Московский Государственный Университет
имени М.В. Ломоносова
Российская Академия Наук
Международная Академия Технологических Наук
Российская Академия Естественных Наук

Интеллектуальные Системы.

Теория и приложения

ТОМ **22** ВЫПУСК 4 * **2018**

МОСКВА

ISSN 2411–4448

УДК 519.95; 007:159.955
ББК 32.81

Издается с 1996 г.*

В журнале «Интеллектуальные системы. Теория и приложения» публикуются научные достижения в области теории и приложений интеллектуальных систем, новых информационных технологий и компьютерных наук.

Издание журнала осуществляется под эгидой МГУ имени М. В. Ломоносова, Научного Совета по комплексной проблеме «Кибернетика» РАН, Отделения «Математическое моделирование технологических процессов» МАТН, Секции «Информатики и кибернетики» РАЕН.

Учредитель журнала: ООО «Интеллектуальные системы».

Журнал входит в список изданий, включенных ВАК РФ в реестр публикаций материалов по кандидатским и докторским диссертациям по математике и механике.

Спонсором издания является:

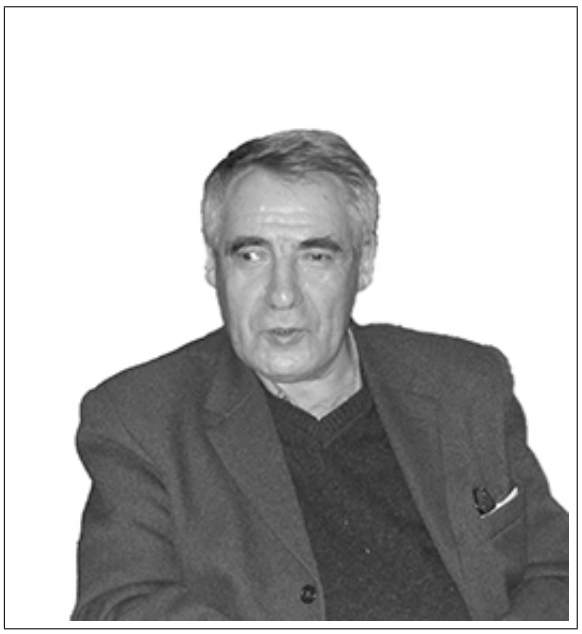
ООО «Два Облака»
Разработка корпоративных информационных систем
http://www.dvaoblaka.ru

Индекс подписки на журнал: 64559 в каталоге НТИ «Роспечать».

Адрес редакции: 119991, Москва, ГСП-1, Ленинские Горы, д. 1, механико-математический факультет, комн. 12-01.
Адрес издателя: 115230, Россия, Москва, Хлебозаводский проезд, д. 7, стр. 9, офис 9. Тел. +7 (495) 939-46-37, e-mail: mail@intsysjournal.org

*) Прежнее название журнала: «Интеллектуальные системы».

ОГЛАВЛЕНИЕ



9 января 1941 г. - 25 ноября 2018 г. Памяти Вячеслава Александровича Буевича

Редакционный совет журнала «Интеллектуальные системы. Теория и приложения» с глубоким прискорбием извещает, что 25 ноября после тяжелой болезни на 77 году жизни скончался выдающийся ученый, Заслуженный профессор Московского университета, доктор физико-математических наук, профессор кафедры интеллектуальных систем

ВЯЧЕСЛАВ АЛЕКСАНДРОВИЧ БУЕВИЧ

Вячеслав Александрович Буевич родился 9 января 1941 года в городе Смоленске. Окончил механико-математический факультет МГУ в 1965 году. Уже в сту-

денческие годы проявились его разносторонние дарования. Им была решена задача построения минимального универсального автомата, и этот пример вошел впоследствии в учебники. Он был участником спектаклей студенческого театра МГУ. После окончания факультета работал в институте Академии наук и учился в аспирантуре АН СССР. В 1973 году защитил кандидатскую диссертацию "Об А-полноте для автоматов"и начал работать на механико-математическом факультете МГУ имени М.В. Ломоносова. В 1992 году он защитил докторскую диссертацию "Решение проблемы τ-полноты для автоматов".

В.А.Буевич вошел в историю отечественной науки как крупный ученый, талантливый математик, автор фундаментальных работ в области теории автоматов и дискретных функций. Вячеслав Александрович глубоко разработал новое направление в теории функциональной полноты для автоматов - аппроксимационная полнота, им построен простейший А-универсальный автомат, доказана алгоритмическая неразрешимость проблемы А-полноты для автоматов. Найденное им решение проблемы t-полноты для автоматов в результате привело к новому доказательству критерия полноты для функций k-значной логики. Эти результаты вошли в учебники, широко известны мировой научной общественности и получили заслуженное признание. Он является автором более 40 научных работ, среди которых 3 монографии.

В.А. Буевич был замечательным педагогом. На его глубоко продуманных и тонко выстроенных курсах выросло не одно поколение высококвалифицированных математи-

ков. Он читал на механико-математическом факультете МГУ курсы лекций, среди которых «Дискретные системы и процессы», «Теория алгоритмов», «Вопросы полноты для конечнозначных ограниченно-детерминированных функций», руководил работой спецсеминаров. Под научным руководством В.А. Буевича десятки студентов защитили дипломные работы; шесть его учеников стали кандидатами наук. Подготовленные им специалисты успешно работают в самых различных сферах жизни, от государственной службы до бизнеса. В.А. Буевич входил в состав ГАК, участвовал в работе приемной комиссии.

В.А. Буевич являлся членом редколлегии нашего журнала «Интеллектуальные системы» с момента его создания, входил в состав оргкомитетов многих научных конференций..

Вячеслав Александрович был одним из организаторов семинара «Наука и культура», его общественная деятельность, активная гражданская позиция обеспечили ему авторитет и уважение ученых и деятелей культуры России.

Редакционная коллегия нашего журнала, вся научная общественность выражают соболезнования родным и близким покойного. Память о замечательном человеке, ученом и гражданине сохранится в наших сердцах.

Часть 1.
Общие проблемы теории интеллектуальных систем

Требования, принципы, практика создания отечественных аппаратно-программных платформ для автоматизированных систем в защищенном исполнении критической информационной инфраструктуры Российской Федерации

Михалевич И.Ф.

В работе изложена система требований и принципов, определяющих методологические аспекты создания отечественных аппаратно-программных платформ для автоматизированных систем в защищенном исполнении как основы критической информационной инфраструктуры Российской Федерации, описан опыт создания защищенной аппаратно-программной платформы «Синтез-АПП», в полном объеме удовлетворяющей самым строгим требованиям по информационной безопасности, надежности, масштабируемости, обеспечивающей независимость критической информационной инфраструктуры от зарубежных технологий и программ.

Ключевые слова: автоматизированная система в защищенном исполнении, аппаратно-программная платформа, информационная безопасность, критическая информационная инфраструктура, «Синтез-АПП»

1. Введение.

Бурное развитие информационных и телекоммуникационных технологий и систем сопровождается столь же бурным развитием технологий и средств нарушения информационной безопасности, направленных на совершение киберпреступлений и ведение кибервойн. Несанкционированное раскрытие информации, ее искажение или недоступность могут

привести к катастрофическим последствиям как для отдельных стран, так и миропорядка в целом, особенно, если эта информация касается функционирования критических инфраструктур, включающих объекты атомной и гидроэнергетики, здравоохранения, связи, оборонной промышленности, государственной власти и др. [1]. Поэтому укрепление позиций на мировой арене, построение цифровой экономики и реализация иных прорывных направлений развития страны невозможны без повышения защищенности критической информационной инфраструктуры Российской Федерации (далее – КИИРФ).

Доктриной информационной безопасности Российской Федерации [2], Стратегией развития информационного общества в Российской Федерации на 2017 - 2030 годы [3], Программой «Цифровая экономика Российской Федерации» [4] повышение защищенности КИИРФ неразрывно связано с ликвидацией зависимости отечественной промышленности от зарубежных информационных технологий и средств обеспечения информационной безопасности, повышением безопасности функционирования элементов информационной инфраструктуры, обеспечением безопасности информации, обрабатываемой в автоматизированных и информационных системах и передаваемой по сетям электросвязи, входящим в состав КИИРФ.

Принципиальной особенностью КИИРФ является то, что в ней обрабатывается информация от открытой до содержащей сведения, составляющие государственную тайну. Но и открытая информация КИИРФ должна быть надежно защищена, так как нарушение ее доступности и/или несанкционированное изменение (удаление) может повлечь критические последствия в социальной, политической, экономической и иных критически важных областях деятельности.

Данные условия существенно влияют на платформенные решения для КИИРФ, относят объекты КИИРФ к категории автоматизированных систем в защищенном исполнении, т.е. систем, реализующих информационные технологии выполнения установленных функций в соответствии с требованиями стандартов и/или иных нормативных документов по защите информации [5].

2. Методологические аспекты создания аппаратно - программных платформ для критической информационной инфраструктуры РФ

Создание аппаратно-программных платформ (далее – АПП) КИИРФ должно основываться на ключевых требованиях и принципах, обеспечивающих заданные уровни безопасности информации, информацион-

ной безопасности используемых информационных технологий и объектов КИИРФ в целом. Под безопасностью информации будем понимать состояние ее защищенности, при котором обеспечены ее конфиденциальность, доступность и целостность, а под безопасностью информационной технологии - состояние защищенности информационной технологии, при котором обеспечивается выполнение изделием, реализующим информационную технологию, предписанных функций без нарушений безопасности обрабатываемой информации [6]. По аналогии с [6] под информационной безопасностью объекта КИИРФ будем рассматривать состояние его защищенности, при котором на объекте КИИРФ обеспечивается безопасность информации и автоматизированных средств ее обработки.

С учетом положений [2] - [4], требований тактико-технических заданий на НИОКР по созданию автоматизированных систем в защищенном исполнении, отечественного опыта по обеспечению доверенной среды их функционирования [7, 8] АПП для КИИРФ должна обладать следующими основными свойствами и соответствовать следующим основным общим требованиям [9, 10]:

• полноценность – свойство, отражающее полноту состава и технологий платформы, обеспечивающих создание (модернизацию) и функционирование объектов КИИРФ различного назначения, разных классов защищенности, уровней топологической и архитектурной сложности;

• технологическая независимость и независимость от импорта (импортонезависимость) – свойство платформы сохранять полноценность, заявленные характеристики, развиваться, поддерживаться независимо от внешнеполитических и внешнеэкономических факторов, без применения импортных компонентов, без иностранного участия, принудительного обновления компонентов и управления из-за рубежа, передачи информации, в том числе технологической, за пределы РФ;

• промышленный уровень – свойство платформы сохранять производительность, отказоустойчивость и другие заявленные характеристики объектов КИИРФ при сложных топологии и архитектуре, высоких нагрузках и больших объемах данных в течение всего срока эксплуатации;

• универсальность - свойство платформы обеспечивать на основе собственных базовых компонентов создание (модернизацию) сегментов (объектов) КИИРФ различного назначения, разных классов защищенности и уровней топологической и архитектурной сложности;

• гарантии развития и поддержки – свойство платформы развиваться и обеспечивать эксплуатацию, обслуживание и модернизацию созданных на ее основе объектов КИИРФ.

К основным функциональным требованиям к АПП можно отнести следующие. Платформой, в частности:

• должно обеспечиваться создание (модернизация) и эксплуатация объектов КИИРФ разных классов защищенности, безопасное ведомственное, межведомственное, корпоративное взаимодействие объектов КИИРФ и их взаимодействие с зарегистрированными пользователями;

• должна создаваться защищенная (доверенная) среда функционирования специального программного обеспечения (далее – СПО), разработанного для конкретного сектора (сегмента и т.п.) критической инфраструктуры, обеспечиваться безопасность информации, исходя из класса защищенности объекта КИИРФ;

• должно обеспечиваться условие «мягкой» поэтапной модернизации объектов КИИРФ, предполагающее их функционирование при замене средств вычислительной техники и иного оборудования на новое;

• должно обеспечиваться условие «мягкой» поэтапной модернизации СПО, предполагающее использование СПО, ранее введенного в эксплуатацию;

• должны обеспечиваться организация производительного, устойчивого, масштабируемого вычислительного процесса, надежного хранения больших объемов информации, сохранение ее конфиденциальности, доступности и целостности;

• должна обеспечиваться поддержка основных сетевых служб системного и пользовательского уровней;

• должны обеспечиваться сбор, обработка и хранение данных в территориально распределенных сегментах КИИРФ, возможность безопасного удаленного доступа к этим данным (в установленном порядке), поддержка технологий интеграции вычислительных ресурсов и систем хранения данных, строительство (модернизация) и эксплуатация центров обработки данных;

• должна обеспечиваться поддержка многоуровневости и одновременной работы множества пользователей с одними и теми же данными баз (банков) данных, публикация, поиск, доступ к данным, управление контентом, резервирование и архивирование данных, синхронизация обновлений;

• должен обеспечиваться контроль и управление функционированием всех устройств, комплексов средств автоматизации (информатизации), программно-технических комплексов и т.п., входящих в состав объектов КИИРФ;

• должно обеспечиваться резервирование основных компонент объектов КИИРФ и содержащейся в них информации (данных);

• должна поддерживаться работа комплексов информационно-расчётных, аналитических, прогнозных задач, в том числе с применением Web-технологий обработки геопространственных данных и многоэкранного режима, текстовых, графических редакторов, обработка мультимедийной информации;

• должна поддерживаться доверенная среда разработки СПО.

К основным принципам, которые должны соблюдаться при создании отечественных АПП для КИИРФ, можно отнести следующие:

1. Принцип унификации программного обеспечения.

Проектирование программного обеспечения должно осуществляться таким образом, чтобы в структуре программ схожего назначения максимально применялись заранее учтенные (проверенные) функциональные модули (пакеты и т.п.).

2. Принцип локализации заимствованных программ (модулей, пакетов).

Заимствованные программные средства должны быть локализованы. Под локализацией понимается:

проверки отдельных модулей (пакетов) и программного обеспечения в целом на соответствие функциональному назначению, отсутствие недекларированных возможностей, совместимость с применяемыми в платформе и на объектах КИИРФ средствами защиты информации, невлияние на функционирование средств защиты информации;

доработки и иные мероприятия по приведению программного обеспечения в соответствие с требованиями национальных стандартов и нормативных документов российских регуляторов в сфере информационных технологий и информационной безопасности;

фиксация состояния локализованного программного обеспечения;

оформление программной, технической проектной, рабочей, конструкторской и эксплуатационной документации на локализованное программное обеспечение;

подготовка специалистов и техническая поддержка.

3. Принцип типизации технических решений.

Комбинированием компонент платформы должно обеспечиваться создание типовых технических решений (типовых конфигураций) для объектов КИИРФ различного назначения, разных классов защищенности и уровней топологической и архитектурной сложности.

4. Принцип масштабируемости.

Платформой должны обеспечиваться масштабируемость технических решений, их комплексирование в типовые конфигурации объектов КИИРФ. Платформа не должна быть чувствительной к топологической и архитектурной сложности объектов КИИРФ, численности пользователей, объемам обрабатываемой информации.

5. Принцип универсальности программных средств защиты информации.

Платформой должна обеспечиваться возможность изменения класса защищенности объекта КИИРФ путем изменения настроек программных средств защиты информации без их замены.

6. Принцип оптимизации ресурсов (кастомизации).

Платформой должна обеспечиваться возможность комбинирования технических решений и изменения настроек применяемых компонент под задачи конкретного объекта КИИРФ, снижение стоимости ее создания (модернизации) и владения.

7. Принцип программного доверия и наследования СПО.

Платформа должна обеспечивать возможность «погружения» в свою доверенную среду функционирования прикладных программ и СПО, заявленных владельцем (заказчиком) ведомственного (корпоративного) сегмента (объекта) КИИРФ, в том числе «наследуемых» из модернизируемых объектов КИИРФ.

8. Принцип «мягкой» модернизации.

Платформа должна позволять «мягкую» модернизацию ведомственных (корпоративных) сегментов (объектов) КИИРФ путем замены морально устаревшего оборудования и постепенного переноса существующих объектов КИИРФ на новую платформу без «останова» обслуживания пользователей.

9. Принцип аппаратного доверия.

Платформа должна содержать перечни рекомендованного оборудования для соответствующих классов защищенности и уровней сложности объектов КИИРФ.

10. Принцип динамичности научно-технического потенциала.

Применительно к платформе должны обеспечиваться возможности быстрого наращивания численности специалистов по ее компонентам, адаптации платформенных решений под характеристики сегментов (объектов) КИИРФ.

3.Аппаратно-программная платформа «Синтез-АПП»

В соответствии с тактико-техническими заданиями на инициативные ОКР серии «Синтез», согласованными с регулятором (ФГКУ «Войско-

вая часть 43753»), в 2012-2013 г.г. была разработана первая версия отечественной защищенной аппаратно-программной платформы «Синтез-АПП» (рис. 1), получены документы соответствия требованиям информационной безопасности при обработке открытой, конфиденциальной информации и информации с грифом секретности (до совершенно секретно включительно [11]), зарегистрирован торговый знак продуктов платформы – «СИНТЕЗАЙТИС» [12]. В 2013-2014 г.г. на платформе «Синтез-АПП» были построены первые десятки объектов КИИРФ, проведены обучение их пользователей и подготовка обслуживающего персонала, создана система технической поддержки, обеспечены условия развития платформы.

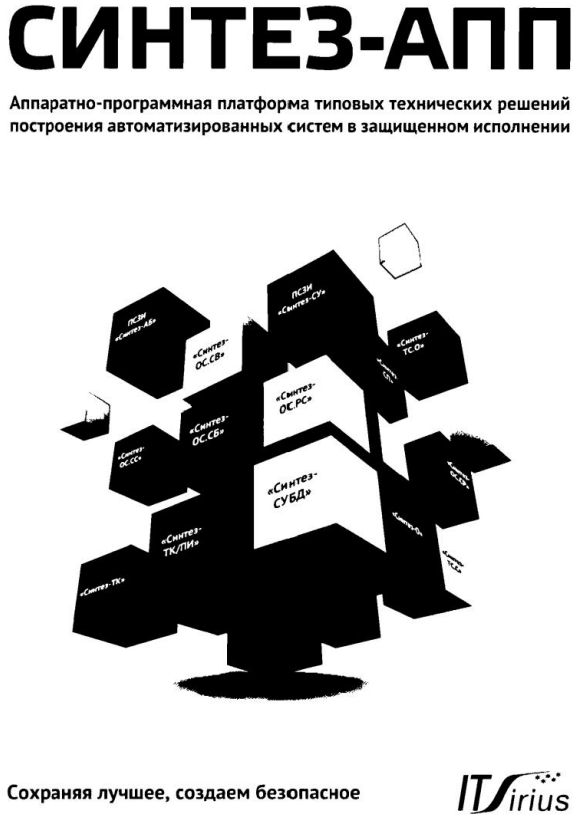


Рис. 1. Аппаратно-программная платформа «Синтез-АПП»

Решения по обеспечению полноценности платформы «Синтез-АПП»

Для обеспечения полноценности платформы «Синтез-АПП» базовый комплект программных средств разрабатывался в следующем составе: семейство защищенных операционных систем (для объектом малой, средней и большой сложности), защищенная система управления базами данных, сервер приложений, встроенные программные средства защиты информации и средства интеграции с внешними средствами защиты информации объектов КИИРФ, средства администрирования, разработки, офисный пакет и другие программы, перечисленные в таблице 1.

Таблица 1. Состав первичного комплекта программных средств платформы «Синтез-АПП»

Назначение изделия	Модификация и обозначение изделия
Операционные системы	серверная специальная «Синтез-ОС.СС»
	серверная вспомогательная «Синтез-ОС.СВ»
	серверная базовая «Синтез-ОС.СБ»
	для виртуальных машин и рабочих станций «Синтез-ОС.РС»
Сервер приложений	«Синтез-СП»
Терминальный сервер	специальный «Синтез-ТС.С»
	объединенный «Синтез-ТС.О»
Терминальный клиент	«Синтез-ТК»
Персональный идентификатор пользователя	«Синтез-ТК/ПИ»
Система управления базами данных	«Синтез-СУБД»
Офисные средства	«Синтез-О»
Программное средство защиты информации	Сервер управления «ПСЗИ «Синтез-СУ»»
	Агент безопасности «ПСЗИ «Синтез-АБ»»
Сервер хранения данных	«Синтез-СХД»
Сервер каталогов	«Синтез-СК»
Сервер обновлений	«Синтез-СО»
Сервер разработки приложений	«Синтез-РП»

Решения по обеспечению импорто- и технологической независимости платформы «Синтез-АПП»

Для достижения независимости платформы «Синтез-АПП» от импорта было принято решение о создании программных компонент на базе ядра Linux [13], их интеграции с локализованными заимствованными модулями открытого программного обеспечения (далее - открытое ПО), сертифицированными средствами защиты информации других российских компаний.

Для гарантированной технологической независимости сегментов (объектов) КИИРФ при проектировании платформы и построении системы технической поддержки были реализованы решения по созданию «воздушного зазора» (см. табл. 2). «Воздушный зазор» исключает возможность доступа к техническим средствам и информации (данным) объектов КИИРФ из-за пределов сегмента (объекта) КИИРФ, обеспечивает технологическое взаимодействие объектов КИИРФ только внутри сегмента КИИРФ и с разработчиком платформы «Синтез-АПП».

Таблица 2. Общая схема организации технической поддержки платформы «Синтез-АПП»

<table>
<tr><td rowspan="3">Объекты КИИРФ (запросы, получение ТП)</td><td colspan="6">Уровень проблем ТП</td></tr>
<tr><td>1-й</td><td>2-й</td><td>3-й</td><td>уровень ПСЗИ платформы «Синтез-АПП»</td><td>уровень ядра Linux, нелокализованного ПО</td></tr>
<tr><td colspan="3">Внутренний контур ТП</td><td>Внешний контур ТП</td></tr>
<tr><td></td><td colspan="2">БД(БЗ)ПлС</td><td>БД(БЗ)ПлС-ПСЗИ</td><td>БД(БЗ)НЛПО</td></tr>
<tr><td colspan="6">Оказание консультативной помощи (решение задач 1-го уровня сложности)</td></tr>
<tr><td>→</td><td>→</td><td></td><td></td><td></td><td></td></tr>
<tr><td>←</td><td>←</td><td></td><td></td><td></td><td></td></tr>
<tr><td colspan="6">Решение задач 2-го уровня сложности</td></tr>
<tr><td>→</td><td>→</td><td>→</td><td></td><td></td><td></td></tr>
<tr><td>←</td><td>←</td><td>←</td><td></td><td></td><td></td></tr>
<tr><td colspan="6">Текущее обслуживание (решение задач 3-го уровня сложности - обновление ПО и т.п.)</td></tr>
<tr><td>→</td><td>→</td><td>→</td><td>→</td><td></td><td></td></tr>
<tr><td>←</td><td>←</td><td>←</td><td>←</td><td></td><td></td></tr>
<tr><td colspan="6">Устранение ошибок программ и ПСЗИ «Синтез»</td></tr>
<tr><td>→</td><td>→</td><td>→</td><td>→</td><td>→</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td>←</td><td></td></tr>
<tr><td>←</td><td>←</td><td>←</td><td>←</td><td></td><td></td></tr>
<tr><td colspan="6">Устранение ошибок ядра Linux, нелокализованного ПО</td></tr>
<tr><td>→</td><td>→</td><td>→</td><td>→</td><td>→</td><td></td></tr>
<tr><td colspan="6">Воздушный зазор</td></tr>
<tr><td></td><td></td><td></td><td></td><td>→ → →</td><td>→ → →</td></tr>
<tr><td></td><td></td><td></td><td></td><td>← ← ←</td><td>← ← ←</td></tr>
<tr><td colspan="6">Воздушный зазор</td></tr>
<tr><td>←</td><td>←</td><td>←</td><td>←</td><td></td><td></td></tr>
</table>

Примечание.

