных в размере 25000, и выполнении запросов к сервисам геолокации, геокодирования, установленных в размере 2500 в день[3].

Основной функцией приложения является предоставление пользователю возможности по построению оптимальных маршрутов быстрой доставки товаров.

Пользовательский интерфейс реализован путем осуществления перехода между различными операциями. Данные для отображения отрисовываются при загрузке приложения в реальном времени.

Разрабатываемое приложение имеет два модуля, в которых реализован основной функционал. Эти модули представлены статичными HTML-страницами с использованием подключенных в зависимости исполняемых файлах Javascript и таблиц стилей.

Логика работы данных модулей зависит от действий, выполняемых пользователем во время пребывания на данных страницах.

При разработке методов решения задачи были использованы алгоритм муравьиной колонии, 2-opt алгоритм и жадный алгоритм. Описание данных алгоритмов будет приведено ниже.

Использованные алгоритмы

Муравьиный алгоритм — один из эффективных полиномиальных алгоритмов для нахождения приближенных решений задачи коммивояжера и решения аналогичных задач поиска маршрутов на графах. Суть метода заключается в исследовании и воспроизведении поведения таких социальных насекомых, как

муравьи, которые ищут пути от колонии к источнику питания. Данный метод представляет собой метаэвристическую оптимизацию[4].

В основе алгоритма заключен метод поиска путей муравьями, а именно маркировка более удачного маршрута большим количеством феромона.

Любой муравьиный алгоритм, вне зависимости от возможных вариантов модификаций, представим в следующем виде:

0. Проверка условия выхода. Если проверка не пройдена, то переход на шаг 2.

1. Создаем муравьев.

Начальная точка, куда помещается муравей, зависит от ограничений, которые были заданы по условию задачи. На данном этапе так же задается начальный уровень феромона, который инициализируется небольшим положительным числом для того, чтобы на начальном шаге вероятности перехода в следующую вершину не были нулевыми.

2. Ищем решения.

Вероятность перехода из вершины i в вершину j определяется по следующей формуле (5.1):

$$P_{ij}(t) = \frac{\tau_{ij}(t)^\alpha \left(\frac{1}{d_{ij}}\right)^\beta}{\sum_{j \in \text{allowed nodes}} \tau_{ij}(t)^\alpha \left(\frac{1}{d_{ij}}\right)^\beta}, \quad (5.1)$$

где $\tau_{ij}(t)$ — уровень феромона,
 d_{ij} — эвристическое расстояние,
 α, β — константные параметры.

При $\alpha = 0$ выбор ближайшего города наиболее вероятен, то есть алгоритм становится жадным.

При $\beta = 0$ выбор происходит только на основании феромона, что приводит к субоптимальным решениям.

3. Обновляем феромон.

Уровень феромона обновляется в соответствии с приведённой формулой (5.2)

$$\tau_{ij}(t+1) = (1-p)\tau_{ij}(t) + \sum_{k \in \text{Colony that used edge}(i,j)} \frac{Q}{L_k}, \quad (5.2)$$

где p — интенсивность испарения,
 L_k — цена текущего решения для k -ого муравья,

Q — параметр, имеющий значение порядка цены оптимального решения.

4. Дополнительные действия (опционально).

Обычно здесь используется алгоритм локального поиска, однако, он может также появиться и после поиска всех решений.

Алгоритм представлен на рисунке 7.



Рисунок 7 – Общий алгоритм муравьиной колонии

Для варианта вычисления кольцевого замкнутого маршрута в приложении использовался алгоритм, включающий в свою работу дополнительный пункт в виде преобразование полученного пути с помощью алгоритма 2-opt.

Суть алгоритма 2-opt заключается в выборе 2 дуг построенного маршрута и их переборки. Если длина полученного пути будет короче, чем длина пути, признанного оптимальным, то новый путь признается оптимальным и записывается.

DirectionsService и вызывается DirectionsService.route() для инициализации запроса в службу Directions

Также, была создана функция, использующая метод жадного алгоритма.

Жадным алгоритмом называется алгоритм, который на каждом своем шаге принимает локально оптимальный выбор, допуская, что итоговое решение также окажется оптимальным. Следует заметить, что далеко не всегда это оказывается верным подходом к решению задачи. Для большого количества алгоритмических задач решение с помощью жадного алгоритма является приемлемым.

Пример использования системы

Данный программный продукт представляет собой модульную систему, реализующую функции поиска кратчайших путей для быстрой доставки товаров. Особенностью является то, что данный продукт можно использовать в качестве встраиваемого модуля в более сложных логистических системах. Стартовый экран представлен на рисунке 8.

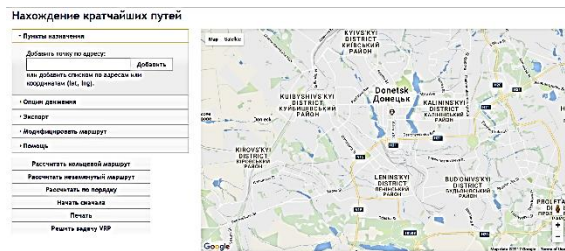


Рисунок 8 – Стартовый (главный) экран приложения

Далее, с главной страницы пользователь получает доступ уже ко всем возможным вариантам использования программной системы. На рисунке 9 показан вариант добавления маркера на карту путем клика на нее

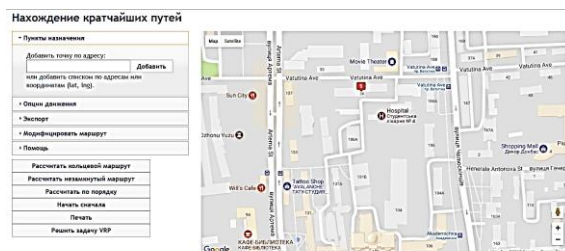


Рисунок 9 – Добавление маркера по клику на карте

При построении кольцевого маршрута заданная точка будет выступать в качестве начальной и конечной точки одновременно. На рисунке 10 показан построенный кольцевой маршрут

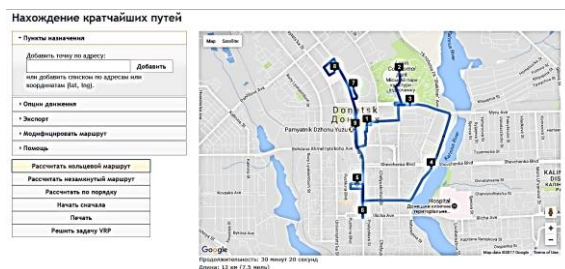


Рисунок 10 – Построенный кольцевой маршрут

Так же на рисунке 11 отображена часть словесных описаний, доступных для рассчитанного маршрута

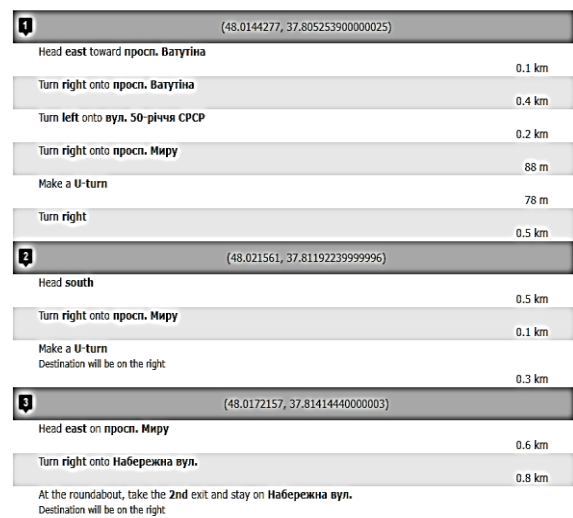


Рисунок 11 – Пример словесного описания построенного маршрута

При нажатии на кнопку «Начать сначала» состояние окна сбросится и вернется к начальному варианту, который был предоставлен на рисунке 8.

При вводе адреса словесно в текстовое поле программа преобразует этот текст в точное географическое название и отметит его на карте. На рисунке 12 показана возможность добавления адреса пользователем с использованием методов геолокации.

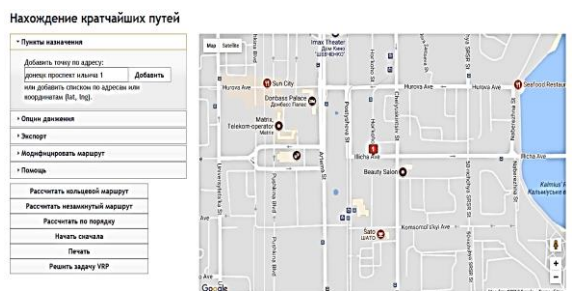


Рисунок 12 – Добавление словесного адреса на карту

При желании пользователя добавить несколько точек на карту, ему будет необходимо нажать на ссылку с текстом «добавить списком по адресам или координатам (lat, lng)».

На рисунке 13 показана возможность добавления нескольких адресов использованием методов прямой и обратной геолокации.

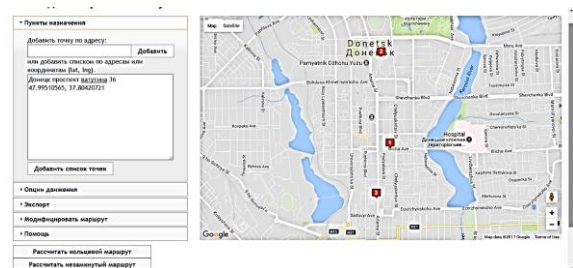


Рисунок 13 – Добавление списка адресов

При построении желаемого маршрута пользователь может указать дополнительные опции маршрута, которые представлены на рисунке 14.

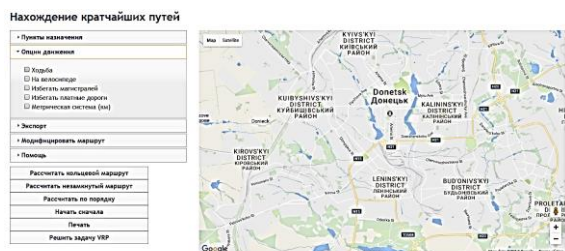


Рисунок 14 – Дополнительные опции построения маршрута

После успешного построения любого из маршрутов пользователю становятся доступны функции экспорта маршрута любым из перечисленных вариантов и его модификация, путем изменения порядка следования на желаемый.

При нажатии на кнопку «Решить задачу VRP» пользователь будет перенаправлен на другую форму, представляющую аналогичный функционал. На этой форме возможно выполнение построения маршрутов для нескольких курьеров с оптимизацией по загруженности всех доступных курьеров и минимизации их пути. под картой отображается суммарная длина всех найденных путей и время, затрачиваемое на выполнение всех заказов, с учетом того, что возможно один курьер будет выполнять все заказы. Форма приведена на рисунке 15.

Тестирование

При разработке данного программного обеспечения было использовано тестирование юзабилити.

Юзабилити-тестирование — это исследование, выполняемое с целью определения, удобен ли некоторый искусственный объект (такой как веб-страница, пользовательский интерфейс или устройство) для его предполагаемого применения[5].

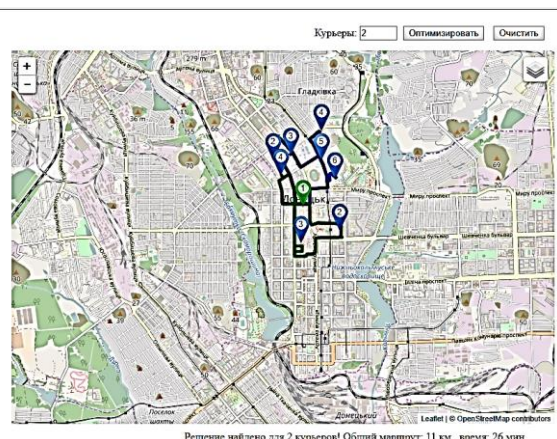


Рисунок 15 – Форма решения задачи VRP

Проверка эргономичности — метод оценки удобства продукта в использовании, основанный на привлечении пользователей в качестве тестировщиков, испытателей и суммировании полученных от них выводов.