БД(БЗ)ПлС – базы данных (базы знаний) ошибок (уязвимостей) программ платформы «Синтез-АПП», решений по их устранению.

БД(БЗ)ПлС-ПСЗИ – базы данных (базы знаний) ошибок (уязвимостей) программных средств защиты информации платформы «Синтез-АПП», решений по их устранению.

БД(БЗ)НЛПО – базы данных (базы знаний) ошибок (уязвимостей), выявленные в ядре Linux и нелокализованном ПО, решений по их устранению.

Решения по обеспечению промышленного уровня платформы «Синтез-АПП»

В целях быстрого достижения промышленного уровня платформы «Синтез-АПП» было принято решение о применении в качестве лока-

лизуемых программ продуктов компании Red Hat, как мирового лидера открытого ПО на основе ядра Linux (рис. 2).

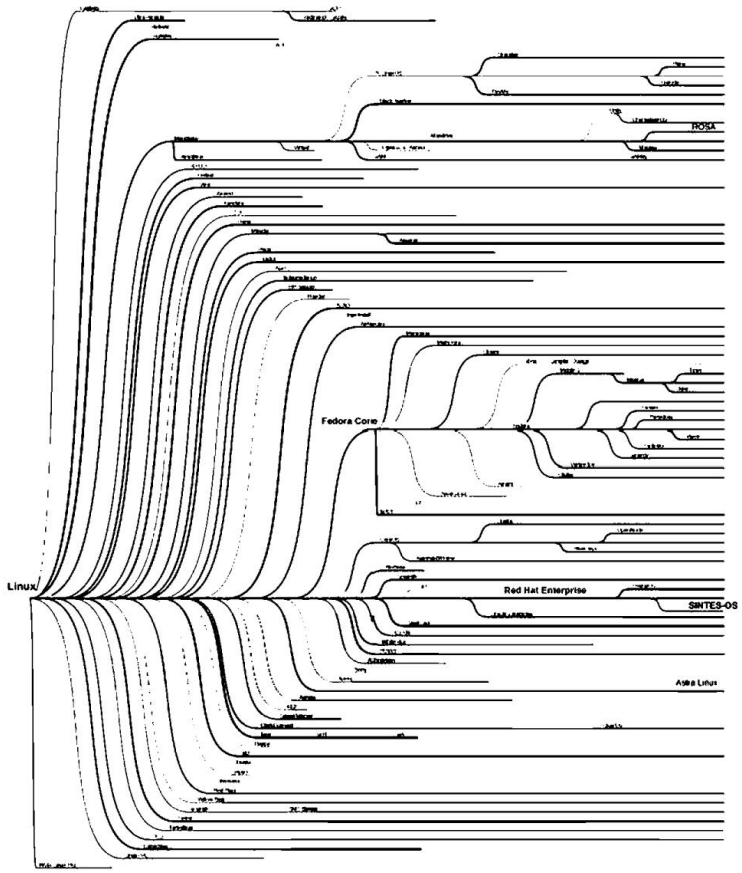


Рис. 2. Семейство операционных систем на основе ядра Linux

Для этой цели в 2012 г. под руководством и с участием автора разработчиком платформы «Синтез-АПП» было подписано соглашение с Red Hat [14], согласно которому в платформу было разрешено встраивание модулей и программ, прошедших внутреннюю сертификацию Red Hat, разработчику платформы был открыт доступ к базам данных (знаний) о совместимости программ и аппаратных средств, ошибках и уязвимостях программ, выявленных Red Hat, путях их устранения.

Взаимодействие с Red Hat обеспечило быстрое достижение компанией-разработчиком требуемого научно-технического потенциала, оперативную локализацию заимствованного ПО, разработку собственных программных средств защиты информации и средств взаимодействия с внешними средствами защиты информации, проведение комплексных нагрузочных и функциональных испытаний, испытаний на совместимость с аппаратными средствами, комплекса исследований на соответствие требованиям регуляторов в области информационной безопасности, создание системы технической поддержки и подготовки специалистов, формирование собственных баз данных (знаний) по продуктам платформы «Синтез-АПП».

Для разработки защищенной СУБД «Синтез-СУБД» было принято решение об использовании кода открытой СУБД PostgreSQL, в создании которого принимали участие известные российские разработчики мирового уровня [15].

По всем направлениям разработки программных компонент платформы обеспечивалось взаимодействие с сообществами разработчиков открытого ПО [16], организованное по образцу Red Hat (рис. 3 [17]).

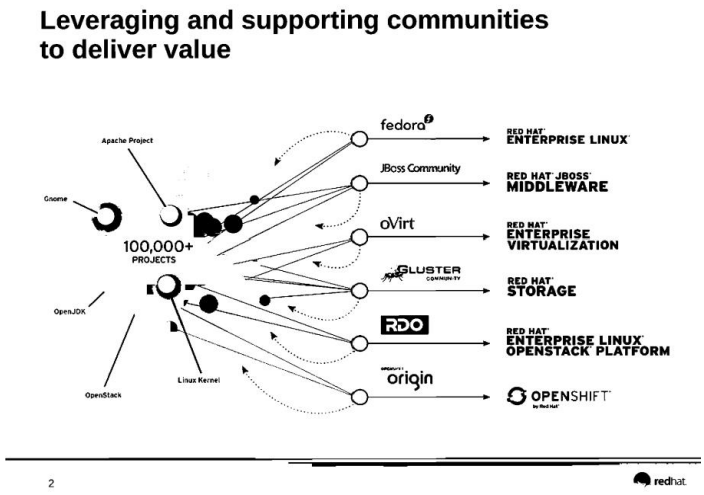


Рис. 3. Использование и поддержка сообществ открытого ПО

Решения по обеспечению универсальности платформы «Синтез-АПП»

Для достижения универсальности платформы было принято решение о создании семейства защищенных операционных систем «Синтез-ОС»

(см. таблицу 1) на основе технологий виртуализации и унификации программных модулей. Состав основных модулей семейства операционных систем «Синтез-ОС» приведен на рис. 4.

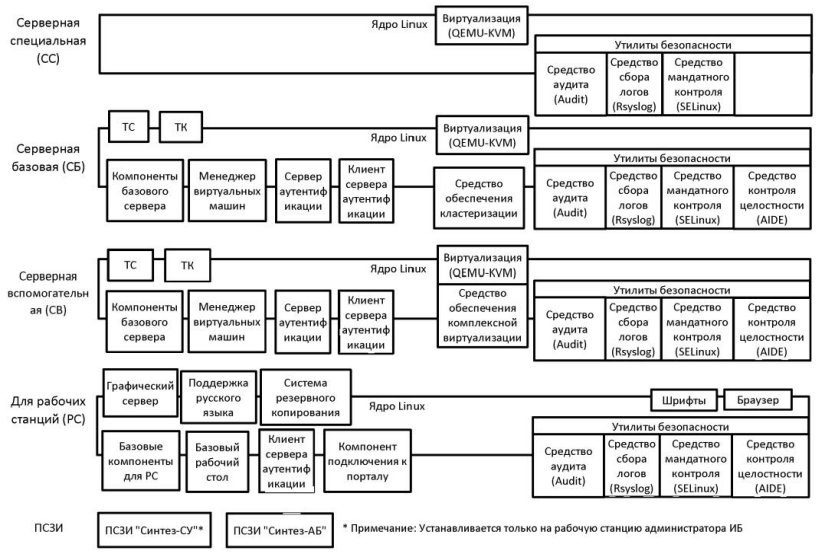


Рис. 4. Модульный состав семейства операционных систем «Синтез-ОС»

На объектах КИИРФ, созданных на платформе, обработка информации осуществляется на виртуальных машинах, развернутых на защищенных серверах. Программные компоненты платформы были разработаны для установки на серверах с аппаратной поддержкой виртуализации, рабочих станциях, терминалах.

Взаимодействие пользователя с виртуальной машиной осуществляется посредством терминального АРМ «Синтез-Т» (рис. 5), который обладает высокой безопасностью, малыми массой и габаритами, низким энергопотреблением, не требует для своего функционирования источников бесперебойного питания. АРМ не содержит собственных средств хранения информации, что исключает утрату защищаемой информации в случае выключения электропитания или хищения АРМ.

Рис. 5. Терминал «Синтез-Т»

Реализованные в платформе меры по защите среды виртуализации исключают несанкционированный доступ как к информации, обрабатываемой в виртуальной инфраструктуре, так и к компонентам виртуальной инфраструктуры: средствам управления виртуальной инфраструктурой, монитору виртуальных машин (гипервизору), системе хранения данных (включая систему хранения образов виртуальной инфраструктуры), сети передачи данных через элементы виртуальной и физической инфраструктуры, гостевым операционным системам, виртуальным машинам, системе и сети репликации, терминальным и виртуальным устройствам, а также системе резервного копирования и создаваемым ею копиям. Пример реализации объекта КИИРФ представлен на рис. 6.

Решения по обеспечению гарантий развития и поддержки платформы «Синтез-АПП»

Проектирование, развертывание, техническая поддержка и квалификационная поддержка (сертификация) подготовки эксплуатационного и обслуживающего персонала объектов КИИРФ по аппаратным и программным компонентам платформы обеспечиваются силами отечественных разработчиков, имеющих также доступ к ресурсам сообществ разработчиков открытого ПО.

Общая схема организации технической поддержки платформы «Синтез-АПП» представлена в таблице 2. Поддержка и развитие платформы, созданных на ней объектов КИИРФ обеспечиваются совокупностью следующих факторов.

1. Научно-техническим потенциалом отечественного разработчика платформы, авторским сопровождением объектов КИИРФ, созданных на платформе.

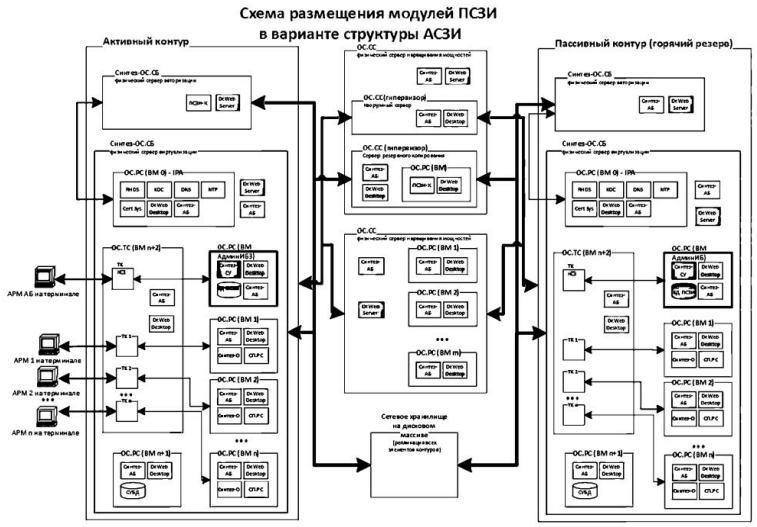


Рис. 6. Пример реализации объекта КИИРФ

2. Ведением разработчиком платформы баз данных (знаний) об ошибках (уязвимостях) программ платформы, имевшихся инцидентах на объектах КИИРФ.

3. Доступностью баз данных (знаний) об ошибках и уязвимостях в заимствованном до локализации программном обеспечении, выявленных сообществами разработчиков открытого ПО.

4. Доступностью для разработчика платформы баз данных (знаний) Red Hat об ошибках и уязвимостях в заимствованном до локализации программном обеспечении, выявленных в процессе внутренней сертификации Red Hat.

5. Доступностью для разработчика платформы сведений о результатах внутренней сертификации Red Hat аппаратных средств, программного обеспечения на их совместимость, производительность и отказоустойчивость различных тестовых реализаций технических решений.

6. Ведением разработчиком платформы баз данных о совместимости сертифицированных средств платформы с аппаратными средствами,

прикладными программами и СПО владельцев (заказчиков) объектов КИИРФ.

4. Система защиты информации платформы «Синтез-АПП»

Особенностью платформы «Синтез-АПП» является то, что ее система защиты информации обеспечивает создание объектов КИИРФ различных классов защищенности: от объектов, на которых информация является открытой до объектов, на которых обрабатывается конфиденциальная информация (для служебного пользования, персональные данные, банковская тайна, коммерческая тайна, врачебная тайна и т.п.) или имеющая гриф секретности (секретно, совершенно секретно).

Комплекс программ «Защищённая операционная система «Синтез» соответствует требованиям ФСБ России по защите информации от несанкционированного доступа с использованием средств криптографической защиты информации в автоматизированных информационных системах, расположенных на территории Российской Федерации, 1 класса, и может использоваться для обработки информации, содержащей сведения, составляющие государственную тайну [18].

Для удовлетворения требованиям по информационной безопасности в составе системы защиты информации платформы «Синтез-АПП» созданы подсистемы управления доступом, регистрации и учета, криптографическая подсистема, подсистемы обеспечения целостности и антивирусной защиты.

Соответствие требуемому классу защищенности объекта КИИРФ достигается многообразными настройками подсистем системы защиты информации. Конфиденциальный характер сведений о решениях и настройках подсистем системы защиты информации не позволяет в рамках статьи остановиться на них подробнее.

Реализованные в платформе механизмы защиты информации обеспечивают безопасное взаимодействие объектов КИИРФ разных классов защиты, в том числе объектов с разными уровнями конфиденциальности обрабатываемой информации. Пример организации безопасного взаимодействия объектов КИИРФ, применительно к классификации, разработанной в [19], приведен на рис. 7

5. Развитие платформенных решений для объектов критической информационной инфраструктуры

Стратегия импортозамещения существенно активизировала разработку отечественного программного обеспечения. По состоянию на ноябрь 2018 г. Единый реестр российских программ для электронных вычислительных машин и баз данных [20] уже содержал 46 продуктов толь-

ко класса «Операционные системы» и 47 продуктов класса «Системы управления базами данных».

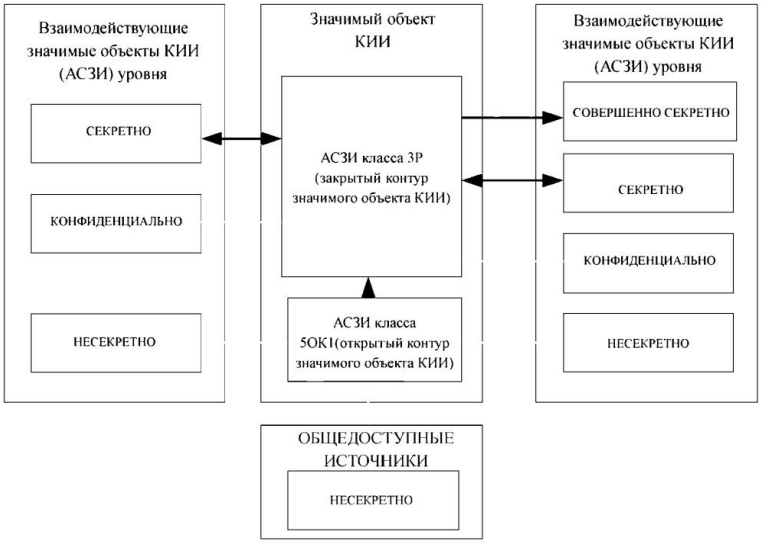


Рис. 7. Пример организации безопасного взаимодействия объектов КИИРФ

Следует отметить, что почти все операционные системы были разработаны на основе ядра Linux, однако далеко не все они могут быть использованы при создании и эксплуатации объектов КИИРФ, что обусловлено требованиями регуляторов в области информационной безопасности, очень немногие отвечают промышленному уровню, обеспечивают универсальность, возможности «наследования» действующих объектов КИИРФ, предоставляют надежные гарантии развития и поддержки.

С позиций изложенной выше методологии критериям «платформа», наряду с платформой «Синтез-АПП», наиболее близко соответствуют линейки продуктов семейства операционных систем «Astra Linux Special Edition» и программных комплексов семейства «Циркон». Подробная информация о данных платформах представлена на сайтах производителей. [21, 22].

Предположительно, платформа «Astra Linux Special Edition» была разработана на основе одного из дистрибутивов операционной системы Debian. Операционные системы Debian используют ядро Linux или FreeBSD [23].

Как отражено на сайте разработчика платформа «Циркон» была разработана на базе одного из дистрибутивов операционной системы CentOS. Операционная система CentOS представляет собой дистрибутив Linux, основанный на коммерческом Red Hat Enterprise Linux (RHEL) [24]. Однако, как неоднократно отмечалось Red Hat [17], операционные системы CentOS являются клонами RHEL, поддержка которых Red Hat не осуществляется.

Детальное сравнение платформ требует отдельного исследования, что в задачи данной статьи не входит. Но можно отметить, что в отличие от других, только платформа «Синтез-АПП» изначально базировалась на программных модулях (пакетах) дистрибутивов RHEL, RHEV (Red Hat Enterprise Virtualization), имевших и имеющих контроль и поддержку промышленного уровня. Это дает дополнительные гарантии защиты от непредвиденных ошибок (уязвимостей), обеспечивает оперативность устранения возможных инцидентов на объектах КИИРФ. В платформе «Синтез-АПП» была реализована описанная выше методология, которая применялась с этапа проектирования платформы и используется сейчас при сопровождении и технической поддержке созданных на ней объектов.

6. Заключение

Курс на импортозамещение программного обеспечения вызвал большое оживление разработчиков. Однако далеко не все разработчики изначально задумываются о необходимости обеспечить безопасность своих программ для информации, которая обрабатывается (хранится, передается, и т.д.) с их помощью или с которой они взаимодействуют. Поэтому, несмотря на значительный объем разработок, выполненных за последние годы, далеко не все программы могут быть применены на объектах КИИРФ. Изложенные в статье методологические аспекты и опыт создания отечественной аппаратно-программной платформы ориентированы на разработчиков программного обеспечения и платформенных решений. Их учет и применение обеспечит повышение качества и оперативности создания объектов КИИРФ и иных разнообразных автоматизированных систем в защищенном исполнении, гарантий их развития и поддержки.

Список литературы

[1] Maitland Hyslop. Critical Information Infrastructures: Resilience and Protection. -Springer, 2007. - 277 p.

[2] Доктрина информационной безопасности Российской Федерации. Утверждена Указом Президента Российской Федерации от 05.12.2016 № 646.

3] Стратегия развития информационного общества в Российской Федерации на 2017 - 2030 годы. Утверждена Указом Президента Российской Федерации от 09.05.2017 № 203.

4] Программа «Цифровая экономика Российской Федерации». Утверждена Постановлением Правительства Российской Федерации от 28.07.2017 г. № 1632-р.

[5] ГОСТ Р 51624-2000. Защита информации. Автоматизированные системы в защищенном исполнении. Общие требования. – М.: Стандартинформ, 2000. – 10 с.

[6] ГОСТ Р 50.1.056-2006. Техническая защита информации. Основные термины и определения. – М.: Стандартинформ, 2006. – 20 с.

[7]Сабанов А.Г. Доверенные системы как средство противодействия киберугрозам // Защита информации. Инсайд. 2015, № 3 (63), с. 17-21.

[8]Муравник В.Б., Захаренков А.И., Добродеев А.Ю. Некоторые предложения по подходу и порядку реализации политики и стратегии импортозамещения в интересах национальной безопасности и укрепления обороноспособности Российской Федерации // Вопросы кибербезопасности. 2016, № 1 (14), с. 2-8.

[9]Михалевич И.Ф. Концепция создания доверенной среды функционирования автоматизированных систем в защищенном исполнении на базе операционной системы «Синтез-ОС». – М.: ООО «АйТиСириус», 2012. – 50 с. [Электронный ресурс]. URL: https://www.itsirius.su/resheniya/ (дата обращения: 20.12.2012).

[10]Михалевич И.Ф. Проблемы создания доверенной среды функционирования автоматизированных систем управления в защищенном исполнении / Труды XII Всероссийского совещания по проблемам управления (ВСПУ-2014, Москва). - М.: Институт проблем управления им. В.А.Трапезникова РАН, 2014. - С. 9201-9207.

[11]Аттестат № СФ/014-3065 от 10.02.2017 соответствия Комплекса программ «Защищённая операционная система «Синтез» требованиям ФСБ России по защите информации от несанкционированного доступа с использованием средств криптографической защиты информации в автоматизированных информационных системах, расположенных на территории Российской Федерации, 1 класса. Выдан ЦЛСЗ ФСБ России.

[12]Свидетельство на товарный знак (знак обслуживания) № 533289 «СИНТЕЗАЙТИС», приоритет товарного знака 03.12.2013 г., зарегистрировано в Государственном реестре товарных знаков и знаков обслуживания РФ 30.01.2015 г.

[13] Robert Love. Linux Kernel Development, 3rd Edition. - Pearson Education, Inc., 2010. - 468 p.

[14] Memorandum of intensions of the parties Red Hat Inc., EMEA Red Hat Limited, «ITSirius» LLC, 2012).[Электронный ресурс]. URL: https://www.itsirius.su/partnery/ (дата обращения: 25.06.2014).

[15] И. Панченко. PostgreSQL: вчера, сегодня, завтра // Открытые системы. СУБД, 2015, № 3, с. 34-37.

[16] Сообщества Linux в интернете.[Электронный ресурс]. URL: https://losst.ru/soobshhestva-linux-v-internete/ (дата обращения 08.01.2018).

[17] The Red Hat Enterprise Linux advantage over CentOS in your enterprise (presentation). 2014, Red Hat.

[18] Выписка из перечня средств защиты информации, сертифицированных ФСБ России. [Электронный ресурс]. URL: http://clsz.fsb.ru/certification.htm (дата обращения: 10.08.2018).

[19] Калашников А.О., Михалевич И.Ф. Унифицированная система классификации защищенности значимых объектов критической информационной инфраструктуры российской федерации по критериям безопасности информации // Информация и безопасность, 2018, т. 21, вып. 1. С. 6-17

[20] Единый реестр российских программ для электронных вычислительных машин и баз данных. [Электронный ресурс]. URL: https://reestr.minsvyaz.ru/reestr/ (дата обращения: 23.11.2018).

[21] Операционные системы «Astralinux». [Электронный ресурс]. URL: http://astralinux.ru/ (дата обращения: 23.11.2018).

[22] Программный комплекс «Циркон». [Электронный ресурс]. URL: https://www.swemel.ru/ (дата обращения: 23.11.2018).

[23] Проект Debian. [Электронный ресурс]. URL: https://www.debian.org/ (дата обращения: 23.11.2018).

[24] Проект CentOS. [Электронный ресурс]. URL: https://www.centos.org/ (дата обращения: 23.11.2018).

Requirements, principles, practice of creating domestic hardware-software platforms for automated systems in the protected execution of the critical information infrastructure of the Russian Federation

Mikhalevich I.F.

The paper sets out a system of requirements and principles that determine the methodological aspects of creating domestic hardware-software platforms for automated systems in the protected execution as the basis of the critical information infrastructure of the Russian Federation, describes the experience of creating a protected hardware-software platform "Sintez-HSP"that fully satisfies the most strict requirements for information security, reliability, scalability, ensuring the independence of critical information infra the structure of foreign technologies and software.
Keywords: automated systems in the protected execution, hardware-software platform, information security, critical information infrastructure, Sintez-HSP

Штрафные, барьерные, квазибарьерные функции и функции, обратные к ним

Бирюков А.Г., Чернов А.В., Чернова Ю.Г., Шароватова Ю.И.