При разработке данного программного продукта был использован непрямой метод тестирования. Была собрана фокус-группа (5 человек) экспертов в предметной области. Каждому эксперту было установлено данное приложение. Взаимодействие между экспертом и приложением протоколировалось. Результаты показаны в таблице 1.

Таблица 1 – Результаты юзабилити-тестирования

Результативность	Эффективность	Удовлетворенность
90 %	30%	+
90%	30%	+
100%	30%	+
90%	30%	+
95%	30%	+
85%	45%	+/-

В результате тестирования, фокус-группа определила, что приложение является интуитивно-понятным и эффективным. Последний респондент остался не слишком доволен скоростью работы программы и дизайном интерфейса

Выводы

В ходе выполнения работы было рассмотрена транспортная задача по определению кратчайших маршрутов доставки товаров. На основании проведенного анализа существующих аналогов были поставлены задачи.

Разработано программное обеспечение, предназначенное для формирования оптимальных маршрутов для быстрой доставки товаров. Данный программный продукт был написан с использованием языков HTML, CSS и Javascript и может быть использован как в качестве

самостоятельного ПО, так и в качестве встраиваемого модуля в системы типа 1С.

В дальнейшем планируется улучшать и дорабатывать систему.

Литература

1. JavaScript API - Google Maps JavaScript API | Google Developers [Электронный ресурс] // Google. – Режим доступа: <https://developers.google.com/maps/documentation/javascript/?hl=ru>. – Загл. с экрана|.

2. JavaScript [Электронный ресурс] // Википедия. – Режим доступа:

<https://ru.wikipedia.org/wiki/JavaScript>. – Загл. с экрана|.

3. Цены и планы | Цены и планы для Google Maps API | Google Developers [Электронный ресурс] // Google. – Режим доступа: <https://developers.google.com/maps/pricing-and-plans/?hl=ru>. – Загл. с экрана|.

4. Муравьиный алгоритм [Электронный ресурс] // YouTube. – Режим доступа: https://www.youtube.com/watch?v=EwDP_bAb-OI. – Загл. с экрана|.

5. Тестирование юзабилити [Электронный ресурс] // artw. – Режим доступа: <https://artw.ru/blog/archives/3537/>. – Загл. с экрана|.

Макогон С.А., Ситникова О.Д. Разработка программного обеспечения для формирования оптимальных маршрутов быстрой доставки товаров. В данной работе была рассмотрена задача построения оптимального маршрута быстрой доставки товаров, существующие программные комплексы, предоставляющие подобный функционал на коммерческой основе, приведено описание проектирования и реализации собственной системы, предоставляющей функционал, необходимый для построения оптимального маршрута быстрой доставки товаров с использованием модифицированного алгоритма муравьиного поиска

Makogon S.A., Sitnikova O.D. Developing of software for constructing optimal routes for the rapid delivery of goods". In this paper was considered the problem of constructing an optimal route for the rapid delivery of goods, existing software complexes that provide such functionality on a commercial basis, describe the design and implementation of its own system that provides the functionality necessary for constructing the optimal route for the rapid delivery of goods using a modified ant colony optimization algorithm.

Статья поступила в редакцию 7.10.2017

Рекомендована к публикации доктором технических наук В.Н. Павлышом

УДК 622.232

Исследование параметров гидросистемы механизированных крепей методом математического моделирования

В.Н. Павлыш, В.И. Зензеров
ГОУ ВПО «Донецкий национальный технический университет»
pavlyshvn@mail.ru

Павлыш В.Н., Зензеров В.И. Исследование параметров гидросистемы механизированных крепей методом математического моделирования. Рассматривается модифицированная математическая модель для исследования и расчета параметров гидросистемы секций механизированных крепей, обеспечивающих их устойчивую работу при выемке наклонных пластов.

Ключевые слова: Исследование, модификация, параметр, математическая модель, гидросистема.

Pavlysh V.N., Zenzerov V.I. The investigation of mechanical timbers hydrosystems parameters by mathematical modeling method. The modified mathematical model for researching and calculation of parameters of the hydrosystem for provided of stability of the sections of mechanized supports on inclined coal stratum are considered.

Keywords: investigation, modification, parameter, mathematical model, hydrosystem.

Актуальность задачи

Одной из причин снижения технико-экономических показателей процесса очистной выемки наклонных пластов является потеря устойчивости и направленности передвижения механизированной крепи в системе «угольный пласт – механизированная крепь – комбайн – конвейер» (до 56% общего времени простоев крепи связано с потерей устойчивости и направленности движения секций).

Особенно актуален этот вопрос для шахт, дорабатывающих запасы шахтных полей, где применение высокопроизводительных комплексов нового поколения экономически нецелесообразно, а применение существующих комплексов очистного оборудования требует научного обоснования вводимых конструктивных изменений.

Механизированная крепь, как одна из базовых машин очистного комплекса, в значительной мере определяет его производительность, надежность управления и безопасность работы. В свою очередь, работа механизированных крепей во многом определяется параметрами их гидросистемы. Как показывает опыт разработки наклонных пластов, именно за счет совершенствования гидросистем удастся повысить устойчивость и направленность передвижения секций механизированной крепи.

В этой связи обоснование и развитие методов исследования и расчета параметров при проектировании гидросистем механизированных крепей с характеристиками, обеспечивающими их эффективную эксплуатацию на наклонных пластах в условиях повышения нагрузок на очистной забой, является актуальной научной и практической задачей, имеющей отраслевое значение.

Состояние вопроса

Решение задачи исследования и расчета параметров механизированных крепей посвящены работы ряда авторов [1, 2, 5, 7]. Разработаны аналитические методы, проведены экспериментальные исследования. Вместе с тем, на современном этапе полученные результаты не обеспечивают достаточно полное решение задачи обеспечения устойчивости секций механизированных крепей при работе на наклонных угольных пластах, за счет внесения дополнительных элементов в конструкцию гидросистемы. В статье предлагается усовершенствованная математическая модель гидросистемы, позволяющая решить задачу исследования и расчета параметров гидросистемы секций механизированных крепей.

Цель и задачи работы

Целью работы является разработка современных средств исследования параметров гидросистем секций механизированных крепей для их модернизации, обеспечивающей

повышение устойчивости секций на наклонных угольных пластах и эффективности эксплуатации механизированных комплексов.

Для достижения поставленной цели решается задача разработки математической модели гидросистемы секций механизированных крепей и исследования параметров динамических процессов в гидросистеме при выполнении крепью операций технологического цикла в условиях отработки наклонных угольных пластов;

Содержание работы

В работе предлагается математическая модель гидросистемы крепи, суть которой сводится

к следующему. Система передвижения механизированной крепи представляет собой гидромеханическую систему.

Так как гидросистема крепи является многосекционной, то работу всей системы можно рассматривать на примере выполнения операций технологического цикла одной секцией крепи.

На рис. 1 представлена гидромеханическая схема гидросистемы механизированной крепи. При составлении схемы использованы исследования и графические обозначения, приведенные в работах [1, 2, 3].

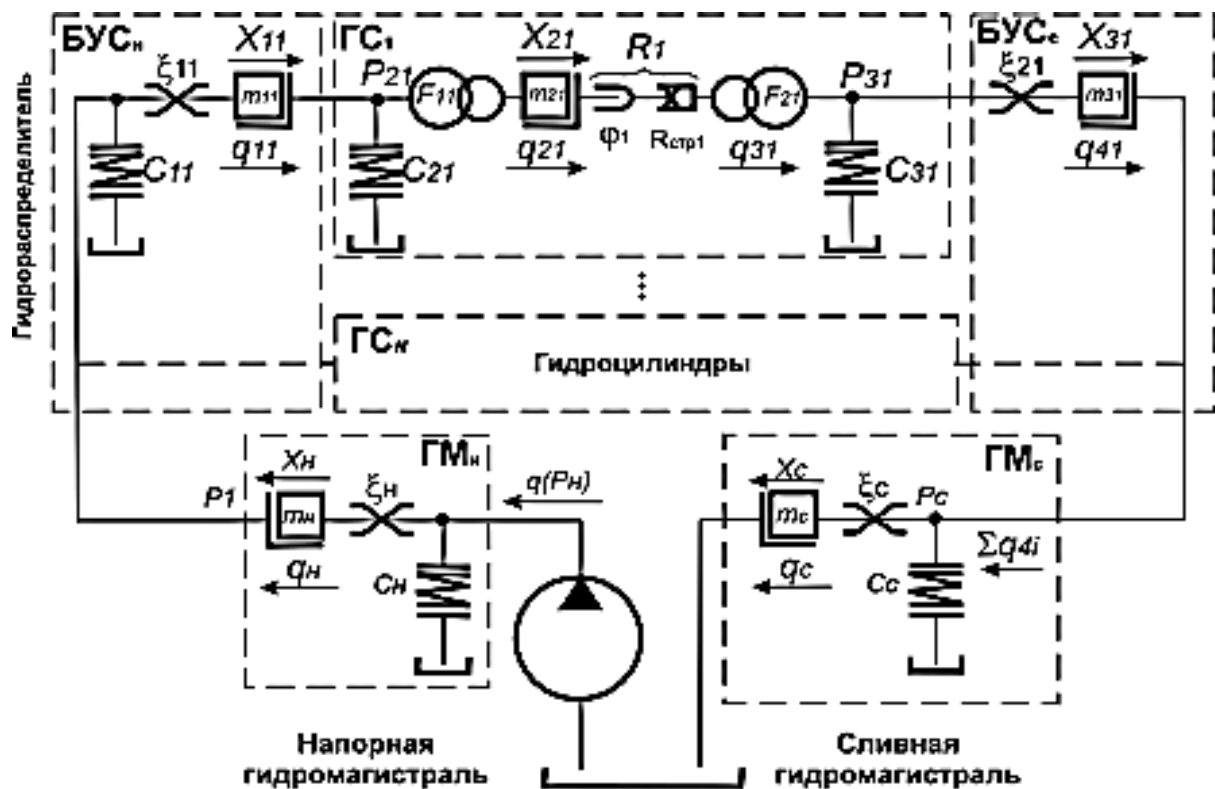


Рисунок 1 - Гидромеханическая схема гидросистемы механизированной крепи

Работа гидропривода описывается двумя типами уравнений:

- дифференциальные уравнения движения элементов системы, составленные по принципу Даламбера;
- уравнения баланса расходов, учитывающие, что жидкость в гидросистеме неразрывна и однородна.

Гидросистема крепи, схема которой приведена на рис. 1, состоит из N отдельных ветвей, моделирующих работу гидроцилиндров, и общих для всех гидроцилиндров участков

напорной и сливной магистралей. На основании правила узлов и ветвей [3] составим математическую модель каждого из этих участков.

Движение рабочей жидкости от насоса до гидрораспределителя секции крепи описывается системой трех дифференциальных уравнений (1).

$$\left. \begin{aligned} C_H \frac{dP_H}{dt} &= q(P_H) - q_H \\ \frac{m_H}{(f_H)^2} \cdot \frac{dq_H}{dt} &= P_H - \xi_H (q_H)^2 - P_1 \\ C_\Sigma \frac{dP_1}{dt} &= q_H - \sum_{i=1}^N q_{1i} \end{aligned} \right\}. \quad (1)$$

Математическая модель работы каждого гидроцилиндра состоит из системы шести дифференциальных уравнений (2).

$$\left. \begin{aligned} \frac{m_{1i}}{(f_{PBДi}^H)^2} \cdot \frac{dq_{1i}}{dt} &= P_1 - \xi_{1i} (q_{1i})^2 - P_{2i} \\ C_{2i} \frac{dP_{2i}}{dt} &= q_{1i} - q_{2i} \\ \frac{m_{2i}}{(F_{1i})^2} \cdot \frac{dq_{2i}}{dt} &= P_{2i} - \beta_i R_i(X_{2i}, X_{2i}^*) - \alpha_i P_{3i} \\ \frac{dX_{2i}}{dt} &= \beta_i q_{2i} \\ C_{3i} \frac{dP_{3i}}{dt} &= \alpha_i q_{2i} - q_{4i} \\ \frac{m_{3i}}{(f_{PBДi}^C)^2} \cdot \frac{dq_{4i}}{dt} &= P_{3i} - \xi_{2i} (q_{4i})^2 - P_C \end{aligned} \right\}. \quad (2)$$

Течение жидкости по сливной гидромагистральной от секции крепи в бак гидронасоса описывается системой двух дифференциальных уравнений (3).

$$\left. \begin{aligned} C_C \frac{dP_C}{dt} &= \sum_{i=1}^N q_{4i} - q_C \\ \frac{m_C}{(f_C)^2} \cdot \frac{dq_C}{dt} &= P_C - \xi_C (q_C)^2 \end{aligned} \right\}. \quad (3)$$

Системы дифференциальных уравнений (1), (2) и (3) образуют математическую модель гидросистемы механизированной крепи. Количество уравнений математической модели зависит от количества одновременно работающих гидроцилиндров N при моделировании различных технологических операций.