Рассматриваются методы внешних штрафных функций, внутренних штрафных функций и квазибарьерных функций для решения задач математического программирования. Предложены новые квазибарьерные функции. Доказаны теоремы сходимости указанных методов к решению задач математического программирования. Рассмотрены свойства указанных функций при их преобразованиях: дифференцирование, интегрирование, построение обратных к ним функций.

Ключевые слова: внешние штрафные функции, внутренние штрафные функции, барьерные штрафные функции, обратные функции, квазибарьерные функции, задача математического программирования, дифференциальные барьеры, степенные дифференциальные барьеры, энтропийные дифференциальные барьеры, сходимость методов дифференциальных барьеров к решению задачи математического программирования.

Введение

Методы внешних штрафных и внутренних штрафных (барьерных) функций широко используются в практике решения задач математического программирования (МП) ([1, 2, 3]). Менее известны так называемые *методы квазибарьерных функций* ([4, 5]). Все эти методы можно отнести к классу методов последовательной безусловной минимизации ([6, 7]).

Рассмотрим задачу МП в виде:

$$\begin{gathered} \min f(x),\ x \in G \subset \mathrm{R}^n, \\ G = \left\{ x \in \mathrm{R}^n : \varphi_i(x) \le 0,\ i = \overline{1,m};\ h_j(x) = 0, j = \overline{1,l} \right\}, \end{gathered} \tag{1}$$

где функции f и φ_i, $i = \overline{1,m}$ – дифференцируемые; h_j, $j = \overline{1,l}$ – непрерывно дифференцируемые на R^n. Пусть $x^* \in G$ – решение задачи (1).

Рассмотрим при $\tau > 0$ вспомогательную задачу:

$$\begin{gathered}\min_{x\in \mathrm{R}^n} F(x,\tau),\\ F(x,\tau) = f(x) + \sum_{i=1}^{m} P_i(\tau,\varphi_i) + \sum_{j=1}^{l} P_{m+j}(\tau,h_j).\end{gathered} \tag{2}$$

Обозначим $x(\tau)$ – решение задачи (2). Тогда методы штрафных и барьерных функций для решения задачи (1) заключаются в последовательном решении задачи (2) для последовательности $\{\tau_k\}_{k=0}^{\infty}$, такой, что $\lim\limits_{k\to\infty} \tau_k = 0$, причем $x^* = \lim\limits_{k\to\infty} x(\tau_k)$. В задаче (2) функция $P_i(\tau,\varphi_i)$, $i \in [1,m]$ может быть внешней штрафной функцией (ШФ), внутренней (барьерной) ШФ или квазибарьерной (КБ) функцией, а все функции $P_{m+j}(\tau,h_j)$ $j = \overline{1,l}$ – внешние ШФ, например, $P_{m+j}(\tau,h_j) = \frac{1}{\tau}\left|h_j(x)\right|^s, s = 1,2,3,...,$ Основным объектом наших исследований будут функции $P_i(\tau,\varphi_i)$, $i = \overline{1,m}$.

Целями настоящей работы являются:

1) расширение класса КБ функций;

2) доказательство сходимости: $x^* = \lim\limits_{\tau\to 0} x(\tau)$ для методов штрафных и КБ функций при решении задачи (2);

3) изучение свойств функций $P_i(\tau,\varphi_i)$, $i = \overline{1,m}$ при их преобразованиях:

   - дифференцировании по φ_i;
   - построении функции $R(\tau,\lambda_i)$ обратной к $\frac{\partial P_i(\tau,\varphi_i)}{\partial \varphi_i}$;
   - интегрировании функции $R(\tau,\lambda_i)$ по λ_i.

1. Методы штрафных функций

Укажем кратко требования, которым должны удовлетворять внешние и внутренние ШФ [2], полагая $\tau > 0$.

- Для внешней ШФ и ограничения вида $\varphi(x) = 0$ или $\varphi(x) \le 0$:

$$\begin{gathered}P(\tau,\varphi(x)) = 0 \text{ или } \lim_{\tau\to 0+} P(\tau,\varphi(x)) = 0 \text{ при } x \in G;\\ P(\tau,\varphi(x)) > 0 \text{ при } x \notin G;\\ P(\tau_1,\varphi(x)) < P(\tau_2,\varphi(x)) < +\infty \text{ при } x \notin G \text{ и } \tau_1 > \tau_2;\\ \lim_{\tau\to 0+} P(\tau,\varphi(x)) = +\infty \text{ при } x \notin G.\end{gathered} \tag{3}$$

- Для внутренней ШФ и ограничения вида $\varphi(x) \leq 0$ при условии $\text{int}G \neq \emptyset$:

$$\begin{gathered} P(\tau_1, \varphi(x)) > P(\tau_2, \varphi(x)) > 0 \text{ при } x \in \text{int}G \text{ и } \tau_1 > \tau_2; \\ \lim_{\tau \to 0+} P(\tau, \varphi(x)) = 0 \text{ при } x \in \text{int}G; \\ \lim_{\tau \to 0+} P(\tau, \varphi(x)) = +\infty \text{ при } x \notin \text{int}G. \end{gathered} \tag{4}$$

Приведем примеры штрафных функций для решения задачи (2). Рассмотрим вначале внешние штрафные функции:

- Степенная функция, которая используется для ограничений-равенств $h(x) = 0$:

$$P(\tau, h(x)) = \frac{1}{\tau}(h(x))^{\beta}, \text{ где } \beta = 2, 4, 6, ...$$

Данная функция является внешней ШФ. Наиболее часто применяется квадратичная ШФ, т.е. $\beta = 2$.

- Степенная ШФ, которая используется для ограничений-неравенств $\varphi(x) \leq 0$:

$$P(\tau, \varphi(x)) = \frac{1}{\tau}(\varphi(x)_+)^{\beta}, \ \beta = 2, 3, 4...$$

Здесь $\varphi(x)_+ = \max(0, \varphi(x))$ – так называемая «функция-срезка», которая широко применяется на практике. При $\beta = 2$ она непрерывно дифференцируема. Для повышения порядка ее дифференцируемости надо увеличивать степень β, т.е. $\beta = 3, 4$ и т.д.

- Показательные ШФ для ограничений-неравенств $\varphi(x) \leq 0$:

$$P(\tau, \varphi(x)) = \tau a^{\frac{\varphi(x)}{\tau}} \text{ при } a > 1.$$

Такие функции бесконечно дифференцируемы по φ. При $a = e$ такая функция называется экспоненциальной ШФ:

$$P(\tau, \varphi_i(x)) = \tau e^{\frac{\varphi_i(x)}{\tau}}.$$

В качестве внутренних можно применять традиционные ШФ, которые определены при $x \in \text{int}G$, вида:

$$\begin{gathered} P(\tau, \varphi_i(x)) = -\tau \ln(-\varphi_i(x)) \text{ – логарифмическая ШФ;} \\ P(\tau, \varphi_i(x)) = -\frac{\tau}{\varphi_i(x)} \text{ – обратная ШФ.} \end{gathered}$$

Пусть J_{in} – некоторое подмножество индексов из $[1, m]$, где m – количество ограничений неравенств, $J_{ext} = [1, m] \backslash J_{in}$. Введем множества:

- Множество $M_1 = \{x \in \mathrm{R}^n : \varphi_i(x) \le 0,\ i \in J_{in}\}$. К ограничениям в этом множестве применяется метод внутренних ШФ.

- Множество $M_2 = \left\{x \in \mathrm{R}^n : \varphi_i(x) \le 0,\ i \in J_{ext},\ h_j(x) = 0,\ j = \overline{1,l}\right\}$. К ограничениям во множестве M_2 применяется метод внешних ШФ.

Таким образом множество M_1 определено произвольной выборкой J_{in} ограничений-неравенств задачи (1), а множесво M_2 определяется ограничениями-равенствами и ограничениями-неравенствами J_{ext}, не вошедшими в J_{in}, поэтому $G = M_1 \bigcap M_2$.

Под ε-окрестностью точки x будем понимать шар $B_\varepsilon(x) = \{y \in \mathrm{R}^n : \|y - x\| < \varepsilon\}$.

Сформулируем теорему о сходимости метода ШФ со вспомогательной задачей (2) в случае, когда функции $P_i(\tau, \varphi_i(x))$ могут быть как внутренними, так и внешними ШФ [2].

Теорема 1. *[2]*

Пусть для задачи (1) *выполняются условия:*

- *функции* f, φ_i, $i = \overline{1,m}$, h_j, $j = \overline{1,l}$ *– непрерывно дифференцируемы;*
- *существует* x^* *– решение задачи* (1)*;*
- *для некоторого* $\varepsilon > 0$ *выполнено условие* $B_\varepsilon(x^*) \bigcap M_2 \bigcap \mathrm{int} M_1 \neq \emptyset$*;*
- *множество* $G_1(x_0) = \{x \in G : f(x) \le f(x_0)\}$ *– компакт при* $x_0 \in G$*;*
- *выполнены условия* (3) *и* (4) *для множеств* M_1 *и* M_2*.*

Тогда существует монотонно убывающая положительная последовательность τ_k, $k = 0, 1, \ldots$, $\lim\limits_{k\to\infty} \tau_k = 0$ *и соответствующая ей последовательность* $x_k = \arg\min\limits_{x\in\mathrm{R}^n} F(x, \tau_k)$ *такая, что* $x^* = \lim\limits_{k\to\infty} x_k$*.*

Замечание 1. *1) Если для решения задачи* (2) *применяется только метод внешних ШФ, то в теореме 1 можно исключить условие* $B_\varepsilon(x^*) \bigcap M_2 \bigcap \mathrm{int} M_1 \neq \emptyset$*, при этом множество* $M_1 = \mathrm{R}^n$*.*

2) Как видно из определения M_1 *и* M_2 *существует множество вариантов решения задачи* (1) *методами внутренних и внешних ШФ.*

3) Вместо условия G_1 *– компакт, можно взять условие:* $G_2 = \{x \in \mathrm{R}^n, x_0 \in G : F(x_0, \tau_k) \le F(x, \tau_k), k = 1, 2, \ldots\}$ *– компакт.*

2. Метод квазибарьерных функций

Основным свойством методов ШФ является «локализация» точки решения задачи (1): вспомогательная функция $F(x,\tau)$ строится так, что решение задачи $\min\limits_{x\in \mathrm{R}^n} F(x,\tau)$, точка $x(\tau)$, является приближением к решению x^* задачи МП и тем более точным, чем меньше коэффициент штрафа $\tau > 0$. При определенных условиях $x^* = \lim\limits_{\tau\to+0} x(\tau)$.
Рассмотрим задачу, допустимое множество которой определяется ограничениями-неравенствами:

$$\begin{gathered}\min f(x),\ x\in G,\\ G=\left\{x\in \mathrm{R}^n : \varphi_i(x)\le 0,\ i=\overline{1,m}\right\}.\end{gathered} \tag{5}$$

Одними из возможных штрафов для решения этой задачи являются степенные штрафы [2, 6]:

- $P(\tau,\ \varphi_i(x)) = \frac{1}{\tau}\left(\varphi_i(x)_+\right)^\alpha$, при $\alpha \ge 1$ – внешняя ШФ;
- $P(\tau,\ \varphi_i(x)) = \tau\left(-\varphi_i(x)\right)^\alpha$, при $\alpha < 0$ – внутренняя ШФ (или барьерная функция).

Для значений степени $\alpha \in (0,1)$ предложены квазибарьерные функции [4, 5]:

$$P(\tau,\varphi_i(x)) = -\tau\left(-\varphi_i(x)\right)^\alpha,\ 0<\alpha<1,\ x\in G; \tag{6}$$

и соответсвующая вспомогательная функция

$$F(x,\tau) = f(x) - \tau\sum_{i=1}^{m}\left(-\varphi_i(x)\right)^\alpha,\ 0<\alpha<1,\ x\in G. \tag{7}$$

Спецификой задачи МП (5) является то, что всегда $x^*\in \partial G$, где ∂G – граница множества G[1].

Определение 1. *Множество $J(x) = \{i\in[1,m] : \varphi_i(x) = 0\}$ называется множеством индексов активных ограничений в точке x.*

Теорема 2. *Пусть для задачи* (5)

- *функция f дифференцируема на допустимом множестве G;*
- *множество G имеет непустую внутренность:* $\mathrm{int}G \ne \emptyset$*;*

[1]Если $x^*\notin \partial G$, то в такой задаче $x^*\in \mathrm{int}G$, и задача МП становится задачей безусловной минимизации

- *функции* φ_i, $i = \overline{1,m}$ *дифференцируемы на* R^n;
- *решение задачи* (5) $x^* \in G$ *существует;*
- *вспомогательная функция* $F(x,\tau)$ *определена в* (7);
- *при* $\overline{x} \in G$ *множество* $G_1(\overline{x}) = \{x \in \mathrm{R}^n : F(x,\tau) \le F(\overline{x},\tau)\}$ – *компакт.*

Тогда существует $x(\tau) \in \mathrm{int}G$ – *решение задачи:* $\min F(x,\tau)$, $x \in G$, *причем*

$$x(0) = x^*, \quad f(x^*) = F(x^*,0).$$

Доказательство. Очевидно, что множество G замкнуто, и существует решение $x(\tau) \in G_1 \subset G$. Покажем, что $x(\tau) \in \mathrm{int}G$.

Пусть $x_0 \in \partial G$ – решение задачи $\min F(x,\tau)$, $x \in G$, а число $\gamma > 0$ и направление $s \in R^n$ такие, что $x = x_0 + \gamma s \in \mathrm{int}G$. Тогда, производная функции $F(x,\tau)$ в точке x_0 по направлению s:

$$\begin{aligned} F'(x_0,\tau,s) = & \lim_{\gamma\to 0+} \tfrac{F(x_0+\gamma s,\tau)-F(x_0)}{\gamma} = \\ = & \lim_{\gamma\to 0+} \tfrac{f(x_0+\gamma s)-f(x_0)}{\gamma} + \lim_{\gamma\to 0+} \sum_{i=1}^{m} \tfrac{P_i(\tau,\,\varphi_i(x_0+\gamma s))-P_i(\tau,\,\varphi_i(x_0))}{\gamma} \\ = & \nabla f(x_0)^{\mathrm{T}} s - \tau \lim_{\gamma\to +0} \sum_{i=1}^{m} \tfrac{((-\varphi_i(x_0+\gamma s))^\alpha - (-\varphi_i(x_0))^\alpha)}{\gamma}. \end{aligned}$$

Если $\varphi_i(x_0) = 0$, $i \in J(x_0)$, то с учётом $-\nabla\varphi_i(x_0)^{\mathrm{T}} s > 0$ при $x \in \mathrm{int}G$:

$$\begin{aligned} \lim_{\gamma\to 0+} \tfrac{(-\varphi_i(x_0+\gamma s))^\alpha - (-\varphi_i(x_0))^\alpha}{\gamma} = & \lim_{\gamma\to 0+} \tfrac{(-\varphi_i(x_0+\gamma s))^\alpha}{\gamma} = \\ = & \lim_{\gamma\to 0+} \tfrac{\left(-\varphi_i(x_0) - \gamma\nabla\varphi_i(x_0)^{\mathrm{T}} s + o(\gamma)\right)^\alpha}{\gamma} = \\ = & \lim_{\gamma\to 0+} \tfrac{(-\gamma\nabla\varphi_i(x_0)^{\mathrm{T}} s)^\alpha}{\gamma} = \\ = & \lim_{\gamma\to 0+} \tfrac{\left(-\nabla\varphi_i(x_0)^{\mathrm{T}} s\right)^\alpha}{\gamma^{1-\alpha}} = +\infty. \end{aligned}$$

Если $\varphi_i(x) < 0$, т.е. $i \notin J(x_0)$, то функция $P_i(\tau,\varphi_i(x))$ дифференцируемая, и

$$\lim_{\gamma\to +0} \frac{P_i(\tau,\varphi_i(x_0+\gamma s) - P_i(\tau,x_0)}{\gamma} = \nabla_x P(\tau,\varphi_i(x_0))^{\mathrm{T}} s < \infty.$$

Таким образом значение $F'(x_0, \tau, s)$ имеет вид (т.к. точка $x_0 \in \partial G$, то $J(x_0) \neq \emptyset$):

$$F'(x_0, \tau, s) = \nabla f(x_0)^{\mathrm{T}} s + \\ + \sum_{i \notin J(x_0)} \nabla_x P(\tau, \varphi_i(x_0))^{\mathrm{T}} s - \tau \sum_{i \in J(x_0)} \lim_{\gamma \to +0} \frac{\left(-\nabla \varphi_i(x_0)^{\mathrm{T}} s\right)^{\alpha}}{\gamma^{1-\alpha}} = -\infty. \quad (8)$$

Из (8) следует, что точка $x_0 \in \partial G$ не может быть точкой минимума функции $F(x, \tau)$, т.к. по теореме 4.21 ([8], стр.163) должно выполняться неравенство $F'(x_0, \tau, s) \geq 0 \; \forall s \in T(x_0, G)$[2]. Итак, доказано, что $x_0 \in \mathrm{int}G$.

Так как в (7) функция $\sum\limits_{i=1}^{m} (-\varphi_i(x_0))^{\alpha}$ ограничена на $G_1 \subset G$, то при $\tau = 0, \; x(0) = x^*, F(x(0), 0) = f(x^*)$. □

Следствие 1. *Пусть теперь задача* (5) *выпукла, т.е. функции f и φ_i, $i = \overline{1, m}$ выпуклы на R^n, существует её решение x^* и множество G_1 – компакт. Тогда справедливо утверждение теоремы 2.*

Доказательство. Пусть $x_0 \in \partial G$ – решение вспомогательной задачи. Полагая $s \in T(x_0, G)$ и $x = (x_0 + \gamma s) \in intG$, найдем $F'(x_0, \tau, s)$:

$$F'(x_0, \tau, s) = f'(x_0, s) + \sum_{i \in J(x_0)} P_i'(\tau, \varphi_i(x_0), s) + \sum_{i \notin J(x_0)} P_i'(\tau, \varphi_i(x_0), s).$$

Если $i \in J(x_0)$, то в силу того, что производная по направлению выпуклой функции $\varphi'(x_0, s)$ существует [1]:

$$\begin{aligned} P'(\tau, \varphi_i(x_0), s) = & \; -\tau \lim_{\gamma \to +0} \tfrac{(-\varphi_i(x_0 + \gamma s))^{\alpha} - (-\varphi_i(x_0))^{\alpha}}{\gamma} = \\ = & \; -\tau \lim_{\gamma \to +0} \tfrac{(-\varphi_i(x_0) - \gamma \varphi'(x_0, s) - o(\gamma))^{\alpha}}{\gamma} = \\ = & \; -\tau \lim_{\gamma \to +0} \tfrac{(-\gamma \varphi'(x_0, s))^{\alpha}}{\gamma} = \lim_{\gamma \to +0} \tfrac{-\tau(-\varphi'(x_0, s))^{\alpha}}{\gamma^{1-\alpha}} = -\infty. \end{aligned}$$

Если $i \notin J(x_0)$, $\varphi_i(x_0) < 0$ и $\varphi_i'(x_0, s)$ – производная функции φ_i по направлению s. Тогда:

[2]Здесь $T(x_0, G)$ – касательный конус в точке x_0 ко множеству G

$$
\begin{aligned}
P'(\tau,\varphi_i(x_0),s) = & -\tau \lim_{\gamma\to 0} \tfrac{(-\varphi_i(x_0+\gamma s))^\alpha-(-\varphi_i(x_0))^\alpha}{\gamma} = \\
= & -\tau \lim_{\gamma\to 0} \tfrac{\left(-\varphi_i(x_0)-\gamma\varphi_i'(x_0,s)+o(\gamma)\right)^\alpha-(-\varphi_i(x_0))^\alpha}{\gamma} = \\
= & -\tau \lim_{\gamma\to 0} \tfrac{(-\varphi_i(x_0)-\gamma\varphi'(x_0,s))^\alpha-(-\varphi_i(x_0))^\alpha}{\gamma} = \\
\tau\alpha(-\varphi_i(x_0))^{\alpha-1}\varphi_i'(x_0,s),
\end{aligned}
$$

т.е. $P'(\tau,\varphi_i(x_0),s)-$ существует.

Таким образом, $F'(x_0,\tau,s)=-\infty$ и справедливо утверждение теоремы.

□

Определение 2. *Будем называть функцию $P(\tau,y)$, $y\in R^1$ – дифференциальным барьером (ДБ) для задачи (1) с допустимым множеством G и её ограничения $\varphi_i(x)\le 0$, если в точках $\overline{x}$ таких, что $\varphi_i(\overline{x})=0$, функция $P(\tau,\varphi_i(\overline{x}))$ недифференцируема по направлению $s\in T(\overline{x},G)$ и $\lim\limits_{x\to\overline{x},x\in G,\varphi_i(x)<0}\|\nabla_x P(\tau,\varphi_i(x))\|=+\infty$.*

Метод решения задачи (1), использующий функцию функцию $F(x,\tau)$ с дифференциальным барьером $P(\tau,y)$ будем называть «метод дифференциальных барьеров» (МДБ).

Замечание 2. *Как видно из доказательства теоремы 2, локализацию точки x^* обеспечивает <u>недифференцируемость функции</u> $P(\tau,\varphi_i(x))=-\tau(-\varphi_i(x))^\alpha$ при $\tau>0$ и $0<\alpha<1$ по направлению $s\in T(x,G)$ в точке $x^*\in\partial G$, когда $\varphi_i(x^*)=0$. Т.е. функция $P(\tau,\varphi_i(x))$ в этом случае является «барьером» для $x(\tau): x(\tau)\ne x^*$ при $\tau>0$. По этой причине метод решения задачи (1), использующий функцию $F(x,\tau)$ с «барьером» $P(\tau,\varphi_i(x))$ вида (6), является «методом дифференциальных барьеров», а функция $P(\tau,\varphi_i(x))$ – дифференциальный барьер.*

Укажем требования, которым должны удовлетворять ДБ для ограничений $\varphi(x)\le 0$:

$$
\begin{gathered}
P(\tau,y)\xrightarrow{\tau\to 0+,\ y<0} 0,\ P(\tau,0)=0; \\
P(0,\varphi(x))=0,\ x\in Q\subset G,\ \text{где } Q \text{ – компакт} \\
\lim_{y\to 0-}\tfrac{\partial P(\tau,y)}{\partial y}=+\infty,\ \tau>0.
\end{gathered}
\tag{9}
$$

Этим требованиям удовлетворяет также «энтропийная функция» [9]:

$$
P(\tau,y)=-\tau y\ln(-y), \tag{10}
$$

для которой

$$P(\tau,0) = -\tau \lim_{y\to 0-} y\cdot\ln(-y) = 0;$$
$$\lim_{y\to 0-} \frac{\partial P(\tau,y)}{\partial y} = \lim_{y\to 0-} [-\ln(-y) - 1] = +\infty.$$

Дифференциальным барьером будет также функция:

$$P(\tau,\varphi) = \tau(-\varphi)^{\alpha}\ln(-\varphi), 0 < \alpha \leq 1.$$

Определение 3. *Функцию* (10) *назовем энтропийный дифференциальный барьер, а функцию* (6) *– степенной дифференциальный барьер.*

Очевидно, для функций (6) и (10)

$$\begin{gathered}\frac{\partial P(\tau,\varphi_i(x))}{\partial\varphi_i(x)} = \tau\alpha(-\varphi_i(x))^{\alpha-1},\ \alpha\in(0,1)\\ \text{В точке } x^*: \varphi_i(x^*) = 0,\ \lim_{\varphi_i\to -0}\frac{\partial P(\tau,\varphi_i)}{\partial\varphi_i} = +\infty.\end{gathered} \tag{11}$$

Замечание 3. *Введенные функции ДБ – функциональный синоним КБ функций. Но в смысловом плане квазибарьерные функции – настоящие барьерные функции (не квази), не являющиеся барьерными штрафными функциями, т.к.* $P(\tau,0) = 0$. *Поэтому далее мы будем считать КБ функции синонимом функций ДБ и применять оба понятия.*

На рис. 1 представлена геометрическая иллюстрация квазибарьерных функций в пространстве R^1.