В математическую модель гидросистемы механизированной крепи входят семь групп параметров:

- давление жидкости в характерных точках гидросистемы - $P_H, P_1, P_{2i}, P_{3i}, P_C$;
- расход жидкости на отдельных участках гидросистемы - $q(P_H), q_H, q_{1i}, q_{2i}, q_{4i}, q_C$;
- коэффициенты упругости элементов гидросистемы - $C_H, C_{тр}, C_{рвд}, C_a, C_\Sigma, C_{2i}, C_{3i}, C_C$;
- коэффициенты гидравлических

сопротивлений участков гидросистемы - $\xi_H, \xi_{1i}, \xi_{2i}, \xi_C$;

- значения массы жидкости, перемещаемой на отдельных участках гидросистемы - $m_H, m_{1i}, m_{2i}, m_{3i}, m_C$;
- площади поперечных сечений элементов гидросистемы - $f_H, f_{PBДi}^H, f_{PBДi}^C, f_C$;
- сопротивления перемещению гидроцилиндров - $R_i(X_{2i}, X_{2i}^*)$.

Первые две группы параметров образуют вектор начальных значений, а в процессе моделирования - результирующий вектор зависимых переменных. Коэффициенты упругости определяют расход на деформацию элементов гидросистемы и сжимаемость жидкости, заполняющей ее под давлением. Для металлических труб коэффициент упругости рассчитывается по формуле [4]:

$$C_{TP} = \pi \left(\frac{1-x}{8} \cdot \frac{d^3 l_{TP}}{E_{TP} S_{TP}} + \frac{d^2 l_{TP}}{4 E_{жс}} \right), \quad (4)$$

где d, l_{TP} - соответственно диаметр, длина и толщина стенок трубы;

$E_{TP}, E_{жс}$ - объемные модули упругости материала трубы и рабочей жидкости;

x - коэффициент Пуассона.

Анализ формулы (4) показывает, что коэффициент упругости зависит от внутреннего объема трубы. Так как гидроцилиндры конструктивно выполнены из труб, то формулой (4) можно пользоваться для определения коэффициентов упругости полостей гидроцилиндров. Однако, при перемещении поршня гидроцилиндра объем напорной и сливной полостей изменяется.

Следовательно, в процессе моделирования величины коэффициентов упругости полостей гидроцилиндров будут изменяться в зависимости от величины перемещения поршня X_{2i} и их необходимо рассчитывать по формулам:

- для напорной полости i -го гидроцилиндра:

$$C_{2i} = (H_{oi} + X_{2i}) \left(\frac{1-x}{2} \cdot \frac{F_{1i} D_i}{E_{TP} S_{1i}^H} + \frac{F_{1i}}{E_{жс}} \right), \quad (5)$$

где H_{oi} - начальное расстояние от торца до поршня i -го гидроцилиндра;

D_i, S_{1i}^H - диаметр и толщина стенок напорной полости i -го гидроцилиндра.

- для сливной полости i -го гидроцилиндра:

$$C_{3i} = (H_{max} - X_{2i}) \left(\frac{1-\chi}{2} \cdot \frac{F_{2i} d_i}{E_{TP} S_{\psi i}} + \frac{F_{2i}}{E_{Ж}} \right), \quad (6)$$

где H_{max} – величина хода поршня i -го гидроцилиндра;

$d_i, S_{\psi i}$ – диаметр и толщина стенок сливной полости i -го гидроцилиндра.

Коэффициент упругости для рукавов высокого давления рассчитывается по формуле [4]:

$$C_{PВД} = \frac{V_{PВД}}{E_{np}}, \quad (7)$$

где E_{np} – приведенный модуль упругости, учитывающий сжимаемость жидкости и деформации рукавов;

$V_{ред}$ – объем рукавов высокого давления.

В качестве демпфирующего устройства на насосной станции устанавливаются гидроаккумуляторы. Коэффициент упругости гидроаккумулятора равен [5]:

$$C_a = m_a \frac{Q_a P_a^{\frac{1}{1+\nu}}}{\nu P_H^{\frac{1}{\nu}}}, \quad (8)$$

где m_a – число аккумуляторов, установленных в гидросистеме;

Q_a – объем газовой камеры при давлении зарядки;

P_a – давление зарядки гидроаккумулятора;

ν – показатель политропы ($\nu = 1,3 \div 1,4$).

Все элементы и узлы гидросистемы, по которым протекает жидкость, характеризуются гидравлическими сопротивлениями как местными (в заделках и ответвлениях, в сужениях, расширениях и поворотах потока), так и линейными, зависящими от протяженности каналов. Экспериментально доказано существование двух режимов установившегося движения жидкости. При малых скоростях движения ее частицы движутся параллельно друг другу и оси канала, не перемешиваясь. Такой режим назван ламинарным. При больших скоростях движения жидкости ее частицы движутся в направлении потока по хаотичной траектории и перемешиваются между собой. Такой режим течения жидкости назван турбулентным [2]. Для расчетов гидросистем механизированных крепей установлено, что, на основании определения числа Рейнольдса и сравнения его с критическим, режим течения жидкости является турбулентным [5]. Так как

поток жидкости проходит по трубопроводам и каналам разного сечения, то при определении коэффициента гидравлического сопротивления участка гидросистемы, все коэффициенты различных местных сопротивлений приводятся к выбранной площади поперечного сечения участка по формуле:

$$\xi_i = \xi'_i \frac{\rho}{2 f_i^2}, \quad (9)$$

где ξ'_i – безразмерный коэффициент местного гидравлического сопротивления, который определяется на основании экспериментальных и справочных данных для конкретного характера местного сопротивления;

ρ – плотность рабочей жидкости;

f_i – площадь поперечного сечения, по которому рассчитывается коэффициент гидравлического сопротивления.

Линейные потери давления в канале определяются по справочным данным [4, 6], либо аналитически по формуле Дарси-Вейсбаха:

$$\Delta p_{\lambda} = \lambda \frac{L_K}{d_K} * \frac{\rho}{2 f_K^2} q^2, \quad (10)$$

где λ – коэффициент сопротивления;

L_K, d_K, f_K – соответственно длина, диаметр и площадь поперечного сечения канала;

q – расход жидкости в канале.

Моделирование работы гидросистемы секций механизированной крепи заключается в моделировании совместной работы N гидроцилиндров, количество которых зависит как от конструкции и технологии работы комплекса, так и от моделируемой технологической операции.

Выражение для определения сопротивления перемещению гидродомкратов секции крепи на основании проведенных исследований [8]:

$$R_j = \frac{1}{n_{\Sigma}} \left\{ G \cos \alpha + \sum_{i=n_2+1}^{n_3} (P_{2i} F_{1i} - P_{3i} F_{2i}) \eta * \frac{\sin \beta}{\cos(\beta_{\psi} - \gamma_{\Delta})} \right\} f_n + \left\{ \sum_{i=1}^{n_1} (P_{2i} F_{1i} - P_{3i} F_{2i}) + \sum_{i=n_2+1}^{n_3} [(P_{2i} F_{1i} - P_{3i} F_{2i}) K_{конс}] - G_{верх} \right\} (f_n + f_{кр}) + \left\{ G_{кон} (\sin \alpha - \cos \alpha f_n) * \frac{l_Q(l_A + l_B)}{l_A^2 + l_B^2} (1 + \frac{L_R}{L_b}) - 0.5 P_p \frac{l_b}{L_b} + \sum_{i=n_2+1}^N P_{2i} F_{1i} K_{ci} \right\} * f_{TP} + R_{упл.}, \quad (11)$$

где $K_{конс}$ – коэффициент связи усилия в гидроцилиндре и реакции управляемой консоли на кровлю пласта;

$R_{упл.}$ – сила трения в уплотнениях поршня и штока;

j – номер гидродомкрата передвигающего; $j = n_2 + 1 \div n_3$;

$G_{\text{верх.}}$ – вес верхняка и других перемещаемых в процессе разгрузки элементов конструкции секции крепи.

Сопротивление перемещению гидростоек при их разгрузке равно:

$$R_{j=1 \div n_1} = R_{\text{упл.}} - (G_{\text{верх.}} - R_{\text{кр}} - R_n) \frac{1}{n_{\text{с}}}, \quad (12)$$

где $R_{\text{кр}}, R_n$ – реакции кровли и почвы, действующие до момента отрыва верхняка от кровли.

При распоре гидростоек сопротивление перемещению равно:

$$R_{j=1 \div n_1} = R_{\text{упл.}} + G_{\text{верх.}}. \quad (13)$$

После контакта верхняка с кровлей происходит повышение усилия поддержания кровли до усилия начального распора. Тогда:

$$R_{j=1 \div n_1} = R_{\text{упл.}} + G_{\text{верх.}} + R_{\text{кр}} + R_n. \quad (14)$$

Фронтальное выдвижение конвейера включает в себя большое количество одновременно работающих гидродомкратов. Сопротивление перемещению одного гидродомкрата равно:

$$R_{j=n_2+1 \div n_3} = R_{\text{упл.}} + G_{\text{кон.}} \cos \alpha f_n. \quad (15)$$

Предложенная математическая модель отличается от ранее разработанных тем, что в ней учтены полученные выражения для определения сопротивления перемещению гидроцилиндров при передвижении секций механизированной крепи на наклонных пластах, и это позволяет решать задачи как прямых гидравлических расчетов, при которых известно потокораспределение в системе и требуется определить рациональные значения конструктивных параметров гидравлических элементов, так и обратных, когда известны конструктивные и технологические параметры механизированной крепи, а необходимо промоделировать динамические процессы в гидросистеме и определить время выполнения различных операций. Исследование статических и динамических характеристик отдельных гидроэлементов и гидросистем в целом позволит выявить факторы, оказывающие решающее влияние на режим работы крепи, и обосновать допущения, которые могут быть приняты в инженерных расчетах. Результатом исследований являются также модели отдельных гидроэлементов механизированной крепи, которые в различных сочетаниях используются для расчета на ПЭВМ вариантов гидросистем проектируемых крепей.

На рисунках 2–7 представлены графики изменения параметров секции крепи и ее гидросистемы, построенные по результатам экспериментальных исследований и данным моделирования.