1) $P_M = -\tau\varphi\ln(-\varphi)$, $P_A = -\tau(-\varphi)^{\alpha}$, $\alpha = 0,5$ $(0 < \alpha < 1)$, $\tau = 0,1$, $\varphi = x \leq 0$.

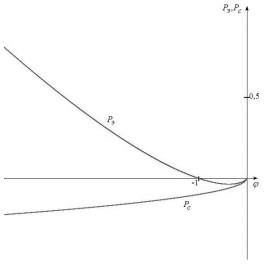


Рис. 1. Квазибарьерные функции в пространстве R^1.

Следствие 2. *Теорема 2 и следствие 1 верны для дифференциального барьера*

$$P(\tau, \varphi(x)) = -\tau\varphi(x)\ln(-\varphi(x)).$$

Доказательство. 1) Пусть $\varphi_i(x)$ – дифференцируемая функция и $\varphi_i(x_0) \equiv \varphi(x_0) = 0$. Рассмотрим предел

$$\begin{aligned} &-\tau \lim_{\gamma\to 0+} \tfrac{\varphi(x_0+\gamma s)\ln(-\varphi(x_0+\gamma s)-\varphi(x_0)\ln(-\varphi(x_0))}{\gamma} = \\ &= -\tau \lim_{\gamma\to 0+} \tfrac{(\gamma\varphi(x_0)^{\mathrm{T}}s+o(\gamma))\ln(-\gamma\nabla\varphi(x_0)^{\mathrm{T}}+o(\gamma))}{\gamma} = \\ &= -\tau\nabla\varphi(x_0)^{\mathrm{T}}s \cdot \lim_{\gamma\to 0+} \ln(-\gamma\nabla\varphi(x_0)^{\mathrm{T}}s) = -\infty. \end{aligned}$$

Если $\varphi(x_0) < 0$, то функция $P(\tau, \varphi_i(x))$ дифференцируемая, поэтому производная по направлению $s \in R^n$ существует и записывается в виде $P'(\tau, \varphi(x_0), s) = \nabla_x P(\tau, \varphi(x_0))^{\mathrm{T}}s$. Таким образом, утверждение теоремы 2 для энтропийного дифференциального барьера справедливо.

2) Пусть теперь функции f, φ_i, $i = \overline{1,m}$ выпуклы и $\varphi_i(x_0) = 0$. Найдем $P'(\tau, \varphi(x_0), s)$, учитывая $\nabla\varphi(x_0, s) < 0$:

$$\begin{aligned} P'(\tau, \varphi(x_0), s) = &-\tau \lim_{\gamma\to+0} \tfrac{\varphi(x_0+\gamma s)\ln(-\varphi(x_0+\gamma s))}{\gamma} = \\ = &-\tau \lim_{\gamma\to+0} \tfrac{\gamma\nabla\varphi(x_0)^{\mathrm{T}}s\ln(-\gamma\nabla\varphi(x_0)^{\mathrm{T}}s)}{\gamma} = \\ = &-\tau\nabla\varphi(x_0)^{\mathrm{T}}s \lim_{\gamma\to+0} \ln(-\gamma\nabla\varphi(x_0)^{\mathrm{T}}s) = -\infty. \end{aligned}$$

Если $\varphi(x)$ – выпукла, $\varphi(x_0) < 0$, то $\varphi'(x_0, s)$ для выпуклой функции существует, и

$$\varphi(x_0 + \gamma s) = \varphi(x_0) + \gamma\varphi'(x_0, s) + o(\gamma).$$

Опуская некоторые промежуточные выкладки, покажем, что производная по направлению существует:

$$P'(\tau, \varphi(x_0), s) =$$

$$\begin{aligned} = &-\tau \lim_{\gamma\to+0} \tfrac{(\varphi(x_0)+\gamma\varphi'(x_0,s))\ln(-\varphi(x_0)-\gamma\varphi'(x_0,s))-\varphi(x_0)\ln(-\varphi(x_0))}{\gamma} = \\ = &\ \varphi'(x_0, s)\ln\left(-\varphi(x_0)\right) + \varphi'(x_0, s) = \\ = &\ \varphi'(x_0, s)\left(\ln\left(-\varphi(x_0)\right) + 1\right). \end{aligned}$$

Следовательно, справедливо утверждение следствия 1 для дифференциального барьера (10).

□

На простых примерах покажем применение МДБ для решения задачи (5).

Пример 1. *Найти* $\min(-2x)$, *если* $\varphi(x) = x - 3 \le 0$.

Решение. *Для такой задачи функция* (10) *записывается в виде:*

$$F(x,\tau) = -2x + \tau(3-x)\ln(3-x),\ x \in \mathrm{R}^1.$$

Для такой функции:

$$\frac{\partial F}{\partial x} = -2 + \tau(-\ln(3-x) - 1) = 0.$$

Следовательно:

$$-\tfrac{2}{\tau} - 1 = \ln(3-x)\ \textit{и}\ 3 - x = \tfrac{1}{e^{1+2/\tau}};$$
$$x(\tau) = \left(3 - \tfrac{1}{e^{1+2/\tau}}\right) \in G.$$

Очевидно, $x^* = \lim\limits_{\tau\to 0} x(\tau) = 3$.

Пример 2. *Найти* $\min(-2x)$, *если* $0 \le x \le 3$.

Решение. *В этом случае:* $\varphi_1(x) = -x \le 0$ *и* $\varphi_2(x) = x - 3 \le 0$.
Применим функцию (10) *при* $\alpha = 1/2$:

$$F(x,\tau) = -2x - \tau\left(\sqrt{x} + \sqrt{3-x}\right)$$

Значит:

$$\tfrac{\partial F}{\partial x} = -2 - \tfrac{\tau}{2}\left(\tfrac{1}{\sqrt{x}} - \tfrac{1}{\sqrt{3-x}}\right) = 0$$
$$4\sqrt{x(3-x)} = \tau(\sqrt{x} - \sqrt{3-x})$$
$$16x(3-x) = \tau^2\left(3 - 2\sqrt{x(3-x)}\right).$$

Обозначим $y = \sqrt{x(3-x)}$, *тогда* $16y^2 = \tau^2(3-2y)$.
Решая полученное квадратное уравнение, находим $y = \frac{\sqrt{3}}{4}\tau + o(\tau)$ *и*

$$y^2 = -x^2 + 3x = \frac{3}{16}\tau^2 + o(\tau^2).$$

Тогда $x(\tau) = 3 - \frac{\tau^2}{16} + o(\tau^2)$ *и* $x^* = \lim\limits_{\tau\to 0} x(\tau) = 3$.

3. Условия перестановки операций и теоремы существования решений задач математического программирования в методах штрафных функций и дифференциальных барьеров

Рассмотрим задачу МП (1), в которой f и φ_i, $i = \overline{1,m}$ – непрерывные функции, а h_j, $j = \overline{1,l}$ – непрерывно дифференцируемые функции. Вспомогательная функция для нее имеет вид (2):

$$F(x,\tau) = f(x) + \sum_{i=1}^{m} P_i\left(\tau, \varphi_i(x)\right) + \sum_{j=1}^{l} P_{m+j}\left(\tau, h_j(x)\right).$$

Рассмотрим теперь метод решения задачи (2), в котором любая из функций $P_i(\tau, y)$, $i = \overline{1,m}$ может быть внешней, внутренней ШФ, а также функцией ДБ, а функции $P_{m+i}(\tau, y)$, $i = \overline{1,l}$ – внешние ШФ.

Пусть

$$G_i = \begin{cases} G = \bigcap_{i=1}^{m+l} G_i; \\ \{x \in \mathrm{R}^n : \varphi_i(x) \le 0\},\ i = \overline{1,m}, \\ \{x \in \mathrm{R}^n : h_{i-m}(x) = 0\},\ i = \overline{m+1, m+l}. \end{cases}$$

Пусть $\delta(x,G)$ – индикаторная функция множества[3]. Очевидно, что $\delta(x,G) = \sum\limits_{i=1}^{m+l} \delta(x,G_i)$.

Рассмотрим задачу:

$$\begin{gathered} \min \Phi(x,G),\ x \in \mathrm{R}^n; \\ \Phi(x,G) = f(x) + \delta(x,G) = f(x) + \textstyle\sum_{i=1}^{l} \delta(x,G_i). \end{gathered} \tag{12}$$

Лемма 1. *Пусть задача* (1) *имеет решение* $x^* \in G$*, тогда и задача* (12) *имеет решение, т.е.*

$$f(x^*) = \min_{x \in G} f(x) = \min_{x \in \mathrm{R}^n} \Phi(x,G).$$

Замечание 4. *Т.к. множество* G *замкнуто, то в лемме* (1)*:*

$$\min_{x \in G} f(x) = \inf_{x \in \mathrm{R}^n} \left(f(x) + \delta(x,G)\right) = \min_{x \in \mathrm{R}^n} \left(f(x) + \delta(x,G)\right).$$

[3]Напомним, что индикаторная функция множества может быть записана в виде $\delta(x,G) = \begin{cases} 0,\ x \in G \\ +\infty,\ x \notin G \end{cases}$

Если незамкнуто хотя бы одно из множеств G_i, $i \in [1,m]$, то незамкнуто и множество G. В этом случае вместо задачи $\min\limits_{x\in \mathrm{R}^n}(f(x)+\delta(x,G))$ надо рассматривать задачу $\inf\limits_{x\in \mathrm{R}^n}(f(x)+\delta(x,G))$. Такая ситуация возникает в методе внутренних (барьерных) ШФ: хотя множество G_i замкнуто, но функция $\delta(x,G_i)$ определена на $intG_i$, т.е. $\delta(x,G_i)=+\infty$, если $x \notin \mathrm{int}G_i$.

Так как $\inf\limits_{x\in \mathrm{R}^n}\Phi(x,G)$ – точная нижняя грань множества значений функции $\Phi(x,G)$, то $\inf\limits_{x\in \mathrm{R}^n}\Phi(x,G)=\min\limits_{x\in G} f(x)=f(x^*)$, где $\Phi(x,G)$ не определена в точке x^*.

Штрафные функции, рассматриваемые выше, обладают свойствами [6, 10]:

1) Пусть $\tilde{G}=\left(\bigcap\limits_{i=1}^{m}\tilde{G}_i\right)\bigcap\ \bigcap\limits_{j=1}^{l}G_{m+j}\Bigg)$. Здесь $G_i=\{x\in R^n|\varphi_i\le 0\}$, если $i=\overline{1,m}$, и $G_{m+j}=\{x\in R^n_j|h_j=0\}$, если $j=\overline{1,l}$. При этом $\tilde{G}_i=G_i$ для внешних ШФ и $\tilde{G}_i=\mathrm{int}G_i$ для внутренних ШФ[4]. Если для некоторых $i\in[1,m]$ $P_i(\tau,\varphi_i)$ – внутренние ШФ, то $\tilde{G}$ – незамкнутое множество, если $\forall i\in[1,m]$ $P(\tau,\varphi_i)$ – внешние ШФ, то $\tilde{G}=G$ – замкнутое множество. Тогда

$$\delta(x,\tilde{G})=\sum_{i=1}^{m}\delta(x,\tilde{G}_i)+\sum_{i=m+1}^{m+l}\delta(x,G_i). \tag{13}$$

2) Обозначим $\delta_k(x,G)=\sum_{i=1}^{m+l}\delta_k(x,G_i)$, где $\delta_k(x,G_i)=P_i(\tau_k,\varphi_i)$, $i=\overline{1,m}$, $\delta_k(x,G_{m+j})=P_{m+j}(\tau_k,h_j)$, $j=\overline{1,l}$. Тогда:

$$\delta(x,\tilde{G})=\lim_{k\to\infty}\ \delta_k(x,G),\ \tau_k\to 0.$$

Рассмотрим задачу

$$\min F(x,\tau),\ x\in \mathrm{R}^n,\ \tau>0. \tag{14}$$

Теорема 3. *Предположим, что существует $x^*\in G$ – решение задачи (1) и множество $G_1=\{x\in \mathrm{R}^n : F(x,\tau)\le F(x_0,\tau),\ x_0\in G\}$ – компакт,*

[4]Функция $P_i(\tau,\varphi_i(x))$ внешняя ШФ, если $\lim\limits_{\tau\to 0}P_i(\tau,\varphi_i(x))=\delta(x,G_i)$; функция $P_i(\tau,\varphi_i(x))$ внутренняя ШФ, если $\lim\limits_{\tau\to 0}P_i(\tau,\varphi_i(x))=\delta(x,\mathrm{int}G_i)$

а также справедливо условие перестановки операций:

$$\lim_{k\to\infty}\min_{x\in\mathrm{R}^n}\left(f(x)+\delta_k(x,G)\right)=\inf_{x\in\mathrm{R}^n}\lim_{k\to\infty}\left(f(x)+\delta_k(x,G)\right),\ \tau_k\to 0,\ k\to\infty. \tag{15}$$

Тогда существует $x(\tau)$ *– решение задачи* (14) *такое, что* $x^*=\lim\limits_{\tau\to 0}x(\tau)$ *и* $F(x^*,0)=\lim\limits_{\tau\to 0}F(x(\tau),\tau)=f(x^*)$.

Доказательство. Так как множество G_1 – компакт, то существует $x(\tau)$ – решение задачи (14). Полагая $\lim\limits_{k\to\infty}\tau_k=0$, рассмотрим

$$\lim_{k\to\infty}\min_{x\in\mathrm{R}^n}F(x,\tau_k)=\lim_{k\to\infty}\min_{x\in\mathrm{R}^n}\left(f(x)+\delta_k(x,G)\right).$$

Используя условие перестановки (15) операций, лемму 1 и замечание 4, получим:

$$\begin{aligned}\lim_{k\to\infty}\min_{x\in\mathrm{R}^n}\left(f(x)+\delta_k(x,G)\right)= &\inf_{x\in\mathrm{R}^n}\lim_{k\to\infty}\left(f(x)+\delta_k(x,G)\right)=\\ =&\inf_{x\in\mathrm{R}^n}\left(f(x)+\delta(x,\tilde{G})\right)=\min_{x\in G}f(x)=f(x^*),\end{aligned}$$

где $\tilde{G}$ может быть незамкнутым множеством.
Из этих равенств следует, что $\lim\limits_{\tau\to 0}x(\tau)=x^*$ и $\lim\limits_{\tau\to 0}F(x(\tau),\tau)=F(x^*,0)=f(x^*)$. □

Замечание 5. *Для того, чтобы доказанную теорему можно было применять для анализа решения конкретных задач, необходимо убедиться, что:*

1) *в случае ограничений-неравенств выполнены условия* $\lim\limits_{\tau\to 0}P_i(\tau,\varphi_i)=\delta(x,\tilde{G}_i)$, $i=\overline{1,m}$ *для внешних и внутренних ШФ, для функций дифференциального барьера;*

2) *в случае ограничений-равенств для используемых внешних ШФ выполнены условия* $\lim\limits_{\tau\to 0}P_i(\tau,h_j)=\delta(x,G_i)$, $i=m+j$, $j=\overline{1,l}$;

3) *справедливо условие перестановки операций для* $\lim\limits_{k\to 0}(\cdot)$ *и* $\inf\limits_{x\in\mathrm{R}^n}(\cdot)$.

Выше были приведены требования (3) и (4), которым должны удовлетворять внутренние и внешние ШФ для ограничений $\varphi_i(x)\le 0$, $i=\overline{1,m}$ и внешние ШФ для ограничений $h_j(x)=0$, $j=\overline{1,l}$. При выполнении

этих требований выполнены условия для $\delta(x, \tilde{G}_i),\ i = \overline{1,m}$ и $\delta(x, G_i)$, $i = m + j,\ j = \overline{1,l}$, [1].

Для функций ДБ условия для $P(\tau, y)$ можно доопределить:

$$P(\tau, y) = \begin{cases} \leq c, \text{если } y \leq 0; \\ +\infty,\ y > 0. \end{cases}$$

Здесь $x \in G_i \bigcap Q$, Q – компакт, $c_i \in R_+$ – число. Указанные выше условия непротиворечивы, т.к. точки $x \notin G_i$ в методе ДБ не рассматриваются. Тогда

$$\delta(x, G_i) = \lim_{\tau \to 0} P(\tau, \varphi_i) = \begin{cases} 0,\ x \in G \\ +\infty,\ x \notin G, \end{cases}$$

Т.е. выполнены условия для $\delta(x, G_i),\ i \in [1, m]$.

Лемма 2. *[1] Пусть в методе внутренних ШФ* (4) *выполнены условия* $F(x, \tau_k) \geq f(x)$, $x \in \text{int}G$, $\lim\limits_{k\to\infty} \tau_k = 0$ *и* $\lim\limits_{k\to\infty} F(x, \tau_k) = f(x)$, $x \in G$. *Тогда*

$$\lim_{k\to\infty} \inf_{x\in G} F(x, \tau_k) = \inf_{x\in G} \lim_{k\to\infty} F(x, \tau_k) = \inf_{x\in G} f(x).$$

Лемма 3. *[1] Пусть в методе внешних ШФ выполнены условия* (3). *Тогда*

$$\lim_{k\to\infty} \min_{x\in \mathrm{R}^n} F(x, \tau_k) = \min_{x\in \mathrm{R}^n} \lim_{k\to\infty} F(x, \tau_k) = \min_{x\in G} f(x).$$

Лемма 4. *В методе дифференциального барьера (ДБ) справедливо равенство:*

$$\lim_{k\to\infty} \min_{x\in G} F(x, \tau_k) = \min_{x\in G} \lim_{k\to\infty} F(x, \tau_k) = \min_{x\in G} f(x).$$

Доказательство. Пусть $F(x, \tau) = f(x) + \sum_{i=1}^{m} P_i(\tau, \varphi_i) \equiv f(x) + P(\tau, \varphi)$. Из условия (9) следует, что $\exists N : \forall k \geq N$

- $|P(\tau_k, \varphi)| = |F(x, \tau_k) - f(x)| \leq \frac{\varepsilon}{3},\ x \in G$;
- для любого $\varepsilon > 0$ найдется $x_0 \in G$ такой, что $F(x_0, \tau_k) \leq \min\limits_{x\in \mathrm{R}^n} F(x, \tau_k) + \frac{\varepsilon}{3}$.

Пусть N – некоторый достаточно большой номер, тогда при произвольном $k \geq N$ оценим разность:

$$\begin{aligned} \left| \min_{x\in G} f(x) - \lim_{k\to 0} \min_{x\in G} F(x, \tau_k) \right| \leq & \left| \min_{x\in G} f(x) - \min_{x\in G} F(x, \tau_k) \right| + \frac{\varepsilon}{3} \leq \\ \leq & |f(x_0) - F(x_0, \tau_k)| + \frac{\varepsilon}{3} + \frac{\varepsilon}{3} \leq \varepsilon \end{aligned}$$

Полученная оценка доказывает лемму. □

Доказанная теорема 3 может иметь широкое применение: она справедлива как для гладких, так и для выпуклых задач, а также в тех случаях когда функции $\varphi_i(x)$, $i = \overline{1,m}$ - внешние и внутренние ШФ и функции ДБ.

4. Преобразования ШФ и функций ДБ

В работе [10] предложен метод гладких ШФ для решения гладкой задачи МП вида:

$$\begin{gathered} f(x) \to \min\limits_{x \in G}; \\ G = \left\{ x \in R^n | \varphi_i(x) \le 0, i = \overline{1,m}, x_j \ge 0, j = \overline{1,n} \right\}. \end{gathered} \tag{16}$$

В этом методе, названном «методом обратных связей» (МОС), задача (16) сводится к решению системы нелинейных уравнений (СНУ). При построении МОС исходная ШФ $P(\tau, \varphi)$ функции $\varphi(x) \le 0_m$ последовательно подвергалась операциям:

- дифференцирования – $\frac{dP(\tau,\varphi)}{d\varphi}$;
- построения функции $R(\tau, \lambda)$, $\lambda \ge 0$, обратной[5] к $\frac{dP(\tau,\varphi)}{d\varphi}$;
- интегрирования – $J(\tau, \lambda) = \int R(\tau, \lambda) d\lambda$.

При этом оказалось, что функции $\frac{dP(\tau,\varphi)}{d\varphi}$ и $R(\tau, \lambda), J(\tau, \lambda)$ при $\lambda = -\varphi$ являлись в свою очередь ШФ, барьерными или квазибарьерными функциями (функции ДБ).

Кроме указанной последовательности операций возможны и другие их варианты. Интерес к рассмотрению таких последовательностей операций заключается в том, что в результате их (операций) выполнения найдутся примеры новых штрафных (барьерных и квазибарьерных) функций. Приведем примеры:

Пример 3. *Пусть* $P(\tau, \varphi) = \tau \cdot e^{\frac{\varphi}{\tau}}$, *тогда* $\frac{\partial P}{\partial \varphi} = e^{\frac{\varphi}{\tau}}$, $R(\tau, \lambda) = \tau \ln \lambda$, $J(\tau, \lambda) = \tau\lambda(\ln \lambda - 1)$. *Тогда:*

- *функция* $\frac{\partial P}{\partial \varphi}$ – *внешняя ШФ;*

[5] Для функции $y = f(x)$, $x, y \in \mathrm{R}^1$ обратной будет $x = g(y)$, причем $g(f(x)) = x$. Если функции $f(x)$ и $g(y)$ дважды дифференцируемы, то $g'(y) = \frac{1}{f'(x)}$ и $g''(y) = -\frac{f''(x)}{(f'(x))^3}$ [11].

- *если* $\lambda = -\varphi$, *то функция* $\tau \ln(-\varphi)$ – *внутренняя (барьерная ШФ) для задачи* $\max f(x)$, $x \in G$;

- *функция* $\tau(-\varphi) \cdot (\ln(-\varphi) - 1)$ – *новая квазибарьерная функция.*

Квазибарьерными функциями будут также более простая на вид функция $J(\tau, \varphi) = \tau(-\varphi) \cdot \ln(-\varphi)$ *и функция* $P(\tau, \varphi) = \tau(-\varphi)^{\alpha} \ln(-\varphi)$, $0 < \alpha \leq 1$.

Пример 4. *Пусть* $P(\tau, \varphi) = \tau(-\varphi) \cdot \ln(-\varphi)$, *то* $\frac{\partial P(\tau,\varphi)}{\partial \varphi} = -\tau\left(\ln(-\varphi) + 1\right)$, $R(\tau, \lambda) = -\frac{1}{e} \cdot e^{-\frac{\lambda}{\tau}}$, $J(\tau, \lambda) = \frac{\tau}{e} \cdot e^{-\frac{\lambda}{\tau}}$. *Тогда*

- *функция* $\frac{\partial P}{\partial \varphi}$ – *барьерная ШФ;*

- *при* $\lambda = -\varphi$ *функция* $R(\tau, \lambda)$ – *внешняя ШФ из п.1 с точностью до множителя* $\frac{1}{e}$ *для задачи* $\max f(x)$, $x \in G$.

Пример 5. *Пусть* $P(\tau, \varphi) = -\tau \cdot \ln(-\varphi)$, *тогда* $\frac{\partial P}{\partial \varphi} = -\frac{\tau}{\varphi}$, $R(\tau, \lambda) = -\frac{\tau}{\lambda}$, $J(\tau, \lambda) = -\tau \ln \lambda$. *Очевидно, данная барьерная функция – «симметричная»:* $P(\tau, \varphi)$ *совпадает с* $J(\tau, \lambda)$ *при* $\lambda = -\varphi$, *а* $\frac{\partial P}{\partial \varphi} = -R(\tau, -\varphi)$. *При этом* $\frac{\partial P}{\partial \varphi}$ *и* $R(\tau, -\varphi)$ – *внутренние ШФ для задачи* $\max f(x)$, $x \in G$.

Пример 6. *Пусть* $P(\tau, \varphi) = -\frac{\tau}{\varphi}$, *то* $\frac{\partial P}{\partial \varphi} = \frac{\tau}{\varphi^2}$, $R(\tau, \lambda) = -\sqrt{\frac{\tau}{\lambda}}$, $J(\tau, \lambda) = -\sqrt{\tau\lambda}$. *Тогда*

- *функция* $\frac{\partial P}{\partial \varphi}$ – *барьерная ШФ;*

- *при* $\lambda = -\varphi$ *функция* $R(\tau, \lambda)$ – *барьерная ШФ для задачи* $\max\limits_{x \in G} f(x)$;

- *функция* $J(\tau, \lambda)$ – *квазибарьерная степенная функция.*

Пример 7. *Пусть* $P(\tau, \varphi) = -\tau \cdot (-\varphi)^{0,5}$, *то* $\frac{\partial P}{\partial \varphi} = \frac{\tau}{2\sqrt{-\varphi}}$, $R(\tau, \lambda) = -\frac{\tau^2}{4\lambda^2}$, $J(\tau, \lambda) = \frac{\tau^2}{4\lambda}$.

Тогда

- *функция* $\frac{\partial P}{\partial \varphi}$ – *барьерная ШФ;*

- *при* $\lambda = -\varphi$ *функция* $R(\tau, \lambda)$ – *барьерная ШФ для задачи* $\max f(x)$, $x \in G$;

- *функция* $J(\tau, \lambda)$ – *барьерная ШФ.*

Приведенные примеры показывают, что операции дифференцирования, интегрирования и определения обратных функций превращают штрафные, барьерные и квазибарьерные функции в одну из функций из указанных классов. Такое свойство предложенных преобразований позволяет находить новые ШФ и функции ДБ неизвестные ранее на практике, например, функцию (10). Кроме указанной последовательности преобразований функций из [10] можно строить и другие их последовательности, однако эта тема выходит за границы нашей работы.

Исследование Чернова А.В. выполнено при финансовой поддержке РФФИ в рамках научного проекта 18-31-00219.

Список литературы

[1] А.Г. Сухарев, А.В. Тимохов, В.В. Федоров. Курс методов оптимизации // М.: ФИЗМАТЛИТ, 2005.

[2] Э. Полак. Численные методы оптимизации // М.: Мир, 1974.

[3] Ф. Гилл, У. Мюррей, М. Райт. Практическая оптимизация // М.: Мир, 1985.

[4] M. Hamala. Quasibarriar method for convex programming // IX International simposium on mathematical programming, – Budapest, 1976.

[5] А.С. Хохлов. Квазибарьерные штрафные функции // Автоматика и телемеханика, – М.: 1979, – Вып. 5, – С.188–191.

[6] В.Г. Жадан. Методы оптимизации. Часть 2. Численные алгоритмы : учебное пособие // М.: МФТИ, 2015.

[7] А. Фиакко, Г. Мак-Кормик. Нелинейное программирование. Методы последовательной безусловной минимизации // М.: Мир, 1972.

[8] В.Г. Жадан. Методы оптимизации. Часть 1. Введение в выпуклый анализ и теорию оптимизации : учебное пособие // М.: МФТИ, 2014.

[9] Ю.С. Попков. Теория макросистем: Равновесные модели // М.: УРСС, 2013.

[10] Е.А. Умнов, А.Е. Умнов. Методы параметрической линеаризации, использующие штрафные функции со всюду обратимой производной для решения пар двойственных задач // Труды МФТИ, – М.: МФТИ, 2011 – т.3, № 1.

[11] Л.Д. Кудрявцев. Курс математического анализа, т.1 // М.: Высшая школа, 1981.

Penalty, barrier, quasi-barrier functions and functions inverse to them

Birjukov A.G., Chernov A.V., Chernova Yu.G., Sharovatova Yu.I.

The methods of external penalty functions, internal penalty functions and quasi-barrier functions for solving problems of mathematical programming are considered. New quasi-barrier functions are proposed. The theorems of convergence of the indicated methods to the solution of mathematical programming problems are proved. The properties of these functions are considered for their transformations: differentiation, integration, construction of functions inverse to them.

Keywords: external penalty functions, internal penalty functions, barrier penalty functions, inverse functions, quasi-barrier functions, mathematical programming problem, differential barriers, power differential barriers, entropy differential barriers, convergence of differential barriers methods to solving mathematical programming problems.

Применение алгоритма Витерби к восстановлению стертого фрагмента музыкального произведения

Ботхолов А.Ж.

Приведены методы восстановления стертого фрагмента музыкальной композиции, основанные на алгоритме Витерби, а именно метод, основанный только на алгоритме Витерби, метод, использующий кроме алгоритма Витерби, также смещенную высоту и смещенную длительность, метод, основанный на алгоритме Витерби и принципах музыкальной гармонии. Выяснен метод с наилучшим результатом (алгоритм Витерби, совмещенный с методом музыкальной гармонии), введен способ оценки похожести фрагментов, приведены примеры мелодий, такты которых были лучше всего восстановлены. Все методы вписаны в программу на языке Java, которая решает заданную задачу, кроме того используется программа GuitarPro, помогающая перевести всю необходимую информацию о мелодии в текстовый формат.

Ключевые слова: алгоритм Витерби, смещенная высота, смещенная длительность, гармонизация аккордами.

Введение

За последнее время было проведено много исследований, связанных с музыкальными произведениями — поиск закономерностей в мелодиях, генерация последовательностей нот на основе выбранного стиля, использование математических моделей для создания музыкальных отрывков и другие. Всегда казалось, что последовательность звучащих нот в мелодии подчиняется закону. Возник вопрос: насколько хорошо пригоден алгоритм Витерби к восстановлению стертого фрагмента музыкальной композиции?

В данной работе мы имеем дело с файлами, содержащими нотные записи музыкальных произведений (мелодии одноголосные). Существуют специальные программы (редакторы нот), с помощью которых со-

здаются такие файлы в формате MusicXML. Этот формат достаточно популярен среди большого числа программ-редакторов, которые дают возможность читать и вносить изменения в нотные записи. Для проверки пригодности алгоритма Витерби к восстановлению стертого фрагмента будет использоваться запись музыкального произведения в виде MusicXML файла, обладающий определенной структурой.

Структура MusicXML файла

Используемые методы восстановления фрагмента музыкальной композиции работают с файлами в формате MusicXML. Данный вид файлов состоит из набора тегов. Перечислим некоторые из них: <note>, <octave>, <type>, <alter>, <string>, <fret>, <measurenumber =”2”>, отвечающие за ноту, октаву, длительность, присутствие или отсутствие диеза, название струны, лад, номер такта соответственно. Всего существует 55 нот плюс пауза, которую считаем за отдельную ноту.

Методы восстановления фрагмента

Музыкальная композиция представляет собой последовательность тактов. Такт есть множество нот и пауз, находящееся между 2-мя вертикальными линиями (тактовыми чертами) на нотном стане. Нотный стан есть некая разметка, где записываются ноты.

Допустим в музыкальной композиции стерли некоторое множество идущих друг за другом тактов. Наша задача восстановить эти такты с наименьшей погрешностью, то есть узнать количество стертых нот и значения этих нот. В работе были испробованы два способа построения недостающих нот, на каждый из которых накладывались различные методы.

Первый способ состоит в том, что каждый стертый такт мы рассматриваем как последовательность 16-ти нот длительностью, равной размеру такта, деленного на 16. После получения в каждом стертом такте 16-ти нот есть возможность объединять одинаковый ноты, идущие друг за другом в одну ноту длительностью, равной сумме длительностей суммированных нот. Способ же реализуется одним из двух вариантов: первый учитывает длительности нот в оставшихся не стертых тактах, то есть если в оставшихся тактах есть нота C длительностью не больше 1/2, то

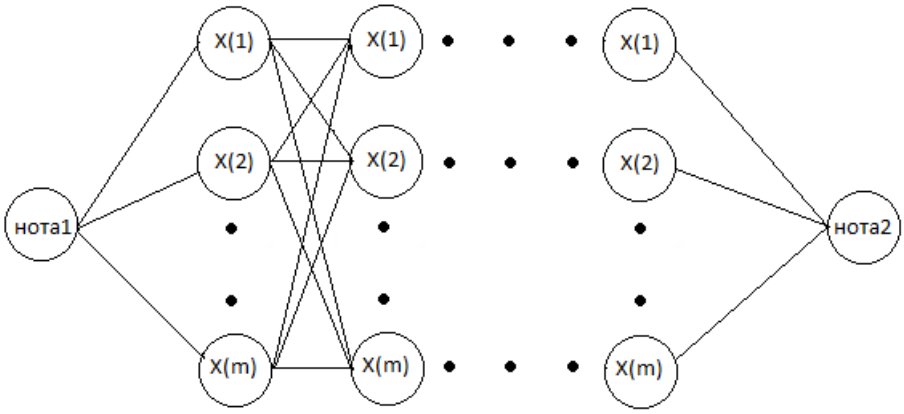
и в восстанавливаемых тактах длительность ноты C не должна превышать 1/2; второй вариант не учитывает длительности нот.

Второй способ основан на том, что количество нот в каждом такте стертого фрагмента равно тому количеству нот, который наиболее часто встречается в не стертых тактах. После выяснения данного количества определяем наиболее вероятные длительности восстановленных нот. (Как казалось, второй способ лучше решает задачу, нежели первый, поэтому в дальнейшем используем только его.)

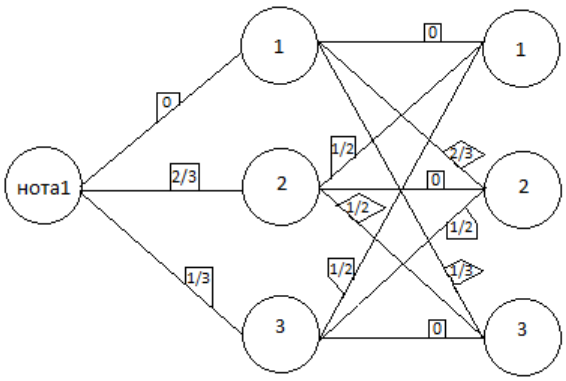
Как было сказано, на эти способы накладывались различные методы решения задачи, причем был испробован некий «симбиоз» методов, то есть их одновременное использование. Рассмотрим эти методы.

Алгоритм Витерби

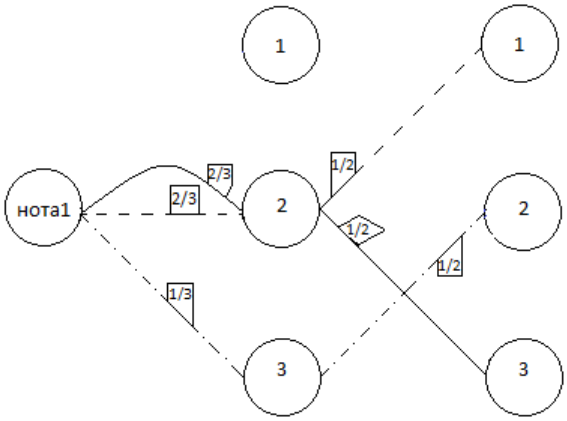
Допустим в мелодии стерли фрагмент, состоящий из нескольких тактов (от 1 до 7). Чтобы восстановить фрагмент, исследуем оставшиеся в мелодии такты с нотами. Предполагаем, что мы оценили, сколько находилось нот во всех тактах фрагмента. Пусть это количество равно N, а ноты, которые есть в оставшихся фрагментах — $x(1)$, $x(2)$, $\ldots$, $x(m)$; то есть разных нот в оставшихся тактах мелодии равно m. Нам известна нота перед фрагментом и после него, обозначим их нота1 и нота2 соответственно. После нота1 может стоять одна из m нот причем с определенной вероятностью (программа вычисляет для каждой пары нот, идущих друг за другом, сколько раз она встречается в не стертых тактах, а затем находит вероятности появления этих пар), после каждой из этих m нот может идти одна из этих же m нот и т. д. Фрагмент же заканчивается нотой, которая переходит в нота2. Таким образом существует большое число последовательностей нот, начинающихся в нота1 и заканчивающихся в нота2. Представить эту схему можно с помощью следующего рисунка



Ясно, что количество столбцов с нотами $x(1)$, $x(2)$, ..., $x(m)$ на рисунке равно числу N. Наша задача состоит в нахождении наиболее вероятной последовательности (пути). Для восстановления используется алгоритм Витерби. Алгоритм состоит том, что на каждом шаге восстановления фрагмента (шаг — переход с одного столбца на другой) для каждой ноты из столбца находим наиболее вероятный путь, выходящий из нота1 и заканчивающийся в данной ноте (следует отметить, что путей может быть несколько). На следующем шаге для вычисления для каждой ноты наиболее вероятного пути используем вычисленные пути на предыдущем шаге. Для понимания процесса разберем пример двух шагов алгоритма для трёх нот. На рисунке на ребрах стоят вероятности появления ноты на правом конце ребра после ноты на левом конце.

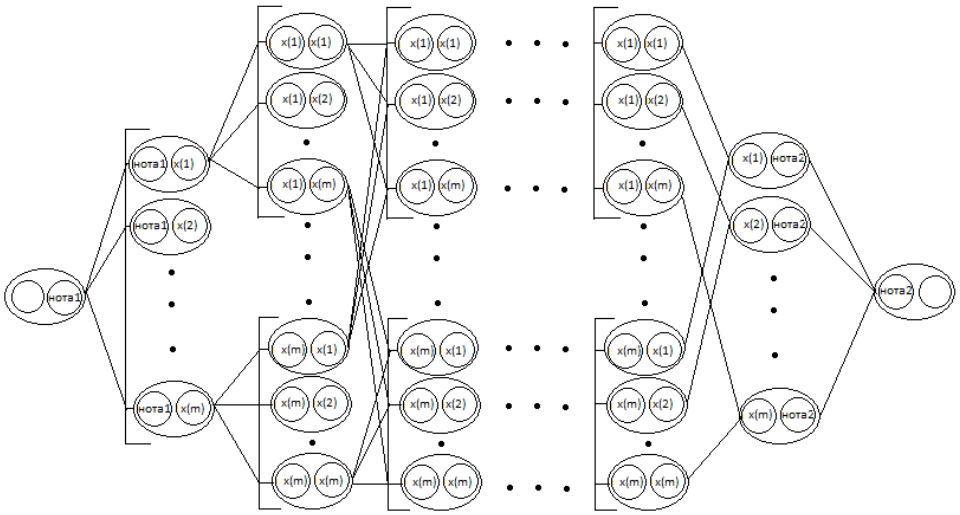


На первом шаге алгоритма для каждой ноты из первого столбца наиболее вероятным путем является ребро, соединяющее нота1 и рассматриваемую ноту, на следующем же шаге для нот второго столбца имеем следующее:



Для ноты 1 обозначен путь с наибольшей вероятностью пунктиром, для ноты 2 — штрихпунктиром, для ноты 3 — обычными линиями. Таким образом, мы рассмотрели вариант метода, который учитывает вероятности появления одной ноты после другой. Этот вариант назовем 1-грамм.

По аналогии определяем вариант 2-грамм, который учитывает вероятности появления одной ноты после двух других. Схематично этот метод можно изобразить так



В работе рассмотрены 4 типа реализации алгоритма: 1-грамм, 2-грамм, 3-грамм, 4-грамм.

Метод, основанный на использовании смещенной высоты и смещенной длительности ноты с алгоритмом Витерби

Смещенная высота ноты — это число, которое вычисляется как разность полутонов данной ноты и предыдущей ноты. Для всех нот мелодии, за исключением первой ноты, вводится понятие смещенной высоты. Подход для описания нот, где вместо высот указывается смещенная высота, дает следующее преимущество — в случае неправильного восстановления в мелодии какой-то отдельной ноты ошибка отразится только на двух соседних смещенных высотах, но не повлияет на оставшуюся часть мелодии.

Покажем на примере отрывка песни "Елочка" (рис.1), как вычисляется смещенная высота. Под каждой нотой кроме первой указаны смещенные высоты. Положительные числа стоят, если предыдущая нота ниже данной, а отрицательные — в противном случае.

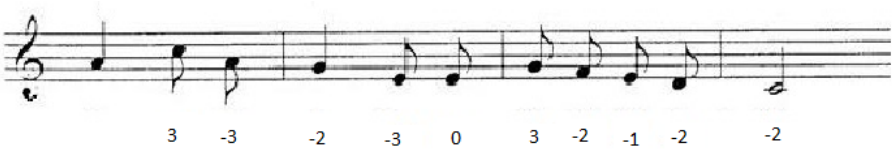


Рис.1 Смещенные высоты нот

Смещенная длительность ноты — это число, которое вычисляется как отношение длительности данной ноты к длительности предыдущей ноты или паузы. Для первой ноты фрагмента смещенная длительность не вычисляется (по аналогии со смещенной высотой). Как определяется смещенная длительность для фрагмента песни "Елочка" показано на рис. 2.

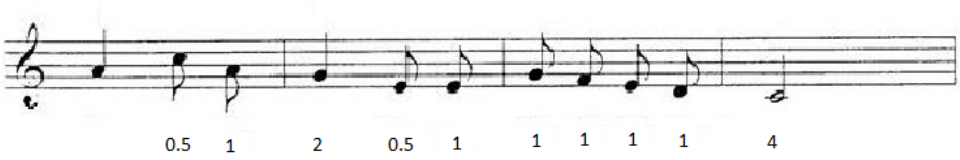


Рис.2 Смещенные длительности нот

Здесь же реализация алгоритма Витерби происходит так же как и в предыдущем пункте, только вместо нот используются смещенные высоты и смещенные длительности нот.

Алгоритм Витерби плюс метод музыкальной гармонии

В теории музыки различают два главных способа изложения мелодии: полифонический — мелодия выполняется несколькими голосами, и каждый отдельный голос может выполнять свою мелодию независимо от других голосов, гомофонный склад — мелодия выполняется одним из голосов, а все остальные голоса выполняют роль аккомпанемента.

Так как мы работаем с одноголосными мелодиями, то у всех рассматриваемых мелодий присутствует гомофонный склад. Наша задача подобрать аккомпанемент к основной мелодии или, другими словами, гармонизировать мелодию аккордами.

Из теории музыкальной гармонии известно, что почти любая мелодия может быть гармонизирована аккордами. Гармонизация мелодии начинается с вычисления тональности мелодии. Понятие тональности состоит из тоники — главной ноты мелодии и гармонического закона, с помощью которого по данной ноте достраивается шесть основных нот. Остальные же ноты могут встречаться в мелодии как дополнительные. Дальше для каждого такта мелодии подбираем аккорды, которые бы гармонично звучали вместе с мелодией. С помощью алгоритма Витерби вычисляем наиболее вероятные аккорды стертого фрагмента и, исходя из полученных аккордов, для восстановления тактов рассматриваем только те ноты, которые соответствуют определенному аккорду. То есть в процессе построения фрагмента мы используем гораздо меньшее число нот.

Схема реализации методов

Предложенные методы реализованы в программе на языке Java. Выбранные мелодии в формате midi переводятся с помощью программы GuitarPro в формат MusicXML, с которым, собственно, и работает программа.

Программа считывает с файла всю необходимую информацию про ноты: значение, длительность, октава и т.д. В зависимости от метода, находит нужные вероятности нотных последовательностей, получает некоторое множество чисел, которые заменяет на теги для нового файла в формате MusicXML. С помощью GuitarPro переводим полученный файл в файл формата Midi.

Процесс можно схематически изобразить следующим образом:

$$\begin{aligned}\text{Midi(1)} \longrightarrow \text{GuitarPro} \longrightarrow \text{MusicXML(1)} \longrightarrow \text{Programm} \longrightarrow \\ \longrightarrow \text{MusicXML(2)} \longrightarrow \text{GuitarPro} \longrightarrow \text{Midi(2)}\end{aligned}$$

Программа в отдельный файл записывает вероятности появления ноты после одной ноты, двух, трех, четырех нот в зависимости от варианта алгоритма и работает с этим файлом для получения результата.

Результаты работы методов

Так как в музыкальной области нет формальной оценки сходства мелодий (в основном оно определяется на слух), для оценки точности восстановления фрагмента была введена формула коэффициента похожести

$$k = \frac{m \cdot \min(n_1, n_2)}{n_1 \cdot \max(n_1, n_2)},$$

k — коэффициент похожести фрагментов,
m — число совпавших нот,
n_1 — число нот фрагмента оригинальной музыкальной композиции,
n_2 — число нот фрагмента, восстановленного программой.

Результатом вычисления коэффициента похожести фрагментов является значение из отрезка $[0, 1]$, где 0 соответствует полному несовпадению фрагментов, а 1 — их полному совпадению. Как видно коэффициент зависит не только от количества «угаданных» нот, но и от того, из скольких нот, как предполагалось, состоит восстанавливаемый фрагмент. Таким образом, если программа указала достаточно много правильных нот, но полученное количество нот, из которых состоит фрагмент, оказалось далеким от истинного, то коэффициент может оказаться достаточно малым по сравнению с коэффициентом в случае правильно указанного количества нот. Этот факт нужно учитывать и понимать, что коэффициент может получиться «плохим» несмотря на то, что количество угаданных нот велико.

В таблице 1 представлены 5 мелодий, в которых лучше всего восстановились ноты (были удалены 2 такта). Мелодии из таблицы можно оценить по полученным коэффициентам при использовании различных вариаций Алгоритма Витерби, метода смещенной высоты и смещенной длительности нот с алгоритмом Витерби, совмещения Алгоритма Витерби и метода музыкальной гармонии.

Таблица 1

Композиции	Алгоритм Витерби				Смещенн.+ Витерби	Вит.+муз. гармония
	1-грамм	2-грамм	3-грамм	4-грамм		
1	0,57248	0,62355	0,62124	0,75913	0,51012	0,85624
2	0,23779	0,37933	0,39408	0,41068	0,36560	0.57890
3	0,23933	0,20453	0,37692	0,37892	0,34494	0.31222
4	0,12672	0,48744	0,24265	0,35827	0,21447	0.39062
5	0,11681	0,17479	0,31733	0,29333	0,25462	0.29444

Как можно заметить, наибольшие коэффициенты похожести имеет способ, совмещающий Алгоритм Витерби и метод музыкальной гармонии. В таблице использованы следующие мелодии: мелодия «Led Zeppeline — Stairway to heaven», мелодия песни «AC/DC — Back In Black», мелодия песни «Metallica — Nothing else matters», мелодия «Бах — Токката», мелодия «Моцарт — Свадьба Фигаро».

Наименьшие коэффициенты имеет метод алгоритм Витерби 1-Грамм. Также можно сделать вывод, что в большинстве случаев с увеличением количества стертых тактов фрагмента уменьшается точность восстановления нот.