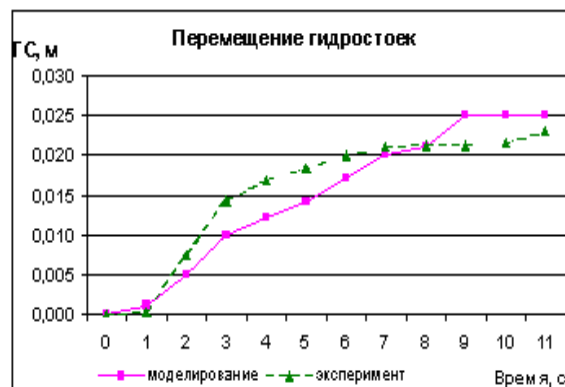


Рисунок 2 – Перемещение гидростоек



Рисунок 3 – Перемещение секций крепи



Рисунок 4 – Давление в поршневой полости гидростоек

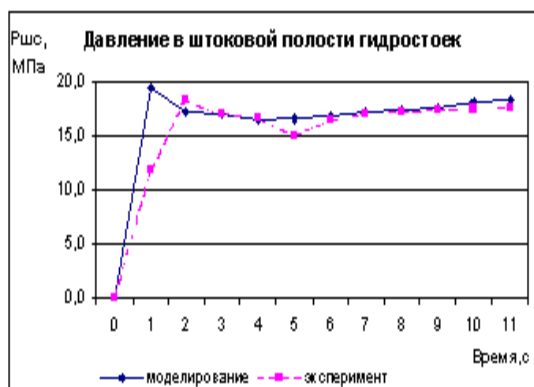


Рисунок 5 - Давление в штоковой полости гидростоек

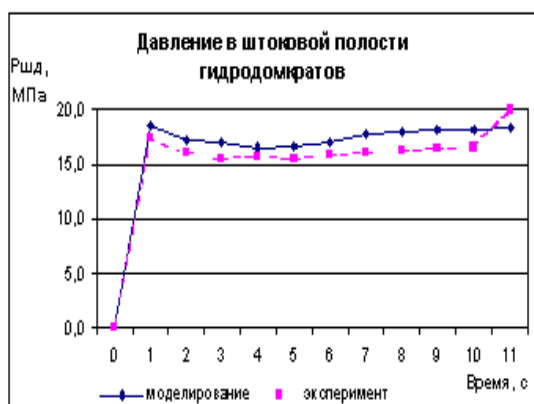


Рис. 6

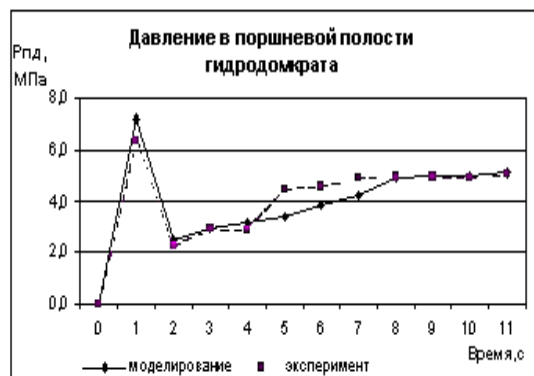


Рисунок 7 - Давление в штоковой полости гидродомкратов

Их анализ подтверждает адекватность модели для исследования процессов изменения параметров при передвижении секций механизированной крепи.

Выводы

В результате проведенных исследований разработана модифицированная математическая модель гидросистемы секций механизированной крепи, которая отличается от ранее

применявшихся тем, что в ней учтены полученные выражения для определения сопротивления перемещению гидроцилиндров при передвижении секций механизированной крепи на наклонных пластах.

Экспериментальными исследованиями подтверждена достоверность теоретических результатов и предложенных математических моделей. Расхождение между теоретическими и экспериментальными данными с учетом погрешности измерения параметров не превышает 7-15 %.

Разработанная математическая модель гидросистемы механизированной крепи, а также модели ее отдельных гидроэлементов приняты институтами ДонУГИ и Донгипроуглемаш для включения в состав математического обеспечения системы автоматизированного проектирования гидропривода механизированных крепей (САПГМК).

Литература

1. Пономаренко Ю.Ф., Баландин А.А. Инженерная методика проектного расчета параметров гидросистемы механизированных крепей. – М.: ИГД им. А.А. Скочинского, 1981. – 22 с.
2. Расчет и конструирование гидроприводов механизированных крепей / Ю.Ф. Пономаренко, А.А. Баландин, И.Т. Богатырев и др. // Под общ. ред. Ю.Ф. Пономаренко. – М.: Машиностроение, 1961. – 327 с.
3. Хандрос А.Х., Молчановский Е.Г. Динамика и моделирование гидроприводов станков. – М.: Машиностроение, 1969. – 156 с.
4. Альтшуль А.Д. Гидравлические сопротивления. – М.: Недра, 1982. – 368 с.
5. Тесленко А.И. Основы гидравлических расчетов механизированных крепей. – М.: Недра, 1974. – 216 с.
6. Идельчик И.Е. Справочник по гидравлическим сопротивлениям. – М.: Машиностроение, 1975. – 559 с.
7. РД 42.027-84 Крепи механизированные. Расчет устойчивости. – М.: Гипроуглемаш, 1964. – 56 с.
8. Математическое моделирование и расчет параметров систем угольных шахт и забойного оборудования: Монография / С.С. Гребенкин, В.Н. Павлыш, А.В. Агафонов, В.В. Косарев, В.В. Радченко, В.Д. Рябичев, В.П. Глебов, В.И. Зензеров, А.И. Егурнов. – Донецк: ВИК, 2007. – 263 с.

Павлыш В.Н., Зензеров В.И. Исследование параметров гидросистемы механизированных крепей методом математического моделирования. Рассматривается модифицированная математическая модель для исследования и расчета параметров гидросистемы секций механизированных крепей, обеспечивающих их устойчивую работу при выемке наклонных пластов.

Ключевые слова: Исследование, модификация, параметр, математическая модель, гидросистема.

Pavlysh V.N., Zenzerov V.I. The investigation of mechanical timbers hydrosystems parameters by mathematical modeling method. The modified mathematical model for researching and calculation of parameters of the hydrosystem for provided of stability of the sections of mechanized supports on inclined coal stratum are considered.

Keywords: investigation, modification, parameter, mathematical model, hydrosystem.

Статья поступила в редакцию 20.09.2017
Рекомендована к публикации д-ром физ.-мат. наук А.С. Миненко

УДК 621.746

Эмулятор системы диагностики теплового состояния кристаллизатора МНЛЗ в среде LabVIEW.

А.Ю. Харитонов, А.Б. Бирюков, А.В. Алиференко, А.В. Сныткина

Донецкий национальный технический университет, г. Донецк

кафедра компьютерного моделирования и дизайна

ant.kharytonov@gmail.com

Харитонов А.Ю., Бирюков А.Б., Алиференко А.В., Сныткина А.В. Эмулятор системы диагностики теплового состояния кристаллизатора МНЛЗ в среде LabVIEW. В данной статье исследована проблема диагностики и мониторинга тепловых процессов в кристаллизаторе машины непрерывного литья заготовок. Создана модель системы диагностики теплотехнических процессов в кристаллизаторе в среде LabVIEW.

Ключевые слова: машина непрерывного литья заготовок, тепловые процессы, кристаллизатор, LabVIEW.

Введение

В настоящее время вопросы тепловой работы кристаллизаторов МНЛЗ остаются актуальными. Одним из способов решения проблемы являются диагностика и мониторинг. Целью статьи является создание программы для диагностики и мониторинга тепловых процессов в кристаллизаторе машины непрерывного литья заготовок.

Постановка и реализация задачи

Одним из важнейших элементов, определяющих рациональную работу сталеразливочного комплекса и качество непрерывнолитой заготовки, является кристаллизатор. Применяются кристаллизаторы различного типа для сортовых и слябовых МНЛЗ, однако общая идея работы этого узла состоит в том, что во внутреннюю полость заливается жидкий металл из промежуточного ковша, а теплота, которая выделяется при охлаждении расплава отводится через металлическую стенку [1, 2]. Для сортовых МНЛЗ применяют гильзовые кристаллизаторы, основным элементом которых является медная гильза (рис. 1) [3].

В рамках концепции эффективной толщины газового зазора качестве ключевых величин целесообразно использовать среднюю плотность теплового потока с поверхности заготовки в кристаллизаторе ($\bar{q}$), среднее значение коэффициента теплопередачи в кристаллизаторе ($\bar{k}$), средний коэффициент теплоотдачи от корки заготовки к внутренней поверхности гильзы кристаллизатора ($\bar{\alpha}$), эффективную толщину газового зазора ($\delta_{эф}$).

Ниже приведен пример расчетного определения усредненных значений названных величин для исходных данных. Сечение разливаемой заготовки составляет 120x120 мм, а длина кристаллизатора – 750 мм [4].

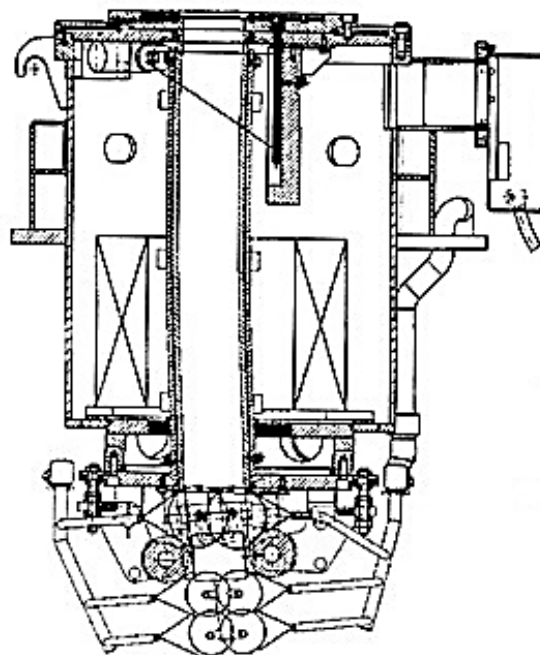


Рисунок 1 – Чертеж кристаллизатора сортовой МНЛЗ

Значение средней плотности теплового потока может быть определено из уравнения усредненного теплового баланса кристаллизатора:

$$G \cdot c \cdot \Delta t = \bar{q} \cdot F, \quad (1)$$

где G - массовый расход воды через кристаллизатор, кг/с;

c - теплоемкость воды, кДж/(кг·К);

Δt - перепад температур охлаждающей воды в кристаллизаторе, °С;

F - поверхность контакта заготовки и кристаллизатора, м².

$$\bar{q} = \frac{G \cdot c \cdot \Delta t}{F} = \frac{1700}{60} \cdot 4,2 \cdot 10^3 \cdot 6,325 = 2007133 \frac{\text{Вт}}{\text{м}^2}$$

С точки зрения закона теплопередачи средняя плотность теплового потока может быть выражена как

$$\bar{q} = \bar{k} \cdot (t_s - t_w^{\text{ав}}), \quad (2)$$

где t_s - температура солидуса для разливаемой марки стали, °С;

$t_w^{\text{ав}}$ - средняя температура охлаждающей воды в кристаллизаторе, °С.

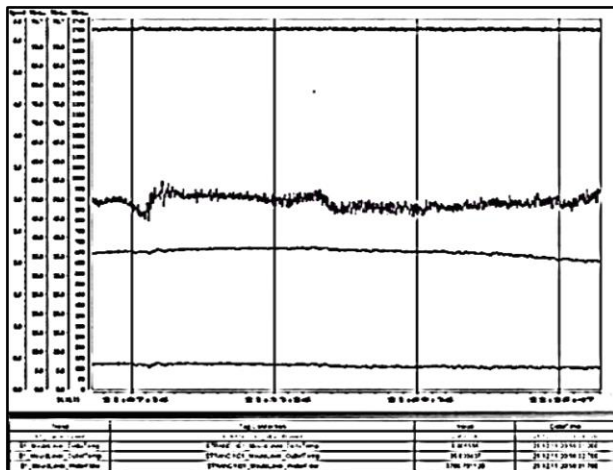


Рисунок 2 – Изменение во времени величин, характеризующих тепловую работу кристаллизатора

Поскольку в качестве движущей силы теплопередачи в выражении (2) взята разница температур от солидуса до средней температуры охлаждающей воды, значение $\bar{k}$ представляет собой «проводимость» тепловой цепи, включающей в себя термические сопротивления передаче тепла через твердую корку, через газовый зазор, через стенку гильзы кристаллизатора и от ее наружной поверхности к охлаждающей воде.