Также хотелось бы отметить, что в данной работе было проведено исследование пригодности именно алгоритма Витерби для восстановления стертого фрагмента. Но если рассмотреть другие методы для восстановления, то как минимум есть один способ, который при дополнительно наложенных условиях может полностью восстановить стертый фрагмент. А именно метод, где мы повторяем мелодию несколько раз и стираем несколько тактов в пределах одной мелодии. Затем если перед стертым фрагментом есть не стертый кусок, то находим данный кусок в мелодии (он обязательно найдется, так как мы повторили мелодию несколько раз) и смотрим какая нота стоит после него и ставим эту ноту в стертый фрагмент, потом находим следующую ноту фрагмента аналогичным способом. Если же перед стертым фрагментом нет нот, то смотрим на не стертый кусок после данного фрагмента и также находим это кусок в мелодии, затем восстанавливаем ноты с конца стертого фрагмента.

Список литературы

[1] Рябинович Е. В. Сжатое представление музыкальных сигналов для поиска музыкальных произведений, 2002.

[2] Славщик А.А. История алгоритмической музыкальной композиции.

[3] MusicXML format. URL: http://www.makemusic.com/musicxml

[4] Typke R. Music Retrieval Based on Melodic Similarity. Ph.D. Thesis, University of Utrecht, 2007.

[5] Rippling: Meta–level Guidance for Mathematical Reasoning.

[6] Зарипов Р.Х. Кибернетика и музыка. М: Изд-во «Знание», 1963.

[7] Вахромеев В.А. Элементарная теория музыки. М.: МУЗГИЗ, 1961.

Applying of the Viterbi algorithm to the recovering of an erased fragment of a musical composition

Botkholov A.J.

The problem of the recovering of an erased fragment of a musical composition based on the Viterbi algorithm, namely, the method based only on the Viterbi algorithm, the method using the Viterbi algorithm and also shifted height and shifted duration, the method based on the Viterbi algorithm and the principles of musical harmony are obtained. The method with the best result was found (Viterbi algorithm combined with the method of musical harmony), the method for estimating of the similarity of fragments was introduced, examples of melodies, the tacts of which were best recovered, are given. All methods are written into the program in the Java language, which solves the given task, in addition, the program GuitarPro is used, which helps to translate all the necessary information about the melody into a text format.

Keywords: the Viterbi algorithm, shifted height, shifted duration, chord harmonization.

Часть 2.
Специальные вопросы теории интеллектуальных систем

Об одном методе персонализации поиска информации

Рыжов А.П., Огородников Н.М.

В работе рассматривается задача персонализации нечётких пользовательских понятий с помощью действия модификаторов на данные нечёткие множества. Исследован и модифицирован существующий алгоритм персонализации, для новой его версии получена оценка числа выполняемых итераций.

Ключевые слова: нечёткие множества, персонализация.

1. Введение

За последние несколько лет произошло качественное изменение использования информации людьми в решении повседневных задач. Количество накопленной информации в доступном для электронной обработки виде изменяется зеттабайтами (триллион гигабайт). Так, в недавно проведенном исследовании ведущих компаний Seagate (www.seagte.com) и IDC (https://www.idc.com) [1] подсчитано, что в 2016 году объем данных измерялся 16 Збайт, а к 2025 году этот показатель увеличится до 163 Збайт. Там же утверждается, что к 2025 году около 20% всей информации будут играть критически важную роль в повседневной жизни, а примерно 10% этих данных будут "сверхкритичными". Кроме того, прогнозируется, что в 2025 году почти 20% генерируемых данных будут представлять собой информацию, получаемую в режиме реального времени. Количество активных SIM-карт, используемых в телефонах, смартфонах и планшетах, впервые в истории превысило число живущих на Земле людей в 2014 году и продолжает расти [2]. Количество сервисов, используемых в повседневной жизни, также растет: трудно представить себе современного человека, не использующего поиск в интернет, новостные агрегаторы, социальные сети, электронные карты, заказ билетов, и многие другие ставшие привычными сервисы.

В сложившейся ситуации способы взаимодействия пользователя с информационными ресурсами также меняются. Одним из признанных трендов таких изменений является персонализация [4-15]. Компании видят существенное влияние персонализации на многие сферы бизнеса от маркетинга и продаж до оптимизации операционной деятельности и принятия стратегических решений. Это справедливо для многих индустрий от массовой торговли до индустрии моды. В этой связи разработка и исследование моделей и алгоритмов персонализации видится востребованной задачей.

Впервые идея алгоритма персонализации была предложена в работе [16], в [23] дана ее детализация и рассмотрены варианты оптимизации. Примеры использования таких моделей в компьютерном обучении, оптимизации новостных лет, социальных сетях представлены в работах [17-22].

Базовой для многих моделей персонализации является задача поиска информации. В общем виде задача поиска информации для человеко-компьютерных систем рассмотрена в [24]. В [25] предложен и изучен один из возможных алгоритмов персонализации на основе итерационного уточнения функций принадлежности. В качестве механизма такого уточнения предлагалось использовать сдвиг функций принадлежности, однако вопрос определения шага для такого сдвига остался открытым, что не позволяет использовать алгоритм в практических задачах. В данной работе предлагается алгоритм, свободный от такого недостатка.

Работа организована следующим образом. В разделе 2 приведена формализация задачи поиска информации. Алгоритм персонализации на основе известного алгоритма нечеткой кластеризации *c-means* представлен в разделе 3, его обобщение для любого эмпирического распределения объектов в универсальном множестве описано в разделе 4. В заключении (раздел 5) подведены краткие итоги исследования и сформулированы связанные с ними задачи.

2. Формализация задачи

Рассмотрим конечное множесто объектов, каждый из которых характеризуется некоторым числовым параметром. Например, это могут быть товары с ценой в качестве характеристики. Важно лишь то, что характеристика объекта принимает значения из ограниченного подмножества числовой прямой, таким образом, считаем, что на множестве объектов задан порядок. Будем называть всё множество объектов базой данных

или универсальным множеством и обозначим его за U. Считаем также, что объекты бывают (условно) *большими* и *маленькими*, то есть, характеризуются своей величиной, а их характеристика описывается словами естественного языка. Это порождает нечёткость, поскольку понятия *большой* и *маленький* не поддаются строгому определению, комфортному каждому человеку. По этим соображениям рассмотрим нечёткие множества *больших* и *маленьких* объектов с функциями принадлежности μ_1 и μ_2 соответственно, определёнными на универсальном множестве U. Ясно, что для разных людей (разных пользователей базы данных) функции μ_1 и μ_2 будут различаться в силу того, что каждый по-своему определяет *большие* и *маленькие* объекты. Далее речь пойдёт о том, как подобрать набор объектов из базы, комфортный конкретному пользователю, другими словами, как построить подходящие ему функции принадлежности μ_1 и μ_2.

В соответствии с принятой в теории нечётких множеств терминологией [21] рассмотрим лингвистическую переменную и её терм-множество. Оно состоит из базовых терминов и их комбинаций с модификаторами. В нашем случае есть два базовых термина - *большие* и *маленькие* объекты. На данном этапе ограничимся четырьмя модификаторами - *значительно больше*, *слегка больше*, *значительно меньше* и *слегка меньше*, однако, ниже рассмотрим и произвольное их количество. Так как данные модификаторы представляют собой конструкции естественного языка, будем считать, что именно они используются в запросах пользователя к базе данных.

Теперь можно переформулировать задачу следующим образом: пусть у нас есть конечное универсальное множество объектов U, требуется задать на нём начальные функции принадлежности μ_1 и μ_2, отвечающие *большим* и *маленьким* объектам, и сформулировать правило их изменения под действием каждого из модификаторов.

В работе [26] описан возможный вариант алгоритма построения и модификации функций μ_1 и μ_2, основывающийся на нечёткой кластеризации *c-means*. Мы сначала рассмотрим этот алгоритм подробнее, а затем предложим способ его улучшения.

3. Алгоритм персонализации на основе *c-means*

При данном подходе к универсальному множеству U применяется алгоритм нечёткой кластеризации *c-means* с количеством кластеров, равным двум. Следовательно, для каждого объекта x из U определены $\mu_1(x)$ и

$\mu_2(x)$ - коэффициенты его принадлежности нечётким множествам *больших* и *маленьких* объектов. Затем, для всех x, не меньших центра кластера c_1, $\mu_1(x)$ полагается равным 1, а $\mu_2(x)$ - равным 0, а для всех x, не больших центра кластера c_2, значение $\mu_1(x)$ берётся равным 0, а $\mu_2(x)$ равным 1. Теперь функция $\mu_1(x)$ является неубывающей на всём U, а функция $\mu_2(x)$ - невозрастающей, что согласуется с порядком, определённым на элементах множества U. Множество тех x, для которых $\mu_1(x) = 1$, обозначим через U_1, а множество тех x, для которых $\mu_2(x) = 1$ - через U_2. Построенные функции принадлежности изображены на картинке ниже.

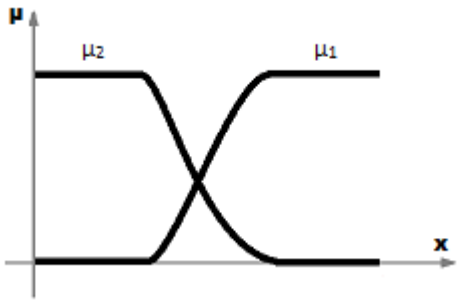


Рис.1. Функции принадлежности μ_1 и μ_2

Сам же алгоритм персонализации или определения действия модификаторов опирался на простое соображение: если при данной функции принадлежности множества *больших* объектов мы хотим получить функцию принадлежности множества объектов, которые *значительно больше*, то нас точно не интересуют объекты из U_2, эталонные *маленькие* объекты. Значит, при определении действия модификатора *значительно больше* будем полагать новые функции $\mu_1(x)$ и $\mu_2(x)$ равными 0 и 1 соответственно для всех x из U_2, а для остальных x снова применяем нечёткую кластеризацию и переопределяем коэффициенты принадлежности для всех $x \geq c_1$ и всех $x \leq c_2$, как делали это ранее. Иными словами, чтобы определить модификатор *значительно больше*, надо удалить из рассмотрения все эталонные *маленькие* объекты. Аналогично, для модификатора *значительно меньше* отбрасываем все эталонные *большие* объекты. Для модификатора *слегка больше* будем отбрасывать не все объекты из U_2, а только те, что не больше его медианы; для *слегка меньше* - те, что не меньше медианы множества U_1.

Более подробное описание такого алгоритма и некоторые его свойства можно найти в [26], здесь же нас будет интересовать его поведение. Для этого обозначим модификатор *значительно больше* через R, модификатор *значительно меньше* через L, модификатор *слегка больше* через r, а *слегка меньше* через l. Центры кластеров, получившиеся после применения модификаторов, будем обозначать, приписывая к c_1 или c_2 слева обозначение соответсвующего модификатора (например, Rc_1 обозначает центр кластера *больших* объектов после применения модификатора *значительно больше*). Исследуем, что будет происходить при чередовании модификаторов R и L. Сначала применяем R. Тогда $Rc_1 > c_1$ и $Rc_2 > c_2$, так как мы отбросили все объекты, не большие c_2. Далее действует L. $LRc_1 < c_1$ и $LRc_2 < c_2$, ведь c_1 и c_2 были получены без отбрасывания объектов, а для построения LRc_1 и LRc_2 были отброшены объекты, большие, чем Rc_1. Теперь снова применяем R и получаем, что $c_1 < RLRc_1 < Rc_1$ и $c_2 < RLRc_2 < Rc_2$. Рассуждая точно так же дальше, можно увидеть, что центр кластера c_1 смещается следующим образом: $c_1 \to c_1 < Rc_1 \to LRc_1 < c_1 < Rc_1 \to LRc_1 < c_1 < RLRc_1 < Rc_1 \to LRLRc_1 < LRc_1 < c_1 < RLRc_1 < Rc_1 \to LRLRc_1 < LRc_1 < c_1 < RLRLRc_1 < RLRc_1 < Rc_1 \to ...$ Центр кластера c_2 смещается аналогично. Если же начать чередование модификаторов с L, то будет "симметричная" ситуация: $c_1 \to Lc_1 < c_1 \to Lc_1 < c_1 < RLc_1 \to Lc_1 < LRLc_1 < c_1 < RLc_1 \to Lc_1 < LRLc_1 < c_1 < RLc_1 < RLRLc_1 \to Lc_1 < LRLc_1 < LRLRLc_1 < c_1 < RLc_1 < RLRLc_1 \to ...$ Центр кластера c_2 смещается аналогично. Так как в базе конечное число объектов, то на определённом шаге мы получим, что в обоих кластерах единичную принадлежность имеют те же самые объекты, что и при c_1 и c_2 в качестве центров кластеров. Причём, если чередование модификаторов начинать с R, то эта ситуация возникнет после какого-то применения R, а если начинать с L - после какого-то применения L. Аналогичные рассуждения можно провести, заменив R на r, а L на l на каждом шаге. Однако если на каких-то шагах заменять R на r, а на каких-то нет, то смещения центров кластеров могут отличаться от приведённых выше. Например, рассмотрим такую последовательность модификаторов: rLR. Тогда $c_2 < rc_2$, далее, $Lrc_2 < c_2 < rc_2$, а положение центра кластера $RLrc_2$ уже невозможно определить, ибо мы не знаем, каких объектов больше: меньших, чем Lrc_2 , или меньших, чем медиана множества U_2 на первом шаге. Таким образом, выполняется следующая теорема.

Теорема 1. *При чередовании модификаторов R и L (или r и l) за конечное число шагов получим такие же эталоные объекты в обоих кластерах, как и до применения модификаторов.*

С одной стороны, хорошо, что, чередуя модификаторы, пользователь в конечном итоге получит то же, что и до их применения, но, с другой стороны, можно несколько раз применить R, а потом один раз L и получить центры кластеров левее, чем начальные c_1 и c_2. Другими словами, запрашивая несколько раз объекты *больше*, а потом один раз *меньше*, получаем объекты, меньшие, чем были изначально. Это недостаток алгоритма, поэтому его необходимо модифицировать.

4. Алгоритм персонализации на основе эмпирического распределения объектов в универсальном множестве

Пример с цепочкой модификаторов rLR показывает, что мы не можем предсказать поведение алгоритма, так как не знаем, сколько объектов на данной итерации содержат множества U_1 и U_2. А из ситуации, описанной после теоремы 1, можно извлечь вывод, что необходимо каким-то образом ограничивать сдвиги функций принадлежности на каждом запросе. Эти соображения наталкивают на идею отказаться от нечёткой кластеризации в пользу привязки нечётких множеств к эмпирическому распределению объектов и не использовать на данной итерации объекты, которые были отброшены на предыдущих шагах.

Для этого рассмотрим квантили эмпирического распределения объектов в универсальном множестве U. Напомним, как для эмпирических распределений определяется квантиль уровня α. Пусть $x_0, ..., x_{N-1}$ - все объекты из U. Составим вариационный ряд $V_0 < V_1 < ... < V_{N-1} = V_N$. Мы добавляем V_N, чтобы формулы ниже были верны и для квантиля уровня 1, а также считаем, что все объекты различны. Определим число $K = \lfloor \alpha \cdot (N-1) \rfloor$ (где $\lfloor \cdot \rfloor$ - округление вниз до ближайшего целого числа) и сравним $K+1$ с $\alpha \cdot N$.

1) если $K+1 < \alpha \cdot N$, то $x_\alpha = V_{K+1}$;
2) если $K+1 > \alpha \cdot N$, то $x_\alpha = V_K$;
3) $K+1 = \alpha \cdot N$, то $x_\alpha = (V_K + V_{K+1})/2$.

Теперь приступим к построению функций принадлежности. Идея, которая поможет нам это осуществить, заключается в том, чтобы работать

не с центрами кластеров, как ранее, а с "границами" множеств эталонных объектов. Выберем два квантиля эмпирического распределения объектов, $x_{\alpha_1} > x_{\alpha_2}$, причём, с точки зрения дисбаланса выделения из множества U эталонных *маленьких* и эталонных *больших* объектов будет разумным взять эти квантили такими, что $\alpha_1 + \alpha_2 = 1$. Для всех $x \geq x_{\alpha_1}$ положим значения функций принадлежности $\mu_1(x) = 1$, $\mu_2(x) = 0$, а для всех $x \leq x_{\alpha_2}$ $\mu_1(x) = 0$, а $\mu_2(x) = 1$. Числа x_{α_1} и x_{α_2} назовём границами множеств эталонов U_1 и U_2 соответственно. Для остальных x линейно достроим функции μ_1 и μ_2, то есть для $x \in (x_{\alpha_2}; x_{\alpha_1})$

$$\mu_1(x) = \frac{x - x_{\alpha_2}}{x_{\alpha_1} - x_{\alpha_2}},\ \mu_2(x) = \frac{x - x_{\alpha_1}}{x_{\alpha_2} - x_{\alpha_1}}.$$

Таким образом, выбор конкретных значений α_1 и α_2 приведёт нас к явно заданным функциям принадлежности. В частности, рассмотрим $\alpha_1 = 0.75$ и $\alpha_2 = 0.25$. Тогда верна следующая лемма.

Лемма 1. *При $\alpha_1 = 0.75$, $\alpha_2 = 0.25$ количество объектов в каждом из множеств U_1, U_2 равно $\lceil N/4 \rceil$, где $\lceil \cdot \rceil$ - округление вверх до ближайшего целого числа.*

Доказательство

Рассмотри сначала множество U_2 и определяющее его число $\alpha_2 = 0.25$. В этом случае, следуя определению квантиля эмпирического распределения выше, имеем $K = \lfloor (N-1)/4 \rfloor$, $K+1 = \lfloor (N+3)/4 \rfloor$. Сравним теперь $\lfloor (N+3)/4 \rfloor$ и $N/4$. Нетрудно получить, что $\lfloor (N+3)/4 \rfloor = N/4$ для $N \equiv 0(4)$, для остальных N имеет место неравенство $\lfloor (N+3)/4 \rfloor > N/4$. Рассмотрим случай с неравенством. Так как оно выполняется, то $x_{0.25} = V_{\lfloor (N-1)/4 \rfloor}$, а значит, множество U_2 будет содержать ровно те объекты, которые не больше, чем $V_{\lfloor (N-1)/4 \rfloor}$, то есть объекты $V_0, ..., V_{\lfloor (N-1)/4 \rfloor}$. При рассматриваемых N верно $\lfloor (N-1)/4 \rfloor = \lceil N/4 \rceil - 1$, поэтому первые $\lceil N/4 \rceil$ членов вариационного ряда и только они попадут в U_2. Теперь обратимся к случаю с $N \equiv 0(4)$, и пусть $N = 4m$. Так как $\lfloor (N+3)/4 \rfloor = N/4$, то

$$x_{0.25} = \frac{V_{\lfloor (N-1)/4 \rfloor} + V_{\lfloor (N+3)/4 \rfloor}}{2} = \frac{V_{m-1} + V_m}{2}.$$

А поскольку $V_{\lceil N/4 \rceil - 1} = V_{m-1}$, по определению, меньше, чем $x_{0.25}$, а $V_{\lceil N/4 \rceil} = V_{\lfloor (N+3)/4 \rfloor} = V_m > x_{0.25}$, то U_2 снова содержит ровно $\lceil N/4 \rceil$ наименьших объектов универсального множества U.

Перейдём к рассмотрению множества U_1 и квантиля $x_{\alpha_1} = x_{0.75}$. Здесь $K = \lfloor (3N-3)/4 \rfloor$, $K+1 = \lfloor (3N+1)/4 \rfloor$. Сравнивая $K+1$ и $3N/4$, получаем три возможных случая:

1) При $N = 4m$ имеем $\lfloor (3N+1)/4 \rfloor = 3N/4 = 3m$. Значит,

$$x_{0.75} = \frac{V_{\lfloor (3N-3)/4 \rfloor} + V_{\lfloor (3N+1)/4 \rfloor}}{2} = \frac{V_{3m-1} + V_{3m}}{2}.$$

Заметим, что $N - \lceil N/4 \rceil = 3m$, и так как $V_{3m-1} < x_{0.75} < V_{3m}$, то $\lceil N/4 \rceil$ последних объектов вариационного ряда $V_0 < ... < V_{N-1}$ и только они попадут в множество U_1.

2) Если $N = 4m + 1$, то $\lfloor (3N+1)/4 \rfloor > 3N/4$, поэтому $x_{0.75} = V_{\lfloor (3N-3)/4 \rfloor} = V_{3m}$. Также, $N - \lceil N/4 \rceil = 3m$, значит, в этом случае множество U_1 снова содержит $\lceil N/4 \rceil$ наибольших объектов из U и только их.

3) Наконец, $\lfloor (3N+1)/4 \rfloor < 3N/4$ при N, равных $4m+2$ и $4m+3$, тогда $x_{0.75} = V_{\lfloor (3N+1)/4 \rfloor}$. То есть, $V_{\lfloor (3N+1)/4 \rfloor} = V_{3m+1}$, а $N - \lceil N/4 \rceil = 3m+1$ для $N = 4m+2$, и в этом случае получаем требуемое утверждение, а для $N = 4m + 3$, аналогично, $V_{\lfloor (3N+1)/4 \rfloor} = V_{3m+2}$ и $N - \lceil N/4 \rceil = 3m + 2$, следовательно, и здесь $\lceil N/4 \rceil$ наибольших объектов множества U образуют множество U_1.

Лемма 1 доказана.

Теперь нам известны мощности множеств U_1 и U_2. Перейдём к описанию алгоритма изменения функций принадлежности μ_1 и μ_2 под действием тех же модификаторов, что и ранее: R, L, r и l, для краткости используем их обозначения вместо названий. Как и ранее, для опеределения модификатора R положим новые значения коэффициентов принадлежности $\mu_1(x) = 0$ и $\mu_2(x) = 1$ для всех x из множества U_2. Для оставшихся x строим функции принадлежности по квантилям эмпирического распределения объектов множества $U \backslash U_2$. Аналогичным образом определяется модификатор L: $\mu_1(x) = 1$, $\mu_2(x) = 0$ для всех x из множества U_1, для оставшихся объектов строим коэффициенты принадлежности по квантилям их эмпирического распределения. Чтобы определить модификатор r, будем полагать $\mu_1(x) = 0$, $\mu_2(x) = 1$ только для тех x из U_2, которые не больше его медианы, для остальных объектов, как и в предыдущих случаях, строим функции. Наконец, для определения модификатора l положим $\mu_1(x) = 1$, $\mu_2(x) = 0$ для всех x, не меньших медианы множества U_1, коэффициенты принадлежности для остальных объектов получим на основе их эмпирического распределения. Таким образом, для определения модификаторов мы всегда отбрасываем часть объектов, полагая их коэффициенты принадлежности равными 0 или 1, а для оставшихся проводим построение функций принадлежности.

Далее, будем считать, что объекты исключаются из рассмотрения безвозвратно, то есть, будучи отброшены на какой-либо итерации алгоритма, они не используются и на всех последующих итерациях. Под U будем понимать не всё универсальное множество, а только те его объекты, которые не были отброшены до данной итерации. Исследуем теперь поведение алгоритма при многократных запросах пользователя. Из построения функций принадлежности μ_1 и μ_2 следует, что мощность множества всех находящихся в рассмотрении объектов уменьшается хотя бы на единицу при переходе к следующей итерации до тех пор, пока текущие множества U_1 и U_2 содержат хотя бы по одному объекту. Из Леммы 1 получаем, что мощность каждого из множеств U_1 и U_2 равна 1, если на момент данной итерации остались не отброшенными как максимум 4 объекта. С другой стороны, вспоминая семантику наших модификаторов, будет разумным утверждать, что в ситуации, когда U_1 и U_2 содержат по одному объекту, алгоритм построения новых функций принадлежности не имеет смысла, так как если есть всего один эталонный *большой* и всего один эталонный *маленький* объект, то нельзя говорить об объектах, которые *больше* или *меньше*. Поэтому в качестве критерия остановки алгоритма (помимо удовлетворённости пользователя текущими коэффициентами принадлежности, то есть ситуацией, когда пользователь нашел удовлетворяющий его объект) определим порог мощности множества U, равный 4. Далее полагаем, что удовлетворённость результатами работы алгоритма не наступает никогда, чтобы исследовать его поведение в этом случае, который является наихудшим в смысле количества выполняемых итераций.