Определив среднюю плотность теплового потока $\bar{q}$ с поверхности заготовки в кристаллизаторе из уравнения теплового баланса (1), среднее значение коэффициента

теплопередачи $\bar{k}$ в кристаллизаторе рассчитывается:

$$\bar{k} = \frac{\bar{q}}{(t_s - t_w^{\text{ав}})}$$

$$\bar{k} = \frac{2007133}{(1480 - 33)} = 1387,1 \frac{\text{Вт}}{\text{м}^2 \cdot \text{К}}$$

Используя с некоторым допущением зависимости, справедливые для стационарного теплообмена, можно записать зависимость среднего коэффициента теплопередачи в кристаллизаторе от всех термических сопротивлений

$$\bar{k} = \frac{\bar{q}}{\frac{1}{\alpha_w} + \frac{\delta_m}{\lambda_m} + \frac{1}{\bar{\alpha}} + \frac{\delta_k}{\lambda_k}}, \quad (3)$$

где α_w – коэффициент теплоотдачи от внешней поверхности гильзы кристаллизатора к воде, Вт/(м²·К);

δ_m – толщина стенки гильзы кристаллизатора, м;

λ_m – коэффициент теплопроводности материала гильзы кристаллизатора, Вт/(м·К);

δ_k – средняя толщина корки металла в кристаллизаторе, м;

λ_k – коэффициент теплопроводности корки заготовки при ее средней температуре, Вт/(м·К).

При помощи уравнения (3) определяется средний коэффициент теплоотдачи от корки заготовки к внутренней поверхности гильзы кристаллизатора:

$$\bar{\alpha} = \frac{1}{\frac{1}{\bar{k}} - \frac{1}{\alpha_w} - \frac{\delta_m}{\lambda_m} - \frac{\delta_k}{\lambda_k}}, \quad (4)$$

$$\bar{\alpha} = \frac{1}{\frac{1}{1387,1} - \frac{1}{5000} - \frac{0,01}{360} - \frac{0,007}{34}} = 3487,1 \frac{\text{Вт}}{\text{м}^2 \cdot \text{К}}$$

Эта величина несет интегральную информацию о тепломеханических процессах, протекающих в кристаллизаторе. Анализ ее значений для кристаллизаторов различной конусности позволит уточнить представления о механизме теплообмена в полости кристаллизатора. Так, зная величину $\bar{\alpha}$, можно определить значение эффективной толщины газового зазора между коркой заготовки и внутренней поверхностью гильзы кристаллизатора ($\delta_{\text{эф}}$). Поскольку при использовании величины эффективной толщины газового зазора для изучения теплообмена в кристаллизаторе предполагается, что тепловой поток от поверхности заготовки формируется

только по двум механизмам: излучением и теплопроводностью через газовый зазор, будет получено следующее выражение для определения $\delta_{эф}$:

$$\delta_{эф} = \frac{\lambda_3}{(\bar{\alpha} - \alpha_l)}, \quad (5)$$

где λ_3 – теплопроводность газовой прослойки, Вт/(м·К);

α_l – коэффициент лучистой теплоотдачи от корки заготовки к внутренней поверхности гильзы кристаллизатора, Вт/(м²·К).

Величина α_l определяется на основании известной зависимости, описывающей лучистый теплообмен:

$$\alpha_l = \frac{C_{пр} \cdot \left[\left(\frac{T_{п}}{100} \right)^4 - \left(\frac{T_{ст}}{100} \right)^4 \right]}{T_{п} - T_{ст}}, \quad (6)$$

где $T_{п}$ – температура поверхности заготовки, К;

$T_{ст}$ – температура внутренней поверхности гильзы кристаллизатора, К;

$C_{пр}$ – приведенный коэффициент излучения при лучистом теплообмене между поверхностью заготовки и внутренней поверхностью гильзы кристаллизатора.

Значения $T_{п}$ и $T_{ст}$ могут быть с достаточно небольшой погрешностью определены на основании зависимостей, справедливых для стационарной теплопередачи через ряд последовательно включенных термических сопротивлений:

$$T_{п} = t_w^{av} + \bar{q} \cdot \left(\frac{1}{\alpha_w} + \frac{\delta_m}{\lambda_m} + \frac{1}{\bar{\alpha}} \right) + 237, \quad (7)$$

$$T_{ст} = t_w^{av} + \bar{q} \cdot \left(\frac{1}{\alpha_w} + \frac{\delta_m}{\lambda_m} \right) + 237.$$

$$T_{п} = 33 + 2007133 \cdot \left(\frac{1}{5000} + \frac{0,01}{360} + \frac{1}{3487,1} \right) + 237 = 1340 \text{ К}$$

$$T_{ст} = 33 + 2007133 \cdot \left(\frac{1}{5000} + \frac{0,01}{360} \right) + 237 = 763 \text{ К}$$

$$\alpha_l = \frac{5,6 \cdot 0,62 \cdot \left[\left(\frac{1340}{100} \right)^4 - \left(\frac{763}{100} \right)^4 \right]}{1340 - 763} = 173,59 \frac{\text{Вт}}{\text{м}^2 \cdot \text{К}}$$

$$\delta_{эф} = \frac{0,09}{(3487,1 - 173,59)} = 2,72 \cdot 10^{-5} \text{ м}$$

Так, численно показано, что с помощью предложенных зависимостей (1-7) в рамках системы АСУ ТП в режиме реального времени можно определять изменение значений всех заявленных величин ($\bar{q}$, $\bar{k}$, $\bar{\alpha}$, $\delta_{эф}$) в зависимости

от таких аргументов как перепад температуры охлаждающей воды в кристаллизаторе, расход первичной воды, теплофизические характеристики разливаемой стали, геометрические параметры гильзы кристаллизатора (рис. 3) [4].

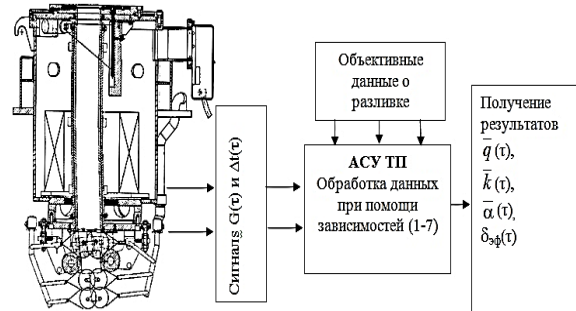


Рисунок 3 – Схема получения информации об изменении величин, дающих углубленное представление о тепловых процессах, которые протекают в кристаллизаторе, в режиме реального времени

Разработка программного продукта

LabVIEW (англ. Laboratory Virtual Instrumentation Engineering Workbench) — это среда разработки и платформа для выполнения программ, созданных на графическом языке программирования «G» фирмы National Instruments (США) [5].

Благодаря мощной системе проектирования LabVIEW появляется возможность построить любую измерительную или управляющую систему за кратчайшее время. В отличие от универсальных инструментов, LabVIEW тесно интегрирует любое оборудование с обширным анализом и библиотеками обработки сигналов, предлагает пользовательские графические пользовательские интерфейсы и позволяет развертывать эти системы на платформе, которая использует новейшие и самые передовые технологии [6].

Программное обеспечение для проектирования систем LabVIEW интегрируется практически с любым оборудованием любого поставщика в одной среде и помогает экономить время разработки благодаря удобству функций и согласованной структуре программирования на всех аппаратных средствах [7].

В LabVIEW возможно использование всех измерительных и управляющих аппаратных средств в одной среде разработки. LabVIEW позволяет интегрирование с различными типами приборов, шин и датчиков. Например, LabVIEW интегрируется с различными устройствами сбора

данных, настольными приборами, модульными приборами и FPGA [8].

С помощью программного обеспечения для проектирования систем LabVIEW имеется возможность быстро разработать пользовательские интерфейсы для визуализации данных и ввода оператора.

Можно интегрировать графические, текстовые и другие подходы к программированию в единой среде, чтобы эффективно создавать собственные программные решения. LabVIEW взаимодействует с Python, ANSI C, .NET и др.

Программа LabVIEW называется виртуальным прибором и состоит из двух частей: блок-диаграммы, описывающей логику работы виртуального прибора и лицевой панели, описывающей внешний интерфейс виртуального прибора. Виртуальные приборы могут использоваться в качестве составных частей для построения других виртуальных приборов [9].

Лицевая панель виртуального прибора содержит средства ввода-вывода, кнопки, переключатели, светодиоды, шкалы и т.п. Они служат для управления виртуальным прибором.

Блок-диаграмма содержит функциональные узлы, являющиеся источниками и средствами обработки данных. Также компонентами блок-диаграммы являются терминалы и управляющие структуры. Функциональные узлы и терминалы объединены в единую схему линиями связей.

Код, написанный на LabVIEW, автоматически масштабируется для работы с несколькими ядрами при перемещении в многоядерный CPU без внесения каких-либо изменений в код и без ручного распределения потоков [10].

Особенностью LabVIEW является создание дистрибутива, в который включено программное обеспечение для корректной работы программы.

В данной среде была разработана программа «Krystallizator», в которой эмулируются сигналы расхода охлаждающей воды в кристаллизаторе (G) и перепада температур на входе и на выходе из кристаллизатора (Δt), а также на основании заданных параметров объекта (рис. 4) рассчитываются средняя плотность теплового потока с поверхности заготовки в кристаллизаторе ($\bar{q}$), среднее значение коэффициента теплопередачи в кристаллизаторе ($\bar{k}$), средний коэффициент теплоотдачи от корки заготовки к внутренней поверхности гильзы кристаллизатора ($\bar{\alpha}$), эффективная толщина газового зазора ($\bar{\delta}_{эф}$) (рис. 5).

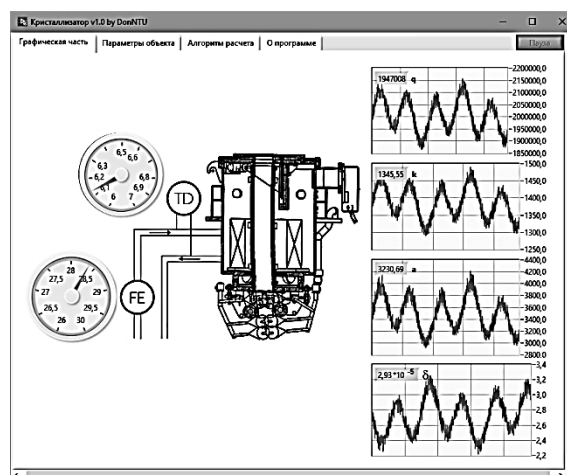


Рисунок 4 – Вкладка «Параметры объекта» на фронтальной панели программы «Krystallizator»

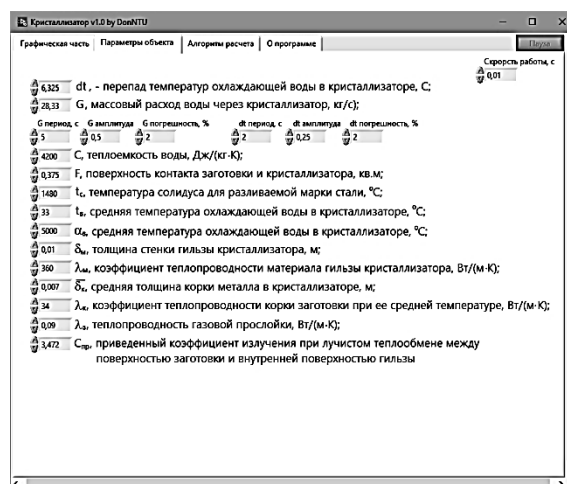


Рисунок 5 – Вкладка «Графическая часть» на фронтальной панели программы «Krystallizator»

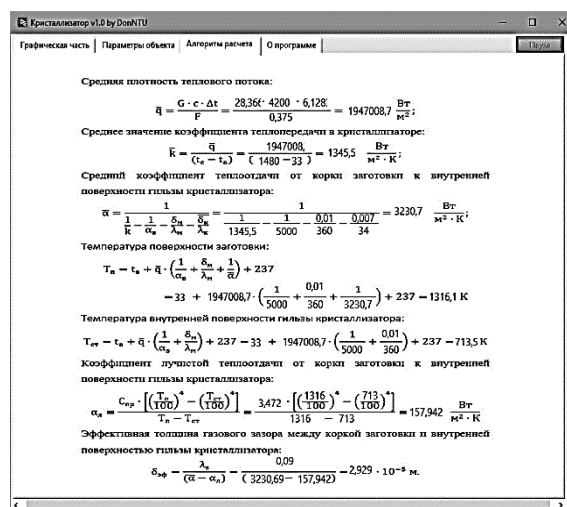


Рисунок 6 – Вкладка «Алгоритмы расчета» на фронтальной панели программы «Krystallizator»

Также на вкладке «Алгоритмы расчета» можно увидеть расчет данных параметров, а также расчет температуры поверхности заготовки (T_n), температуры внутренней поверхности гильзы кристаллизатора ($T_{ст}$), коэффициента лучистой теплоотдачи от корки заготовки к внутренней поверхности гильзы кристаллизатора (α_n) (рис. 6). На блок-диаграмме данной программы используются управляющие структуры такие, как условный оператор «Case Structure», оператор цикла «While loop».