Обратимся к Лемме 1 и установим, как меняется мощность множества U при последовательных итерациях алгоритма. Рассмотрим сначала модификаторы R и L. После первой итерации множество U будет содержать $N - \lceil N/4 \rceil = \lfloor 3N/4 \rfloor$ объектов. Обозначим за N_i мощность U после i-й итерации и заметим, что она таким же образом выражается через N_{i-1}: $N_i = \lfloor 3N_{i-1}/4 \rfloor$, $N_0 = N$. Аналогичные рассуждения, правда, с чуть более громоздкими выкладками можно провести и для модификаторов l и r:

$$N_i = N_{i-1} - \lceil \frac{\lceil N_{i-1}/4 \rceil}{2} \rceil = \lfloor \frac{2N_{i-1} - \lceil N_{i-1}/4 \rceil}{2} \rfloor = \lfloor \frac{\lfloor 7N_{i-1}/4 \rfloor}{2} \rfloor,\ N_0 = N.$$

Далее, пусть через M итераций с модификаторами R и L мощность множества U стала не больше 4; это случится, ибо на каждой итерации она уменьшается хотя бы на 1. Также пусть через m итераций с модификаторами r и l мощность множества U стала не больше 4. Тогда заметим,

что при использовании всех четырёх модификаторов, мощность U будет не превышать 4 через число итераций $M' : m \leq M' \leq M$. Таким образом, выполняется следующая теорема.

Теорема 2. *Алгоритм персонализации функций принадлежности на основе эмпирического распределения объектов обладает свойствами:*

1) *При любых запросах пользователя он совершает конечное число итераций;*
2) *Если используются только модификаторы R и L и не наступает удовлетворённость результатами работы, то алгоритм остановится через M итераций, где M таково, что $N_M = \lfloor 3N_{M-1}/4 \rfloor \leq 4$, $N_0 = N$;*
3) *Если используются только модификаторы r и l и не наступает удовлетворённость результатами работы, то алгоритм остановится через m итераций, где m таково, что $N_m = \lfloor \frac{\lceil 7N_{m-1}/4 \rceil}{2} \rfloor \leq 4$, $N_0 = N$;*
4) *Если же используются все модификаторы R, L, r и l и не наступает удовлетворённость результатами работы, то алгоритм остановится через M' итераций, где M' таково, что $m \leq M' \leq M$.*

5. Заключение

В данной работе была исследована задача персонализации нечётких пользовательских понятий на примере построения и модификации функций принадлежности двух нечётких множеств. Как результат можно выделить описание алгоритма персонализации и оценку количества его итераций в худшем случае.

Для последующего изучения перспективным выглядит вопрос о возможности алгоритма обнаружить в универсальном множестве заранее заданный объект. Также представляется интересным исследование произвольного числа функций принадлежности в данном контексте.

Список литературы

[1] David Reinsel, John Gantz, John Rydning. The digitization of the world: from edge to core. November 2018 (https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf)

[2] В. Елистратов. Количество активных смартфонов и планшетов превысило население Земли. 7 октября 2014. https://tjournal.ru/52596-more-phones-than-people

[3] The Biggest Marketing Trend Of 2017: Personalization. January 31, 2017. https://fusiononemarketing.com/biggest-marketing-trend-2017-personalization/

[4] Dillon Baker. How Personalization Is Changing Content Marketing. March 31st, 2017, https://contently.com/2017/03/31/personalization-changing-content-marketing/

[5] Jim Rund. HOW PERSONALIZATION WILL EVOLVE IN 2018, January 29, 2018, http://adage.com/article/epsilon/personalization-evolve-2018/312048/

[6] Shep Hyken. Recommended Just For You: The Power Of Personalization. May 13, 2017, https://www.forbes.com/sites/shephyken/2017/05/13/recommended-just-for-you-the-power-of-personalization/#613481826087

[7] Discover the Power of Personalization, https://www.demandware.com/resources/power-of-personalized-shopping

[8] Heike Young. Welcome to 2018, The Year of Personalization. Here's 3 Things You Need to Know. DECEMBER 21, 2017, https://www.demandware.com/blog/retail-intelligence/welcome-2018-year-personalization-heres-3-things-need-know

[9] Nigel Wixcey. Made-to-order: The rise of mass personalisation. The Deloitte Consumer Review. https://www2.deloitte.com/tr/en/pages/consumer-business/articles/made-to-order-the-rise-of-mass-personalisation.html

[10] Andy Betts. A new era of personalization: The hyperconnected customer experience. January 23, 2018, https://martechtoday.com/new-era-personalization-hyper-connected-customer-experience-209529

[11] Shauna Robinson. Personalization: The Next Big E-Learning Trend, https://www.td.org/magazines/td-magazine/personalization-the-next-big-e-learning-trend

[12] Fashion in 2018. Getting Personal. BY BOF TEAM AND MCKINSEY & COMPANY, JANUARY 2, 2018, https://www.businessoffashion.com/articles/intelligence/top-industry-trends-2018-4-getting-personal

[13] SPECIAL REPORT: PERSONALIZATION TECH TRENDS, https://www.chiefmarketer.com/special-reports/special-report-personalization-tech-trends/

[14] Thijs Kuin . How to Start with Digital Marketing Personalization. 23 February, 2018, https://www.accenture-insights.nl/en-us/articles/digital-market-personalization

[15] Natalie Mouradian. The Dieline's 2018 Trend Report: Brands Become Hyper-Personalized. February 27, 2018, http://www.thedieline.com/blog/2018/2/26/brands-become-hyper-personalized

[16] Lyapin B. , Ryjov A. A Fuzzy Linguistic Interface for Data Bases in Nuclear Safety Problems. Fuzzy Logic and Intelligent Technologies in Nuclear Science. Proceedings of the 1st International FLINS Workshop, Mol, Belgium, September 14-16, 1994. Edited by Da Ruan, Pierre D'hondt, Paul Govaerts, Etienne E. Kerre, World Scientific. p. 212-215.

[17] Рыжов А. П., Журавлев А. Д., Вахов А. Н., Кривцов В. В. Об одном подходе к персонификации обучения в рамках компьютерных обучающих систем. Интеллектуальные Системы. Теория и приложения. Т. 20, Вып. 3, 2016, с. 180-185.

[18] A. Ryjov, A. Vakhov, V. Krivtsov, and A. Zhuravlev. Personalization and optimization of learning based on technology for evaluation and monitoring of complex processes: Uchi.ru case study. The 2016 International Conference on Computational Science & Computational Intelligence. Ed. by: Hamid R. Arabnia, Leonidas Deligiannidis, and Mary Yang. Las Vegas, Nevada, USA, 15-17 December 2016, pp. 378 - 381.

[19] Рыжов А. П., Кривцов В.В., Журавлев А.Д. Некоторые задачи кластеризации и ранжирования для персонификации компьютерных обучающих систем. Дистанционные образовательные технологии: материалы I Всероссийской научно-практической интернет-конференции (г. Ялта, 19-23 сентября 2016 года) – Симферополь, ИТ «АРИАЛ», 2016. – С. 37-41.

[20] Рыжов А.П., Новиков П.А. Об одной модели цифровых привычек. Интеллектуальные Системы Теория и приложения. Т. 21 Вып. 3, 2017, с. 91-102.

[21] Рыжов А.П. Некоторые задачи оптимизации и персонификации социальных сетей. Saarbrucken, LAP, 2015, 88 с.

[22] Ryjov A. Personalization of Social Networks: Adaptive Semantic Layer Approach. In: Social Networks: A Framework of Computational Intelligence. Witold Pedrycz and Shyi-Ming Chen (Eds.). Springer International Publishing Switzerland 2014, pp. 21-40.

[23] Ryjov A. Personalization and Optimization of Information Retrieval: Adaptive Semantic Layer Approach. In: Zadeh L., Yager R., Shahbazova S., Reformat M., Kreinovich V. (eds) Recent Developments and the New Direction in Soft-Computing Foundations and Applications. Studies in Fuzziness and Soft Computing, vol 361, Springer, Cham, 2018, pp. 15-24.

[24] Рыжов А.П. Модели поиска информации в нечеткой среде. Москва, Издательство Центра прикладных исследований при механико-математическом факультете МГУ, 2004, 96 с. http://www.intsys.msu.ru/staff/ryzhov/FuzzyRetrieval2010.htm

[25] Дмитриенко Г.С. О формализации и модификации понятий при нечетком поске. Дипломная работа. Москва, МГУ им. М.В. Ломоносова, механико-математический факультет, кафедра математической теории интеллектуальных систем, 2009.

[26] Ogorodnikov N. (2017) One Approach to the Description of Linguistic Uncertainties. In: Nguyen N., Papadopoulos G., Jędrzejowicz P., Trawiński B., Vossen G. (eds) Computational Collective Intelligence. ICCCI 2017. Lecture Notes in Computer Science, vol 10449. Springer, Cham. (ISBN:978-3-319-67077-5)

On one method of information retrieval personalization
Ryjov A.P., Ogorodnikov N.M.

In this paper we consider the problem of fuzzy user concepts personalization using the action of modifiers on certain fuzzy sets. The existing algorithm of personalization was investigated and modified; for its new version the number of iterations performed was estimated.

Keywords: fuzzy sets, personalization.

Новая математическая модель протоколов аутентификации и основанный на ней метод верификации

Миронов Андрей Михайлович

Протоколы аутентификации – это распределенные алгоритмы, предназначенные для обеспечения аутентификации агентов и передачи конфиденциальной информации (криптографических ключей, и т.п.) в небезопасной среде. Они используются, например, в электронных платежах, электронных процедурах голосования, системах доступа к базам данных, и т.д. Учитывая большой финансовый и социальный ущерб в случае неправильной работы таких протоколов, необходимо использовать математические методы для обоснования их корректности и безопасности. В настоящей работе вводится новая математическая модель таких протоколов аутентификации, позволяющая описывать как сами протоколы, так и их свойства. Показывается, как на базе данной модели можно решать задачи верификации протоколов аутентификации.

Ключевые слова: протоколы аутентификации, распределенные алгоритмы, верификация.

1. Введение

Протоколы аутентификации (ПА) – это распределенные алгоритмы, используемые для защиты электронных транзакций. ПА представляет собой описание порядка обмена сообщениями между несколькими агентами. Примеры таких агентов – компьютерные системы, банковские карточки, люди, и т.д. Для обеспечения свойств безопасности, (таких например как конфиденциальность передаваемых данных) в ПА используются криптографические преобразования (шифрование, электронная подпись, хэш-функции, и т.п.). Мы предполагаем, что криптографические преобразования, используемые в ПА, являются идеальными, т.е. удовлетворяют некоторым аксиомам, выражающим, например, невозможность

извлечения открытых текстов из шифртекстов без знания соответствующих криптографических ключей.

Наиболее известным ПА является протокол Kerberos [1]. Исходное описание данного протокола содержало уязвимости. Данные уязвимости были обнаружены формальными методами и исправлены, после чего новая версия данного протокола была формально верифицирована [2].

Есть много других ПА, используемых, например, для аутентификации перед провайдерами мобильной телефонной связи, для снятия денег в банкомате, и т.п. Цифровизация современного общества требует более широкого использования ПА для решения самых разных задач, таких например как работа с электронными паспортами (которые могут включать в себя чипы RFID), проведение электронных выборов (в которых предполагается голосование по интернету), и т. д.

Многие уязвимости ПА связаны не с плохими криптографическими качествами используемых в них криптографических примитивов, а с логическими ошибками в протоколах. Например, в ПА входа в портал Google, позволяющем пользователю идентифицировать себя только один раз, а затем обращаться к различным приложениям (таким, например, как Gmail или календарь Google), обнаружена логическая ошибка, позволяющая нечестному поставщику услуг выдавать себя за любого из своих пользователей для другого поставщика услуг.

Существует много других примеров ПА, которые продолжительное время использовались в критических по безопасности системах, однако затем обнаружилось что эти ПА содержат уязвимости следующего вида:

- агенты, участвующие в этих ПА, могут получать искаженные сообщения (или вообще терять их) в результате перехвата, удаления или искажения противником передаваемых сообщений, в результате чего нарушается свойство целостности,
- противник может узнать конфиденциальную информацию, содержащуюся в перехваченных сообщениях, в результате ошибочных или злонамеренных действий участников ПА.

Эти примеры являются обоснованием того что для ПА, используемых в критических по безопасности системах, недостаточно неформального анализа требуемых свойств, необходимо

- построить **математическую модель** анализируемого ПА,
- описать проверяемые свойства анализируемого ПА в виде математического объекта, называемого **спецификацией** этого ПА, и

- построить доказательство утверждения о том, что анализируемый ПА удовлетворяет (или не удовлетворяет) спецификации, процедура построения такого доказательства называется **верификацией** анализируемого ПА.

Исторически первым формальным аппаратом для математического анализа ПА была логика Бэрроуза-Абади-Нидхэма [3]. Данный аппарат имеет очень большие ограничения, в частности он не позволяет рассматривать случай неограниченного порождения сеансов анализируемого протокола.

Более популярным подходом к верификации ПА является формализм пространств нитей (strand spaces), развитый в работах сотрудников корпорации MITRE F. Javier Thayer Fabrega, Jonathan C. Herzog, Joshua D. Guttman [4]. Данный формализм также ограничен невозможностью верификации протоколов, порождающих неограниченное число сеансов.

Ссылки на работы, посвященные описанию различных формализмов, предназначенных для моделирования и верификации ПА, можно найти на странице курса по верификации криптопротоколов профессора Университета Эйндховена Jerry den Hartog [5]. См. также [6].

Одним из формализмов верификации ПА является подход, связанный с использованием клауз Хорна и систем уравнений с ограничениями (Constraint Systems), развитый в работах Абади, Блаше, Кортье и других специалистов (приведем лишь ссылку на современный вводный текст с изложением этого формализма [7], в нем содержатся многочисленные ссылки на недавние научные работы в этой области). И этот формализм имеет свои ограничения, главное из которых – нахождение ложных атак на анализируемый протокол. Говоря неформально, в данном подходе каждой паре (a_1, a_2) соседних действий участников протокола, первое из которых (a_1) - прием сообщения, а второе (a_2) - посылка сообщения, сопоставляется правило вывода, заключающееся в том, что если противник послал участнику какое-либо сообщение, совместимое с форматом сообщения в действии a_1, то он может получить от этого участника сообщение, соответствующее действию a_2, т.е. участники рассматриваются как автоматы с заданной реакцией. Фундаментальным недостатком такого подхода является отсутствие в модели противника хронологической связи между этими парами, о чем авторы [7] пишут на стр. 91 (см. пункт 8.2.4: Approximations), что приводит к тому что некоторые атаки на протокол, обнаруженные в этой модели, являются ложными. Судя по недавним публикациям авторов данного подхода, у них нет идей по поводу того, как можно было бы избавиться от данного недостатка.

В настоящем тексте вскрыта главная причина недостатка этой модели, она заключается в том что каждое правило вывода, соответствующее паре действий "ввод-вывод" какого-либо участника протокола может быть применено к состоянию s противника только с учетом "предыстории", хранящей последовательность действий, приводящих противника в состояние s. Настоящий текст посвящен построению математической модели ПА, реализующей описанный выше подход. Показывается, как на базе данной модели можно решать задачи описания свойств ПА. Одно из наиболее важных достоинств предложенной модели ПА заключается в высокой степени автоматизации решения задачи верификации ПА на основе данной модели.

2. Вспомогательные понятия

2.1. Термы

Мы будем предполагать, что заданы следующие множества:

- множество $\mathcal{T}$, элементы которого называются **типами**,
- множества $\mathcal{X}$ и $\mathcal{C}$, элементы которых называются **переменными** и **константами** соответственно, причем каждой переменной или константе e сопоставлен некоторый тип $t(e) \in \mathcal{T}$,
- множество $\mathcal{F}$, элементы которого называются **функциональными символами (ФС)**, причем каждому ФС $f \in \mathcal{F}$ сопоставлен тип $t(f)$, представляющий собой запись вида $(t_1, \ldots, t_n) \to t$, где $t_1, \ldots, t_n, t \in \mathcal{T}$.

Термы строятся из переменных, констант и ФС. Множество всех термов обозначается символом $\mathcal{E}$. Каждому терму e сопоставлен тип $t(e) \in \mathcal{T}$, определяемый структурой терма e. Правила построения термов имеют следующий вид:

- каждая переменная и константа является термом того типа, который сопоставлен этой переменной или константе, и
- если $e_1, \ldots, e_n \in \mathcal{E}$, $f \in \mathcal{F}$, и $t(f)$ имеет вид $(t(e_1), \ldots, t(e_n)) \to t$, то знакосочетание $f(e_1, \ldots, e_n)$ является термом типа t.

Будем считать, что $\mathcal{T}$ содержит следующие типы:

- `agent`, термы этого типа называются **именами агентов** (или просто **агентами**), они обозначают имена участников ПА,
- `key`, термы этого типа называются **ключами**, они обозначают криптографические ключи, которые участники ПА могут использовать для шифрования или дешифрования сообщений,
- `message`, термы этого типа называются **сообщениями**, они обозначают сообщения, которые участники ПА могут пересылать друг другу в процессе своей работы.
- `start`, `end`, $\texttt{initiator}_i$, $\texttt{initiator}_r$, $\texttt{responder}_i$, $\texttt{responder}_r$, $\texttt{value}_i$, $\texttt{value}_r$, данные типы носят служебный характер, существует единственная переменная каждого из этих типов, мы будем обозначать каждую из этих переменных той же записью, которой обозначается соответствующий ей тип.

Будем использовать перечисляемые ниже ФС.

- ФС h, h_i (где i – индекс) типа $(\texttt{message}) \to \texttt{message}$, данные ФС являются именами **хэш-функций (ХФ)**.
- ФС $tuple_n$ (где n – произвольное натуральное число) типа

$$(\underbrace{\texttt{message}, \ldots, \texttt{message}}_{n}) \to \texttt{message}.$$

  Терм вида $tuple_n(e_1, \ldots, e_n)$ будет обозначаться более короткой записью $(e_1, \ldots, e_n)$ и называться **списком** термов $e_1, \ldots, e_n$.
- ФС $pr_{n,i}$ (где n – произвольное натуральное число, и $i \in \{1, \ldots, n\}$) типа $(\texttt{message}) \to \texttt{message}$.

  Терм вида $pr_{n,i}(e)$ будет обозначаться записью $e_{(i)}$.

  Терм вида $(e_1, \ldots, e_n)_{(i)}$, где $1 \le i \le n$, будет называться **i–й компонентой списка** $(e_1, \ldots, e_n)$ и считаться равным терму e_i.
- ФС enc и dec типа $(\texttt{key}, \texttt{message}) \to \texttt{message}$.

  Терм вида $enc(k, e)$ (или $dec(k, e)$) обозначает сообщение, получаемое шифрованием (или дешифрованием, соответственно) сообщения e на ключе k.

  Терм вида $enc(k, e)$ будет обозначаться записью $k(e)$ и называться **шифрованным сообщением (ШС)**.

- ФС pk и sk типа $(\texttt{agent}) \to \texttt{key}$.

  Терм вида $pk(a)$ (или $sk(a)$) обозначает **открытый** (или **закрытый**, соответственно) ключ агента a.

  Будем предполагать, что

  - ключ вида $pk(a)$ может использоваться любым агентом для шифрования сообщений, и
  - ключ вида $sk(a)$ может использоваться только агентом a и предназначен для дешифрования сообщений, зашифрованных на ключе $pk(a)$.

  Терм вида $enc(pk(a), e)$ будет обозначаться записью $a(e)$.

  Термы вида $pk(a)$ и $sk(a)$ будут обозначаться записями вида a^+ и a^- соответственно.

- ФС key типа $(\texttt{key}) \to \texttt{message}$.

  Терм вида $key(k)$ обозначает сообщение, совпадающее с ключом k, термы такого вида используются для обозначения действий, связанных с передачей ключей.

  В записи терма вида $key(k)$ ФС key и скобки будут опускаться.

- ФС $sign$ типа $(\texttt{message}, \texttt{agent}) \to \texttt{message}$.

  Терм вида $sign(e, a)$ обозначает **цифровую подпись** сообщения e, сделанную агентом a.

  Тройка вида $(e, a, sign(e, a))$ будет обозначаться записью $\langle e \rangle_a$.

$\forall\, e \in \mathcal{E}$ запись X_e обозначает множество переменных, входящих в e.

$\forall\, X \subseteq \mathcal{X}$ запись $\mathcal{E}_X$ обозначает множество $\{e \in \mathcal{E} \mid X_e \subseteq X\}$.

Будем называть **подстановкой** произвольную частичную функцию $\theta : \mathcal{X} \to \mathcal{E}$, такую, что $\forall\, x \in \mathcal{X}$, если $\theta(x)$ определено, то $t(\theta(x)) = t(x)$. Ниже символ Θ обозначает множество всех подстановок, и $\forall\, \theta \in \Theta$

$$dom(\theta) \stackrel{\text{def}}{=} \{x \in \mathcal{X} \mid \theta(x) \text{ определено}\}.$$

$\forall\, X \subseteq \mathcal{X}$ запись Θ_X обозначает множество $\{\theta \in \Theta \mid dom(\theta) \subseteq X\}$.

$\forall\, \theta \in \Theta, \forall\, e \in \mathcal{E}$, запись θe обозначает терм определяемый рекурсивно: если $e = x \in dom(\theta)$, то $\theta e \stackrel{\text{def}}{=} \theta(x)$, если $e = x \in \mathcal{X} \setminus dom(\theta)$, то $\theta e \stackrel{\text{def}}{=} x$, если $e = c \in \mathcal{C}$, то $\theta e \stackrel{\text{def}}{=} c$, и если $e = f(e_1, \ldots, e_n)$, то $\theta e \stackrel{\text{def}}{=} f(\theta e_1, \ldots, \theta e_n)$.

$\forall\, \theta_1, \theta_2 \in \Theta$ записи $\theta_1 \subseteq \theta_2$ и $\theta_2 \supseteq \theta_1$ означают, что $dom(\theta_1) \subseteq dom(\theta_2)$, и $\forall\, x \in dom(\theta_1)\ \ \theta_1(x) = \theta_2(x)$.

2.2. Формулы

Понятие формулы определяется индуктивно следующим образом:

- если $e, e' \in \mathcal{E}$, то запись $[\![e = e']\!]$ является формулой,
- если $\beta_1, \ldots, \beta_n$ – формулы, то запись $\beta_1 \wedge \ldots \wedge \beta_n$ является формулой.

Произвольная формула обозначается символом β (возможно, с индексами). Множество всех формул обозначается символом $\mathcal{B}$.

$\forall\, e, e' \in \mathcal{E}, \forall\, \beta \in \mathcal{B}$ запись $[\![e = e']\!] \in \beta$ означает, что формула β содержит конъюнктивный член $[\![e = e']\!]$.

$\forall\, \beta \in \mathcal{B}$ запись X_β обозначает множество переменных, входящих в β.

$\forall\, X \subseteq \mathcal{X}$ запись $\mathcal{B}_X$ обозначает множество $\{\beta \in \mathcal{B} \mid X_\beta \subseteq X\}$.

$\forall\, \beta \in \mathcal{B}$ запись $\underset{\beta}{=}$ обозначает наименьшую конгруэнцию на $\mathcal{E}$ (относительно операций, соответствующих ФС из $\mathcal{F}$), удовлетворяющую условию: $\forall\, [\![e = e']\!] \in \beta, \forall\, \theta \in \Theta \quad (\theta e, \theta e') \in \underset{\beta}{=}$.