Оператор цикла «While Loop» используется для того, чтобы полученный программный продукт не переставал производить вычисления и работал в режиме реального времени. Для эмуляции сигналов расхода охлаждающей воды в кристаллизаторе (G) и перепада температур на входе и на выходе (Δt) используется функция «Basic Function Generator». Для отображения на фронтальной панели нескольких вкладок используется «Tab Control». Управляющие элементы, в которые заносятся исходные данные, называются «Numeric Control». На блок-диаграмме они расположены в левом верхнем углу. Получаемые значения, которые отображаются на вкладке «Алгоритмы расчета» на фронтальной панели сохраняются в «Numeric Indicator». Блок-диаграмма программы представлена на рис. 7.

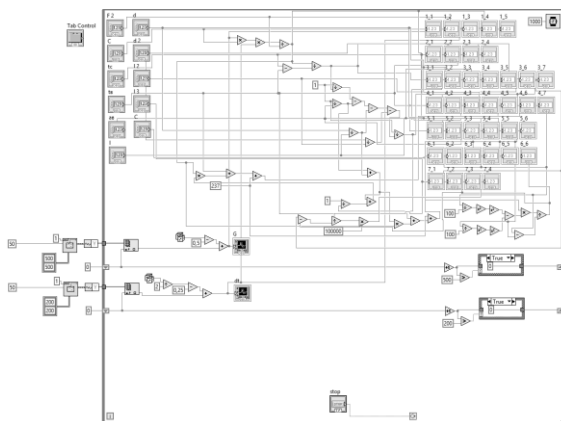


Рисунок 7 – Блок-диаграмма программы «Krystallizator»

Выводы

Проанализированы подходы к диагностике тепловых процессов в кристаллизаторе. В качестве величин, на основании которых ведется диагностика предложено выделить: среднюю плотность теплового потока с поверхности заготовки в кристаллизаторе, среднее значение коэффициента теплоотдачи в кристаллизаторе, средний коэффициент теплоотдачи от корки заготовки к внутренней поверхности гильзы

кристаллизатора и эффективную толщину газового зазора.

Создан эмулятор системы диагностики теплотехнических процессов в кристаллизаторе при помощи программного обеспечения LabVIEW.

Список литературы

1. Васильева А.В. Совершенствование подходов к диагностике тепловых процессов в кристаллизаторе МНЛЗ / А.В. Васильева // Техника и технологии машиностроения : материалы V Междуна. студ. науч.-практ. конф. (Омск, 4–10 апреля 2016 г.) / Минобрнауки России, ОмГТУ ; [редкол.: Е. Н. Еремин (отв. ред.) и др.]. – Омск : Изд-во ОмГТУ, 2016. – 441 с.
2. Васильева А.В. Совершенствование подходов к диагностике тепловых процессов в кристаллизаторе МНЛЗ / А.В. Васильева, А.Б. Бирюков // Металлургия XXI столетия глазами молодых / Материалы II Международной научно-практической конференции студентов. – Донецк: ДонНТУ, 2016. – 306 с.
3. Смирнов А.Н. Непрерывная разливка стали: Учебник / Смирнов А.Н., Куберский С.В., Штепан Е.В. – Донецк: ДонНТУ, 2011. – 482 с.
4. Васильева А.В. Совершенствование подходов к диагностике тепловых процессов в кристаллизаторе МНЛЗ / А.В. Васильева, А.Б. Бирюков // Межвузовский сборник научных работ студентов. Тепломассообмен – Луганск: Издательство Луганского национального университета имени Владимира Даля, 2016. – 76 с.
5. Джеффри Т. LabVIEW для всех, с программой LabVIEW 7 Express. – М.: ДМК Пресс, 2004. – 350 с.
6. Суранов А.Я. LabVIEW 7: справочник по функциям. – М.: ДМК, 2005. – 376 с.
7. Пейч Л.И., Точилин Д.А., Поллак Б.П. LabVIEW для новичков и специалистов. – М.: Горячая линия-Телеком, 2004. – 384 с.
8. Барри Патон. LabVIEW: Основы Аналоговой Электроники. //Перевод В.В. Шаркова. – СПб.: БХВ, 2004. – 340 с.
9. Барри Патон. LabVIEW: Основы Цифровой Электроники. //Перевод В.В. Шаркова. – СПб.: БХВ, 2004. – 262 с.
10. Загидуллин Р.Ш. LabVIEW в исследованиях и разработках. – М.: Горячая линия-Телеком, 2005. – 420с.

A.Y. Kharitonov, A.B. Birukov, A.V. Aliferenko, A.V. Vasileva. Emulator of diagnostical system of thermal condition of CCM crystallizer in LabVIEW environment. In this article, the problem of diagnosing and monitoring of thermal conditions in the crystallizer of continuous casting machine is investigated. A model of diagnostic system of thermal engineering processes in a crystallizer, emulated in LabVIEW environment, is created.

Keywords: continuous casting machine, thermal processes, crystallizer, LabVIEW.

Харитонов А.Ю., Бирюков А.Б., Алиференко А.В., Сныткина А.В. Эмулятор системы диагностики теплового состояния кристаллизатора МНЛЗ в среде LabVIEW. В данной статье исследована проблема диагностики и мониторинга тепловых процессов в кристаллизаторе машины непрерывного литья заготовок. Создана модель системы диагностики теплотехнических процессов в кристаллизаторе в среде LabVIEW.

Ключевые слова: машина непрерывного литья заготовок, тепловые процессы, кристаллизатор, LabVIEW

Статья поступила в редакцию 10.09.2017
Рекомендована к публикации доктором технических наук В.Н. Павлышом

УДК 004.056.5

Подсистема защиты авторского права в сфере программного обеспечения

А.В. Чернышова, Д.В. Кубашевский
Донецкий национальный технический университет
chernyshova.alla@rambler.ru, dehax12@gmail.com

Чернышова А.В., Кубашевский Д.В. Подсистема защиты авторского права в сфере программного обеспечения.

В данной работе раскрыта проблема нелегального использования ПО, описаны недостатки существующих механизмов защиты авторского права на программное обеспечение и представлена реализация новой программной подсистемы защиты ПО с использованием криптографических алгоритмов шифрования.

Введение

В настоящее время широко распространена проблема нелегального копирования и распространения программного обеспечения. Ежедневно огромное количество пиратских копий программных продуктов скачиваются бесплатно в сети Интернет, что наносит огромный ущерб разработчикам этих продуктов.

В последнее время разработчики коммерческого ПО внедряют в свои программные продукты подсистемы защиты от нелегального копирования. Зачастую, подобные подсистемы защиты ненадёжны, так как её разработке уделяется недостаточно сил. Через некоторое время хакеры исследуют и взламывают подсистему защиты, и программный продукт попадает на рынок пиратского ПО [1]. В связи с этим возникает необходимость реализации подсистемы защиты таким образом, чтобы программный продукт не мог функционировать без подтверждения лицензии на использование ПО, а процесс взлома такой защиты был максимально затруднён.

Постановка задачи

Необходимо разработать программную подсистему защиты авторского права в программном обеспечении, которая представляла бы компромиссный вариант между усилиями, затраченными на её реализацию, и эффективностью её функционирования. Для реализации этой задачи необходимо пересмотреть принципы и методы защиты от анализа и взлома ПО.

Обычно проверка лицензии на использование ПО пользователем основана на

активации программного продукта через сервер лицензий. После проверки правильности данных лицензии, предоставленных пользователем, сервер должен дать подтверждение действительности лицензии, и в этом случае подсистема защиты открывает доступ пользователю к функциональным возможностям ПО. Данная идея защиты содержит существенный недостаток — можно реализовать поддельный сервер лицензий, который всегда будет подтверждать действительность лицензии, или модифицировать подсистему защиты таким образом, чтобы проверка активации программного продукта не осуществлялась.

Поэтому была разработана подсистема защиты, которая основана не на предоставлении права на функционирование ПО, а на предоставлении некоторых функциональных возможностей ПО, то есть некоторой части программного кода, без которой продукт не может работать и выполнять важные для пользователя функции. Модификация подсистемы защиты или реализация пиратского сервера лицензий не позволяет получить доступ к необходимым функциям программы.

Структура программной системы

Разработанная программная система имеет клиент-серверную архитектуру. Диаграмма развёртывания представлена на рисунке 1.

Клиентскую часть представляет простейшее тестовое приложение «DLSTestSoft», которое содержит и взаимодействует с модулем подсистемы защиты «DLS». Подсистема защиты выполняет запросы к серверу лицензий «DLSServer». Запросы от клиентской части к серверу будут выполняться посредством TCP-соединения [2].

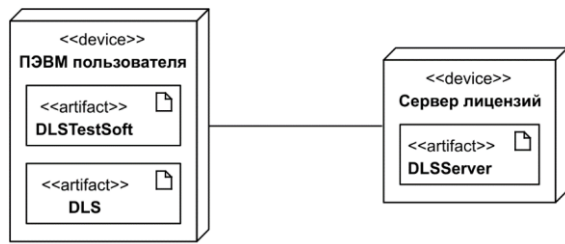


Рисунок 1 — Диаграмма развёртывания

Диаграмма вариантов использования (диаграмма прецедентов) для сервера приведена на рисунке 2.



Рисунок 2 — Диаграмма вариантов использования сервера

Сервер хранит выданные пользователям лицензии на использование ПО в локальной базе данных. Разработчик может выполнять регистрацию новых лицензий. При необходимости в базу данных добавляются записи о новых пользователях и версиях ПО, для каждой версии заполняется список файлов программы, которые будут проверяться на наличие изменений в них.

Среди функций, доступных пользователю сервера — разработчику ПО, обязательными являются запуск и остановка сервера, а также возможность сохранить запись о лицензии в файл для передачи его пользователю защищаемого ПО.

Диаграмма вариантов использования программной системы в целом представлена на рисунке 3.

На диаграмме отображено взаимодействие подсистемы защиты с сервером. При запуске защищаемого программного обеспечения подсистема защиты выполняет попытку проверки и активации лицензии. После успешной активации подсистема защиты включает полнофункциональный режим работы программы.

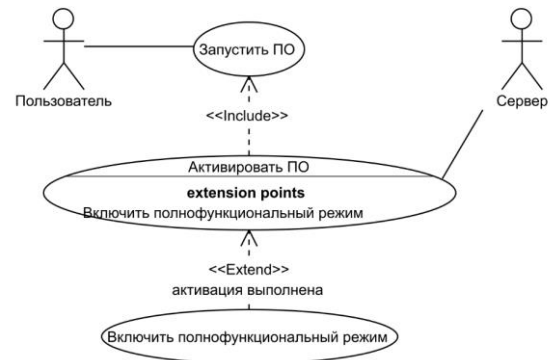


Рисунок 3 — Диаграмма вариантов использования программной системы

Модуль подсистемы защиты отправляет серверу данные лицензии, сервер проверяет правильность данных и наличие в своей базе данных записи о выданной лицензии. После успешной проверки сервер отправляет клиенту зашифрованный файл, который предоставляет дополнительный функционал защищаемому ПО. Клиент получает данный файл, и подсистема защиты переводит программу в полнофункциональный режим, которая сможет использовать дополнительный функционал, предоставленный сервером после успешной активации лицензии.

Проектирование структур системы

Сервер подсистемы защиты взаимодействует с базой данных, которая имеет определённую структуру. В базе данных хранится информация о каждой выданной лицензии, затем разработчик ПО имеет возможность сохранить информацию о лицензии в файл и передать его пользователю ПО для активации лицензии при работе с программным продуктом.

Схема базы данных сервера лицензий представлена на рисунке 4.

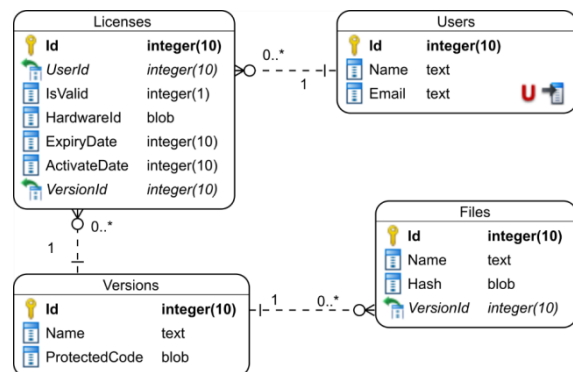


Рисунок 4 — Схема базы данных

База данных содержит 4 таблицы. Таблица «Users» содержит записи о пользователях защищаемого программного обеспечения.

Таблица «Versions» содержит записи о версиях защищаемого программного обеспечения. Наличие записей о версии программы позволяет контролировать процесс активации лицензии. Лицензия, выданная для одной версии, не действительна для работы активации другой версии.

Таблица «Files» содержит записи о файлах ПО, которые будут проверяться на наличие изменений при активации лицензии.

Таблица «Licenses» содержит записи о выданных пользователям лицензиях на использование ПО.

Файл лицензии на использование ПО имеет формат XML [3]. Пример содержимого файла лицензии приведён на рисунке 5.

```

1 <DLS>
2 <Id>1</Id>
3 <Name>Денис Кубашевский</Name>
4 <Email>dehax12@gmail.com</Email>
5 <HardwareId>BK3vkmrkgpCg6T6PD2Hfg=</HardwareId>
6 <Version>1.0.0</Version>
7 <Files>
8 <File Name="log4net.dll" Hash="9ktzPq5EYfZhtWg8r8A==" />
9 </Files>
10 <ActivateDate>2012-01-01</ActivateDate>
11 <ExpiryDate>2020-01-01</ExpiryDate>
12 <Signature>805ucg4CqgTrkVLuz+ICQt8UB8qUrgk0GZd618Df1Dc0Nzq3R9HUSQ=</Signature>
13 <PublicKey>
14 BgTAAAIABAEU1MwAAQAAET7nUo66aPF71YsmksTj+RExEicYH-Aj3pWGAHYhj+AvZcCehs5QbJE1
    UxeLmYxCKQngy17P4drpmqizuybuFOR0KQ8bRnK2+VZUzolkpGAmP/3PKpoIs5WtbbT2vTIg/b
    Odhz8xbBUqnd8/1XcCim/0oorepxdSyrbHP8/dtE9tPgKqv0E316dBm+r84f5hdqppYPGHYAs03
    czAhEeumM5ic97AB62TPMBL1h5pBY2EXb5jkk640Znv+r71JrvIdTOB7g5aw6/VuXbMhMsdcsN12
    Rv6oTF0o08jUDd9fy4u2J9v8oaOp29ub1BxmSaRCFng1ph5pdxNF59QF0HdFPKqR/UXT/s070P5K61s
    RURpxvQAW9S5pTv0ZSh8JG45L4zsqy7p1hYr7FuUzxtMwCjZc410nARZ48aTkkFYBLRKH53U2bQ8q1
    4Lso+7vU0uoHmbT2jks3yU7GUB1B2zxcEvSyC1v6IA+3w4qCKnUup6cm9ELK0P2U2wJk6jukfqrdX
    gshTc/NLicR2mzKuEAA0oe56E7T90VMdcnMyZLF3ntwEH</PublicKey>
14 </DLS>

```

Рисунок 5 — Пример содержимого файла лицензии

Корневой тег «DLS» включает в себя другие XML-теги, которые содержат данные лицензии.

Программная реализация

Для реализации программной системы была выбрана программная платформа «.NET Framework» [4] версии 4.6.1 и высокоуровневый объектно-ориентированный язык программирования C# [5] версии 7.0. В качестве интегрированной среды разработки (IDE) использовалась «Microsoft Visual Studio» версии 2017.

Для реализации базы данных выбрана компактная встраиваемая реляционная БД «SQLite» [6].

Основной задачей реализации программной системы является обеспечение функционирования алгоритма подсистемы защиты авторского права. Полный алгоритм активации лицензии представлен на рисунке 6.

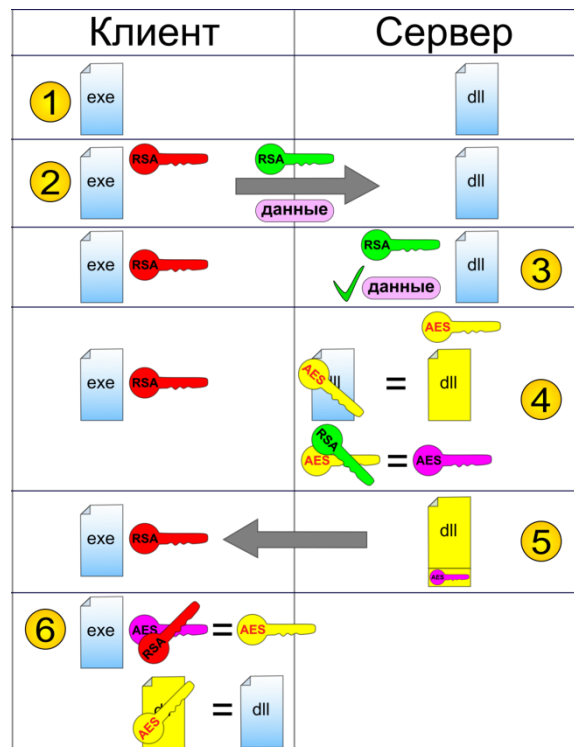


Рисунок 6 — Алгоритм активации лицензии [7]

После успешной проверки файла лицензии начинается отправка данных лицензии на сервер для прохождения проверки и получения данных «защищённого кода». На втором шаге алгоритма модуль подсистемы защиты генерирует пару ключей RSA [8] и отправляет на сервер открытый ключ вместе с данными лицензии.

На третьем шаге сервер подтверждает подлинность и действительность данных, после чего на четвёртом шаге генерирует ключ AES [9] и с помощью его зашифровывает файл «защищённого кода». Ключ AES зашифровывается с помощью полученного от клиента открытого ключа RSA.

На пятом шаге сервер передаёт обратно клиенту зашифрованный «защищённый код» и зашифрованный ключ, который клиент на шестом шаге расшифровывает с помощью своего оставшегося закрытого ключа RSA. После этого расшифрованным ключом AES расшифровывается файл «защищённого кода» и модуль подсистемы защиты сохраняет в памяти данные «защищённого кода», что позволяет перевести ПО в полнофункциональный режим.

Алгоритм проверки и активации лицензии модуля подсистемы защиты представлен на рисунке 7. При запуске тестового приложения происходит обращение к модулю подсистемы защиты с целью проверки лицензии и выполнения активации лицензии.

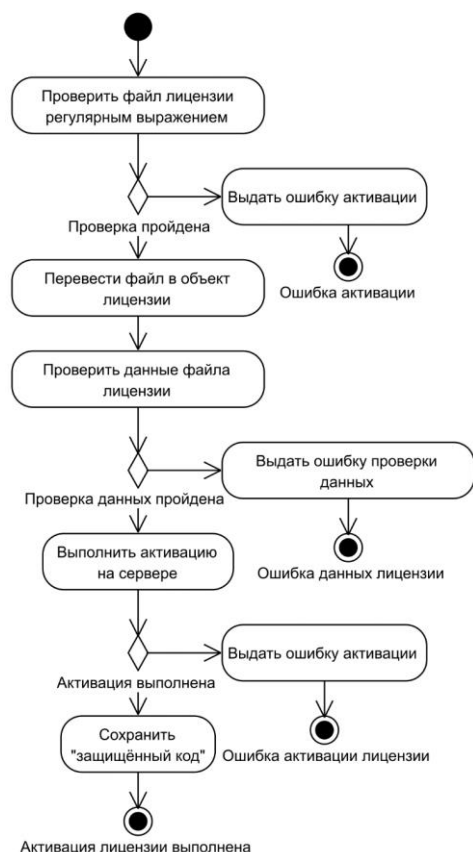


Рисунок 7 — Диаграмма деятельности алгоритма проверки лицензии

Диаграмма деятельности алгоритма запуска тестового приложения представлена на рисунке 8.

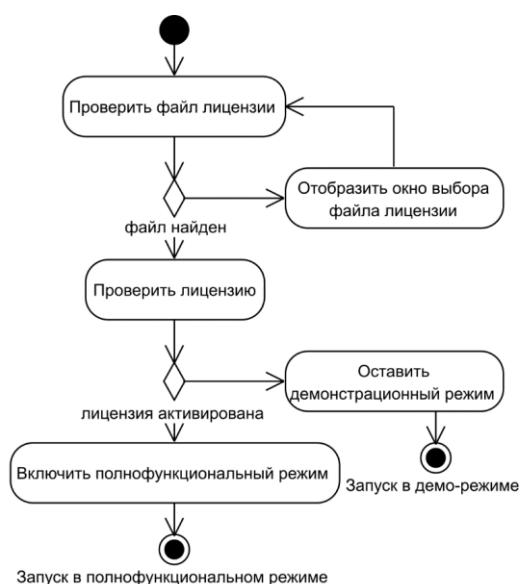


Рисунок 8 — Диаграмма деятельности алгоритма запуска тестового приложения

Пример использования системы

После запуска приложения сервера появится основное окно, которое изображено на рисунке 9.

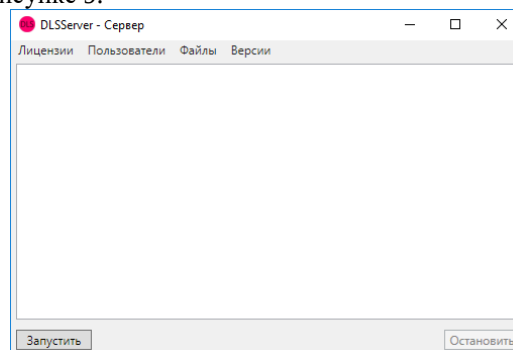


Рисунок 9 — Основное окно приложения сервера

Для запуска сервера, который будет ждать запросов от клиентов, необходимо нажать кнопку «Запустить». После запуска сервера кнопка «Запустить» будет заблокирована и доступной окажется кнопка «Остановить» для остановки функционирования сервера.

Главное меню окна позволяет перейти к окнам просмотра записей таблиц базы данных.

После первого запуска тестового приложения пользователю будет предложено выбрать файл лицензии, который был предоставлен разработчиком ПО при покупке программного продукта. При последующих запусках указывать файл лицензии пользователю не потребуется, в процессе первой активации файл лицензии будет сохранён в каталог приложения в личной папке пользователя. В среде операционной системы Windows путь к данному каталогу — «%LOCALAPPDATA%\DehaxSoft\DLSTestSoft».

После приобретения пользователем программного продукта разработчик должен выдать лицензию на использование ПО. Для этого необходимо в основном окне приложения сервера выбрать пункт меню «Лицензии», чтобы перейти к окну просмотра записей таблицы лицензий БД. Данное окно представлено на рисунке 10.

Для выдачи новой лицензии необходимо нажать кнопку «Добавить», после чего появится окно добавления новой записи о лицензии в БД. Для добавления новой записи о лицензии потребуется, чтобы предварительно были добавлены:

- запись о пользователе, которому выдаётся лицензия;
- запись о версии ПО, для которой будет выдана лицензия;
- записи о файлах ПО, которые подлежат проверке при активации ПО.

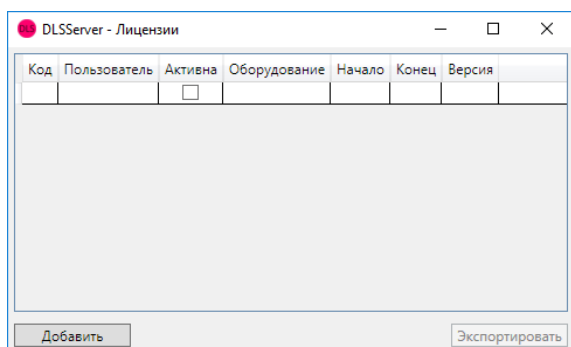


Рисунок 10 — Список выданных лицензий

Добавление этих записей осуществляется подобным образом. Доступ к таблицам предоставляется из главного меню основного окна приложения сервера. В каждом из окон для просмотра записей таблиц пользователей, версий ПО и файлов ПО также присутствует кнопка «Добавить» для создания новых записей БД.

Окно добавления новой записи о лицензии на использование ПО изображено на рисунке 11.

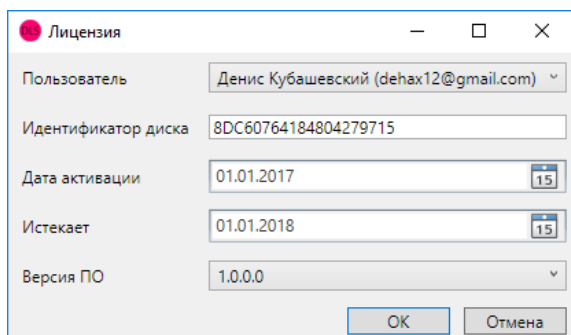


Рисунок 11 — Выдача новой лицензии

В поле «Пользователь» разработчик может выбрать пользователя из списка, которому будет выдана лицензия. В поле «Идентификатор диска» указывается серийный номер первого жёсткого диска ПЭВМ пользователя. Поля «Дата активации» и «Истекает» позволяют контролировать срок действия лицензии. В поле «Версия ПО» из списка также можно выбрать версию программного продукта, для которого предназначена лицензия.

После нажатия кнопки «OK» запись о лицензии будет сохранена в базе данных. Теперь необходимо сгенерировать для пользователя файл лицензии, с помощью которого он сможет выполнить активацию. Для этого в окне списка лицензий необходимо выбрать соответствующую запись о лицензии и нажать кнопку «Экспортировать». Файл лицензии с именем адреса электронной почты пользователя и расширением «dls» будет помещён в папку «Документы» операционной системы. Далее

разработчик передаёт файл пользователю, после чего у пользователя появляется лицензия на использование ПО.

При первом запуске тестового приложения отобразится сообщение о том, что файл лицензии не найден и появится окно, которое изображено на рисунке 12, позволяющее пользователю указать файл лицензии для проведения активации.

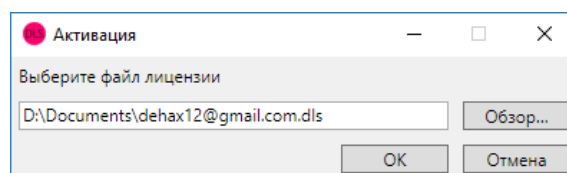


Рисунок 2 — Окно выбора файла лицензии

После нажатия кнопки «OK» будет произведена проверка файла лицензии, затем модулем подсистемы защиты будет выполнен запрос на сервер с передачей данных лицензии. При успешном выполнении активации отобразится основное окно тестового приложения, которое изображено на рисунке 13.

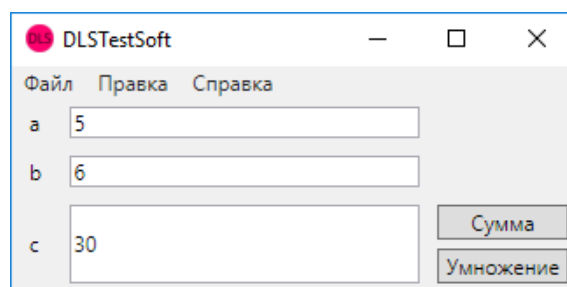


Рисунок 13— Полнофункциональный режим работы тестового приложения

На рисунке 13 представлен полнофункциональный режим работы тестового приложения, в который подсистема защиты перешла после успешного прохождения процедуры активации лицензии.

Если Интернет-соединение на этапе активации отсутствует, пользователю отображается соответствующее сообщение с предложением повторить попытку активации, которое представлено на рисунке 14.

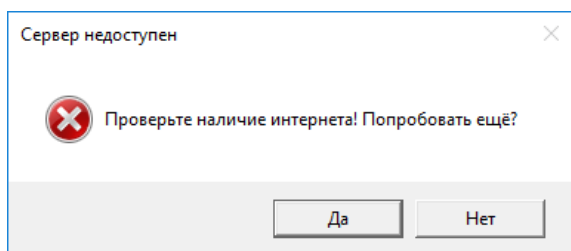


Рисунок 14 — Сообщение об отсутствии соединения с сервером

После нажатия кнопки «Нет» тестовое приложение продолжит работу в демонстрационном режиме, который остаётся активным также в случае возникновения ошибок при активации лицензии. Основное окно тестового приложения в демонстрационном режиме работы представлено на рисунке 15.

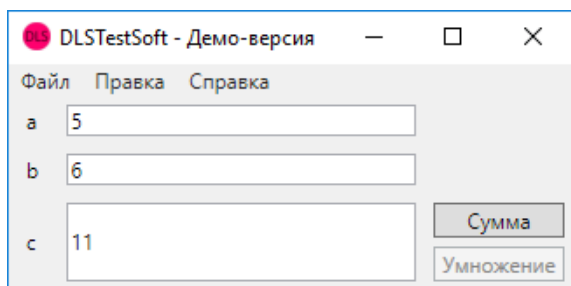


Рисунок 15 — Демонстрационный режим работы приложения

В демонстрационном режиме работы тестового приложения доступна только кнопка «Сумма», которая позволяет выполнить операцию сложения значений полей «а» и «б», получив результат в поле «с».

В полнофункциональном режиме работы доступна также операция умножения и отсутствует надпись «демо-версия» в заголовке окна, что подтверждает успешную активацию лицензии и право на использование ПО.

Выводы

В ходе выполнения данной работы была раскрыта проблема нелегального использования ПО, описаны недостатки существующих механизмов защиты авторского права на программное обеспечение и представлена реализация новой программной подсистемы защиты ПО с использованием криптографических алгоритмов шифрования.

Защита, обеспечиваемая компонентами данной программной системы, позволяет противостоять действиям хакеров среднего уровня, поскольку исследование алгоритма защиты не позволит взломать защиту.

Использование длинных ключей криптографических алгоритмов позволит исключить метод перебора для выполнения расшифровки защищаемых данных.

В дальнейшем планируется расширение реализованной подсистемы защиты до системы защиты ПО, которая будет работать с несколькими программными продуктами и поддерживать распределённые базы данных [10] для оптимизации взаимодействия модуля системы защиты с серверами лицензий и уменьшения на них нагрузки при большом количестве пользователей защищаемого ПО.

Литература

1. Пиратское ПО // SecurityLab.ru. [Электронный ресурс]. – Режим доступа: <https://www.securitylab.ru/news/tags/%EF%E8%F0%E0%F2%F1%EA%EE%E5+%CF%CE/>
2. Протокол TCP // OpenNET. [Электронный ресурс]. – Режим доступа: https://www.opennet.ru/docs/RUS/inet_book/4/44/tcp_443.html
3. Extensible Markup Language (XML) // W3C. [Электронный ресурс]. – Режим доступа: <https://www.w3.org/XML/>
4. .NET // Microsoft. [Электронный ресурс]. – Режим доступа: <https://www.microsoft.com/net>
5. .NET Programming Languages // Microsoft. [Электронный ресурс]. – Режим доступа: <https://www.microsoft.com/net/Learn/Languages>
6. SQLite // SQLite. [Электронный ресурс]. – Режим доступа: <https://www.sqlite.org/>
7. Чернышова, А.В. Подсистема защиты авторского права в программном обеспечении / А.В. Чернышова, Д.В. Кубашевский // Информатика и кибернетика. – 2016. – № 2(4). – С. 68-72.
8. Алгоритм RSA // НОУ ИНТУИТ. [Электронный ресурс]. – Режим доступа: <http://www.intuit.ru/studies/courses/691/547/lecture/12391>
9. Как устроен AES // Хабрахабр. [Электронный ресурс]. – Режим доступа: <https://habrahabr.ru/post/112733/>
10. Распределённые и параллельные системы баз данных // CIT Forum. [Электронный ресурс]. – Режим доступа: http://citforum.ru/database/classics/distr_and_parallel_sdb/

A.V. Chernyshova, D.V. Kubashevskiy. *The Copyright Protection Subsystem in Software.* The article deals with the problem of illegal software using. It describes the disadvantages of existing mechanisms for the copyright protection of software and introduces the realization of the new software protection subsystem which uses cryptographic encryption algorithms.

Keywords: Software license, additional module, software protection, symmetric encryption algorithms, asymmetric encryption algorithms

А.В. Чернышова, Д.В. Кубашевский. *Подсистема защиты авторского права в сфере программного обеспечения.* В данной работе раскрыта проблема нелегального использования ПО, описаны недостатки существующих механизмов защиты авторского права на программное обеспечение и представлена реализация новой программной подсистемы защиты ПО с использованием криптографических алгоритмов шифрования.

Ключевые слова: Лицензия на программное обеспечение, вспомогательный модуль, защита ПО, симметричные алгоритмы шифрования, ассиметричные алгоритмы шифрования

Статья поступила в редакцию 17.10.2017
Рекомендована к публикации доктором технических наук В.Н. Павлышом

CONTENT

Andryukhin A.I. Reflective representations	4
Andryukhin A.I., Novikov E.T. Computer Problem Solving. Associative search and reflexive inference in cognitive systems based on SQL and NoSQL DBMS.I.....	13
Artemenko VA, Andryukhin A.I., Filer Z.E. Computer estimates of the characteristics of multidimensional operations and objects. II	27
Belkov D.V. File allocation algorithms in computer networks.....	39
Efimenko K.N. Optimization of the compositional control unit with optimum addressing microinstructions.	45
Helen Kazakova, Anna Kolomytseva. Model basis the assessment of interaction risk in architecture affiliate network.....	52
Lesnov E.V., Andryukhin A.I. A software system for evaluating the quality of tuning a string instrument.	57
Makogon S.A., Sitnikova O.D. Development of software for the formation of optimal routes for fast delivery of goods.....	63
Pavlysh V.N., Zenzerov V.I. The investigation of mechanical timbers hydrosystems parameters by mathematical modeling method.	71
Kharitonov A.Y., Birukov A.B., Aliferenko A.V. , Vasileva A.V. Emulator of diagnostical system of thermal condition of CCM crystallizer in LabVIEW environment.....	78
Chernyshova A.V., Kubashevskiy D.V. The Copyright Protection Subsystem in Software.....	84
Content.....	91

Электронное периодическое издание

Научный журнал

ИНФОРМАТИКА И КИБЕРНЕТИКА

(на русском, английском языках)

№ 4(10)-2017

Ответственный за выпуск А.И. Андрюхин

Технические редакторы Т.В. Завадская, А.В. Звягинцева

Компьютерная верстка Д.А. Филипишин

Подписано к выпуску 22.12.2017. Усл. печ. лист. 10. Уч.- изд. лист. 7,1.

Адрес редакции: ДНР, 83001, г. Донецк, ул. Артема, 58, ГОУ ВПО «ДонНТУ», 5-й учебный корпус, к. 425. Тел.: +38 (062) 301-08-51 E-mail: infcyb.donntu@yandex.ru, URL: <http://infcyb.donntu.org>