$\forall\, \beta_1, \beta_2 \in \mathcal{B}$ записи $\beta_1 \subseteq \beta_2$ и $\beta_2 \supseteq \beta_1$ означают, что

$$\forall\, [\![e = e']\!] \in \beta_1 \quad (e, e') \in \underset{\beta_2}{=} \tag{1}$$

(нетрудно видеть, что (1) равносильно включению $\underset{\beta_1}{=} \subseteq \underset{\beta_2}{=}$).

Формулы β_1 и β_2 считаются равными, если $\beta_1 \subseteq \beta_2$ и $\beta_2 \subseteq \beta_1$ (т.е. если конгруэнции $\underset{\beta_1}{=}$ и $\underset{\beta_2}{=}$ совпадают).

Нетрудно видеть, что следующие формулы равны:

- $[\![h(e) = h(e')]\!]$ и $[\![e = e']\!]$, где h – имя ХФ,
- $[\![k(e) = k'(e')]\!]$ и $[\![k = k']\!] \wedge [\![e = e']\!]$ где k, k' – ключи.

С каждой подстановкой $\theta \in \Theta$ связана формула, обозначаемая тем же символом θ, и определяемая как конъюнкция формул вида $[\![x = \theta(x)]\!]$, где $x \in dom(\theta)$.

2.3. Замкнутые множества термов

Пусть $E \subseteq \mathcal{E}$ и $\beta \in \mathcal{B}$. Множество E называется β–**замкнутым**, если

1) $\forall\, e \in \mathcal{E}$, если $e = f(e_1, \ldots, e_n)$, где $f \in \mathcal{F}$ и $e_1, \ldots, e_n \in E$, то $e \in E$,
2) $\forall\, e \in E \quad \{e\}_\beta \subseteq E$, где $\{e\}_\beta$ – класс конгруэнции $\underset{\beta}{=}$, содержащий e.

Замкнутые множества термов используются для представления множеств сообщений, которые м.б. известны противнику, и

1) первое из указанных выше условий соответствует операциям, которые противник I может выполнять с имеющимися у него сообщениями, например,

   - если I имеет сообщения e_1, ..., e_n, то он может скомпоновать из них сообщение $(e_1, \ldots, e_n)$,
   - если I имеет сообщение $(e_1, \ldots, e_n)$, то он может получить его компоненты e_1, ..., e_n,
   - если I имеет сообщения k и e, где k – ключ шифрования, то I может создать ШС $k(e)$,
   - если I имеет ШС e и ключ дешифрования k, то он может дешифровать e, т.е. получить $dec(k, e)$, и т.д.,

2) второе из указанных выше условий отражает тот факт, что разные термы могут соответствовать одинаковым сообщениям, и если формула β имеет вид $[\![e_1 = e'_1]\!] \wedge \ldots \wedge [\![e_n = e'_n]\!]$, то она выражает предположение об одинаковости сообщений, соответствующих e_i и e'_i $\forall\, i = 1, \ldots, n$, из которого следует, что включение $\beta \supseteq [\![e = e']\!]$ влечет одинаковость сообщений, представляемых термами e и e', т.е. если противник имеет сообщение, представляемое термом e, то он имеет сообщение, представляемое каждым элементом класса $\{e\}_\beta$.

Нетрудно доказать, что $\forall\, e \in \mathcal{E}$, $\forall\, \beta \in \mathcal{B}$ существует наименьшее (по включению) β–замкнутое множество $cl(e, \beta) \subseteq \mathcal{E}$, такое, что $e \in cl(e, \beta)$.

3. Протоколы аутентификации

3.1. Неформальное понятие протокола аутентификации

Будем рассматривать каждый ПА как распределенный алгоритм, в котором участвуют несколько последовательных процессов. Каждый из этих процессов связан с некоторым агентом. Допускается, что несколько из этих процессов м.б. связаны с одним и тем же агентом. Если в ПА P участвуют процессы $p_1, \ldots, p_n$, то будем обозначать данный факт записью

$$P = (p_1, \ldots, p_n). \tag{2}$$

Выполнение ПА (2) происходит путем выполнения каждого из входящих в него процессов $p_1, \ldots, p_n$. Выполнение каждого из этих процессов заключается в порождении им последовательности действий.

Работа каждого из процессов P_i в протоколе происходит путем последовательного выполнения **внешних действий** процесса P_i, такие действия заключаются в передаче или приеме сообщений, и **внутренних действий** процесса P_i.

4. Действия, процессы и выполнения

Для формального определения понятия протокола аутентификации введем понятия действия, процесса и выполнения процесса.

4.1. Действия

Действие – это запись одного из трех видов: ввод, вывод, внутреннее действие. Понятие действия связано с понятием **выполнения**. В начале выполнения какого-либо действия α определена подстановка θ, называемая **текущей подстановкой**, в результате выполнения этого действия подстановка θ заменяется на подстановку $\theta' \supseteq \theta$, которая будет текущей подстановкой в начале выполнения следующего действия.

Виды действий определяются следующим образом.

- **Ввод** имеет вид $[a?e]$, где a – агент (называемый **отправителем**), и $e \in \mathcal{E}$.

  Выполнение этого действия заключается в

  - получении какого-либо сообщения e' от агента a, и
  - замене текущей подстановки θ на новую текущую подстановку $\theta' \supseteq \theta$, такую, что $\theta' \supseteq [\![e = e']\!]$.

    Если не существует подстановки θ', удовлетворяющей этому условию, то выполнение данного действия невозможно.

- **Вывод** имеет вид $[a!e]$, где a – агент (называемый **получателем**), и $e \in \mathcal{E}$.

  Выполнение этого действия заключается в посылке сообщения e агенту a. Новая текущая подстановка совпадает с θ.

- **Внутреннее действие** имеет вид β, где $\beta \in \mathcal{B}$.

Выполнение этого действия заключается замене текущей подстановки θ на новую текущую подстановку $\theta' \supseteq \theta$, удовлетворяющую условию: $\theta' \supseteq \beta$.

Если не существует подстановки θ', удовлетворяющей этому условию, то выполнение данного действия невозможно.

Множество всех действий обозначается символом $\mathcal{A}$.

4.2. Понятие процесса

В этом параграфе определяется понятие **процесса**. Процессы описывают работу участников ПА.

Процесс – это совокупность $p = (a, S, s^0, R, \beta, e, U)$, где

- a – **агент**, с которым связан процесс p,
- S – множество **состояний**, $s^0 \in S$ – **начальное состояние**,
- $R \subseteq S \times \mathcal{A} \times S$ – множество **переходов**, каждый переход (s, α, s') из R обозначается записью $s \xrightarrow{\alpha} s'$,
- $\beta \in \mathcal{B}$ – **начальное условие**,
- $e \in \mathcal{E}$ – **раскрытый терм** (где под раскрытостью терма понимается предположение о том, что этот терм м.б. известен противнику),
- $U \subseteq \mathcal{X}$ – множество **уникальных переменных** процесса p. (смысл понятия уникальной переменной будет объяснен ниже)

Множество всех процессов обозначается символом $\mathcal{P}$.

$\forall\, p \in \mathcal{P}$ записи a_p, S_p, s^0_p, R_p, β_p, e_p, U_p обозначают соответствующие компоненты процесса p. Множество всех переменных, входящих в p, обозначается записью X_p. Множество всех отправителей и получателей, входящих в p, обозначается записью A_p. Процесс p, такой, что $R_p = \emptyset$ обозначается символом $\mathbf{0}$. Состояние $s \in S_p$ называется **терминальным**, если R_p не содержит переходов вида $s \xrightarrow{\alpha} s'$. Переход $s \xrightarrow{\alpha} s'$ называется **вводом**, **выводом**, или **внутренним переходом**, если α – ввод, вывод, или внутреннее действие, соответственно.

Процесс p можно представлять себе как граф (обозначаемый тем же символом p), вершинами которого являются состояния из S_p, а ребра соответствуют переходам из R_p: каждый переход $s \xrightarrow{\alpha} s'$ соответствует ребру из s в s' с меткой α.

4.3. Понятие выполнения процесса

Выполнение процесса $p \in \mathcal{P}$ можно понимать как обход графа p, начиная с состояния s_p^0, с выполнением действий, которые являются метками проходимых ребер.

Выполнение процесса $p \in \mathcal{P}$ – это граф V вида

$$v_0 \stackrel{\alpha_1}{\to} v_1 \stackrel{\alpha_2}{\to} \ldots \stackrel{\alpha_n}{\to} v_n \quad (n \geq 0) \tag{3}$$

каждое ребро которого помечено некоторым действием $\alpha_i \in \mathcal{A}$, и каждая вершина v которого имеет метку $(s_v, \theta_v, e_v) \in S_p \times \Theta \times \mathcal{E}$, причем $s_{v_0} = s_p^0$, $\theta_{v_0} \supseteq \beta_p$, $e_{v_0} = e_p$, $\forall\, i = 1, \ldots, n \;\; (s_{v_{i-1}} \stackrel{\alpha_i}{\to} s_{v_i}) \in R_p$, $\theta_{v_{i-1}} \subseteq \theta_{v_i}$, и

- если $\alpha_i = [a?e]$ или $[a!e]$, то $e_{v_i} = (e_{v_{i-1}}, e)$,
- если $\alpha = \beta \in \mathcal{B}$, то $\theta_{v_i} \supseteq \beta$, $e_{v_i} = e_{v_{i-1}}$.

$\forall\, i = 0, \ldots, n \;\; e_{v_i}$ можно понимать как раскрытый терм в вершине v_i.

Число n в выполнении (3) называется **длиной** этого выполнения, и обозначается записью $|V|$. Вершина v_n в (3) называется **последней**.

Если какая-либо вершина v выполнения (3) является концом ребра с меткой вида $[a?e]$, или началом ребра с меткой вида $[a!e]$, то

- вершина v называется **получателем** (от агента a) или **отправителем** (агенту a), соответственно, и
- терм e обозначается записью e_v, и называется **сообщением**, **полученным** (или **отправленным**, соответственно) в вершине v.

5. Протоколы аутентификации

5.1. Определение протокола аутентификации

Протокол аутентификации (называемый ниже просто **протоколом**) – это конечная совокупность P процессов $p_1, \ldots, p_n$, удовлетворяющая следующим условиям:

- $\forall\, i = 1, \ldots, n \;\; A_{p_i} \subseteq \{a_{p_1}, \ldots, a_{p_n}\}$,
- множества $X_{p_i} \setminus A_{p_i}$ $(i = 1, \ldots, n)$ дизъюнктны.

Если протокол состоит из процессов p_1, ..., p_n, то такой протокол обозначается записью $(p_1, \ldots, p_n)$.

5.2. Нормальное выполнение протокола

Нормальное выполнение протокола $P = (p_1, \ldots, p_n)$ заключается в совместном выполнении процессов $p_1, \ldots, p_n$, входящих в этот протокол. При нормальном выполнении протокола P выполнение каждого невнутреннего действия α каким-либо процессом p_i, входящим в P, происходит синхронно с выполнением некоторого действия α' (называемого **комплементарным к** α) другим процессом p_j, входящим в P, причем

- либо α имеет вид $[a_{p_j}?e]$, α' имеет вид $[a_{p_i}!e']$,
- либо α имеет вид $[a_{p_j}!e]$, α' имеет вид $[a_{p_i}?e']$,

т.е. пара (α, α') представляет собой пересылку либо сообщения e' от процесса p_j процессу p_i, либо сообщения e от процесса p_i процессу p_j.

Нормальное выполнение протокола $P = (p_1, \ldots, p_n)$ представляет собой ациклический граф V, получаемый из дизъюнктного объединения $V_1 \sqcup \ldots \sqcup V_n$ (где $\forall\, i = 1, \ldots, n$ V_i является выполнением процесса p_i) добавлением новых ребер (называемых **горизонтальными** ребрами, эти ребра соответствуют комплементарным действиям), причем каждое горизонтальное ребро имеет вид $v_i \to v_j$, где

- v_i и v_j – вершины графов V_i и V_j соответственно, причем $i \neq j$, v_i – отправитель агенту a_{p_j}, v_j – получатель от агента a_{p_i}, и
$$\theta_{v_i} \subseteq \theta_{v_j}, \quad [\![e_{v_i} = e_{v_j}]\!] \subseteq \theta_{v_j},$$
- для каждой вершины-отправителя (или вершины-получателя) v графа V существует единственное горизонтальное ребро графа V с началом (или концом, соответственно) в v.

5.3. Выполнение протокола с участием противника

Противник – это процесс (обозначаемый символом I), который может

- перехватывать и уничтожать сообщения от любого процесса, и
- создавать новые сообщения (используя перехваченные сообщения), и посылать их (от имени любого агента) любому процессу.

Из сообщений, которые I либо имел до начала своего выполнения, либо перехватил во время своего выполнения, он может строить новые сообщения, путем применения операций из $\mathcal{F}$.

Выполнение протокола с участием противника I

- представляет собой совокупность выполнений процессов, входящих в этот протокол, и
- происходит так, что выполнение каждого невнутреннего действия какого-либо процесса p, входящего в этот протокол, может выполняться
  - либо так же, как при нормальном выполнении этого протокола, т.е. это действие выполняется синхронно с комплементарным действием другого процесса, входящего в этот протокол,
  - либо синхронно с выполнением соответствующего действия I:
    - при выполнении процессом p действия вида $[a!e]$ посланное сообщение e может получить не агент a, а I, и
    - при выполнении процессом p действия вида $[a?e]$ процесс p может получить сообщение не от агента a, а от I.

Без ограничения общности можно считать, что если протокол выполняется с участием I, то все невнутренние действия, выполняемые процессами этого протокола, заключаются в их взаимодействии с I, т.к. если какое-либо из этих действий заключается в передаче сообщения e от процесса p_i процессу p_j этого протокола, то можно считать что данное действие является комбинацией двух действий:

- перехват противником I того сообщения e, которое послал p_i, и
- посылка этого сообщения e от I процессу p_j.

5.4. Система переходов протокола

Система переходов (СП) протокола P – это граф Σ_P с определяемыми ниже множеством вершин S_P (называемых **состояниями**) и множеством ребер R_P (называемых **переходами**). В Σ_P представлены все возможные совместные выполнения процессов, входящих в P, с участием противника.

Для определения множеств S_P и R_P введем обозначения:

- $\forall\, \theta \in \Theta$, $\forall\, p \in \mathcal{P}$ запись θp обозначает процесс (называемый **вариантом**) процесса p, получаемый из p заменой каждой переменной $x \in X_p$ на терм θx,
- запись V_P обозначает множество пар вида (p, V), где p – вариант процесса, входящего в P, и V – какое-либо выполнение p.

Каждое состояние s из S_P – это тройка (W_s, θ_s, e_s), где

- W_s – конечное подмножество множества V_P, причем если W_s имеет вид $\{(p_i, V_i) \mid i = 1, \ldots, n\}$, то множества $X_{p_1} \setminus A_{p_1}$, ..., $X_{p_n} \setminus A_{p_n}$ дизъюнктны,
- $\theta_s \in \Theta$ и $e_s \in \mathcal{E}$.

Компоненты состояния s понимаются следующим образом:

- W_s представляет собой одно из возможных совместных выполнений (до текущего момента времени) некоторых вариантов процессов, входящих в P, с участием противника,
- θ_s и e_s – текущая подстановка и раскрытый терм в состоянии s.

Множество R_P переходов СП Σ_P состоит из всех пар (s, s') состояний, таких, что $\theta_s \subseteq \theta_{s'}$, и верно одно из двух следующих утверждений:

- $W_{s'}$ получается из W_s путем активизации некоторого варианта p одного из процессов, входящих в P, т.е. $W_{s'} = W_{s'} \cup \{(p, V)\}$, где $|V| = 0$, $\theta_{s'} \supseteq \theta$, $e_{s'} = (e_s, e)$, где θ и e – вторая и третья компонента соответственно метки единственной вершины графа V, или
- $W_{s'}$ получается из W_s путем продвижения на один шаг одного из выполнений (p, V), входящих в W_s, т.е. W_s можно представить в виде $W \cup \{(p, V)\}$, и
  - $W_{s'} = W \cup \{(p, V')\}$, где V' получается из V добавлением ребра $v \xrightarrow{\alpha} v'$, выходящего из последней вершины v выполнения V,
  - * если $\alpha = [a?e]$, то $\theta_{s'} e \in cl(e_s, \theta_s)$, $e_{s'} = e_s$,
    * если $\alpha = [a!e]$, то $e_{s'} = (e_s, e)$,
    * если $\alpha = \beta \in \mathcal{B}$, то $\theta_{s'} \supseteq \beta, e_{s'} = e_s$.

Сформулированные выше условия на переход (s, s') представляют собой возможные варианты продолжения совместного выполнения вариантов процессов, входящих в P:

- либо активизируется новый вариант какого-либо процесса из P,
- либо продолжается выполнение одного из процессов, входящих в s.

Состояние $s \in S_P$ называется **начальным**, если $W_s = \emptyset$.

Путь в СП $\Sigma_P = (S_P, R_P)$ – это последовательность $\pi = (s_0, \ldots, s_n)$ состояний из S_P, такая, что $\forall\, i = 1, \ldots, n \;\; (s_{i-1}, s_i) \in R_P$. Состояние s_0 в пути $\pi = (s_0, \ldots, s_n)$ обозначается записью $\hat{\pi}$.

Множество всех путей в Σ_P обозначается записью Π_P, а множество всех путей π в Σ_P, таких, что $\hat{\pi}$ – начальное состояние – записью Π_P^0.

$\forall\, \pi = (s_0, \ldots, s_n) \in \Pi_P$, $\forall\, s, s' \in S_P$

- запись $s \in \pi$ означает, что $\exists\, i \in \{0, \ldots, n\} : s = s_i$,
- запись $s <_\pi s'$ означает, что $\exists\, i, j \in \{0, \ldots, n\}$: $i < j$, $s = s_i$, $s' = s_j$.

6. Спецификация свойств протоколов

6.1. Истинность формул в состояниях

$\forall\, \pi \in \Pi_P$, $\forall\, s \in \pi$, $\forall\, \beta \in \mathcal{B}$ запись $s \models_\pi \beta$ обозначает утверждение

$$\beta \subseteq \theta_s, \quad \forall\, s' <_\pi s \quad \beta \not\subseteq \theta_{s'}. \tag{4}$$

(4) интерпретируется как утверждение о том, что s – первое состояние на пути π, в котором истинна формула β.

6.2. Свойство соответствия

Свойство соответствия – это запись вида $\beta \rightsquigarrow \beta_1, \beta_2$, где $\beta, \beta_1, \beta_2 \in \mathcal{B}$.

Будем говорить, что протокол P обладает свойством соответствия $\beta \quad \beta_1, \beta_2$, если $\forall\, \pi \in \Pi_P$, $\forall\, s \in \pi$ верна импликация

$$s \models_\pi \beta \quad \Rightarrow \quad \exists\, s' <_\pi s : \;\; s' \models_\pi \beta_1,\; s \models_\pi \beta_2. \tag{5}$$

Если в (5) $\exists\, s'$ заменить на $\exists !\, s'$ (существует единственное s'), то соответствующее свойство протокола P называется **инъективным свойством соответствия**. Такое свойство обозначается записью $\beta \rightsquigarrow^! \beta_1, \beta_2$.

6.3. Корректность протоколов аутентификации как свойство соответствия

В этом параграфе показывается, как задача обоснования корректности ПА сводится к задаче проверки некоторого инъективного свойства соответствия.

Простейший ПА имеет вид (p_a, p_b). В нем участвуют агенты a и b, где

- агент a называется **доказывающим агентом**, он передает агенту b свое имя, и его цель в этом ПА – доказать агенту b, что переданное им имя является его подлинным именем,

- агент b называется **проверяющим агентом**, его цель в этом ПА – установить, совпадает ли полученное им имя с именем того агента, который взаимодействует с ним при выполнении этого ПА.

Будем предполагать, что

- множество X_{p_b} содержит переменную n_a, предназначенную для занесения в нее того значения, которое проверяющий агент должен получить от доказывающего агента в качестве его имени, и

- множество U_{p_a} содержит переменную r_a, значение которой (в составе зашифрованного сообщения) передается процессу p_b и записывается им в переменную $x_a \in X_{p_b}$.

В выполнении этого ПА может принимать участие противник, поэтому не исключено, что в действительности с агентом b взаимодействует совсем не тот агент, имя которого заносится в переменную n_a (т.е. который выдает себя за n_a, но в действительности не является им).

Будем говорить, что описанный выше ПА является **корректным**, если при любом его выполнении с участием противника, всякий раз, когда процесс p_b успешно завершает свою работу и принимает положительное решение (т.е. решает, что значение, записанное в n_a, совпадает с истинным именем агента, с которым взаимодействовал p_b), данное решение является правильным.

Формальное описание корректности ПА (p_a, p_b) выражается в виде излагаемого ниже свойства соответствия.

Обозначим записями $\tilde{p}_a$ и $\tilde{p}_b$ процессы, получаемые из p_a и p_b соответственно следующими модификациями:

- $\tilde{p}_a$ получается из p_a добавлением перехода вида $s \xrightarrow{\alpha} s^0_{p_a}$, где

  - s – новое состояние, которое будет начальным в $\tilde{p}_a$, и
  - $\alpha = [\![(start, initiator_i, responder_i, value_i) = (1, a_p, b, r_a)]\!]$, где r_a – вышеупомянутая переменная из U_{p_a},

- $\tilde{p}_b$ получается из p_b добавлением для каждого терминального состояния $s \in S_{p_b}$ перехода вида $s \xrightarrow{\alpha} s'$, где

– s' – новое состояние, которое будет терминальным в $\tilde{p}_b$, и

– $\alpha = [\![(end, initiator_r, responder_r, value_r) = (1, n_a, b, x_a)]\!]$, где n_a и x_a – вышеупомянутые переменные из X_{p_b}.

Свойство корректности ПА (p_a, p_b) можно сформулировать в виде инъективного свойства соответствия $\beta \rightsquigarrow \beta_1, \beta_2$ где

- $\beta = [\![end = 1]\!]$
- $\beta_1 = [\![start = 1]\!]$
- $\beta_2 = [\![initiator_i = initiator_r]\!] \wedge [\![responder_i = responder_r]\!] \wedge [\![value_i = value_r]\!]$

Поясним смысл изложенного выше описания свойства корректности: оно выражает утверждение о том, что если агент-респондер (b) завершил сеанс протокола (т.е. значение переменной `end` стало равно 1), и при этом

- он считает, что агентом-инициатором данного сеанса был агент, имя которого равно значению переменной n_a, и
- уникальное секретное значение, которое передал агент-инициатор, содержится в переменной x_a,

то в некоторый предшествующий момент времени агент, имя которого равно значению переменной n_a действительно начал сеанс выполнения этого протокола именно с b, и то уникальное секретное значение, которое он передал b, действительно равно значению переменной x_a.

7. Применение изложенного подхода

Для применения данного подхода к анализу какого-либо конкретного протокола необходимо описать конечное множество $\mathbf{P}$ протоколов, в выполнении которых агенты могут принимать участие, и при построении СП Σ_P в качестве P рассматривать не только совокупность процессов, входящих в анализируемый протокол, а все процессы, входящие в какой-либо протокол из $\mathbf{P}$, это необходимо для того чтобы установить существование атак, связанных с одновременным участием противника в нескольких сеансах различных протоколов. При атаке Лоу на протокол Нидхэма-Шредера (NSPK) противник участвует одновременно в двух сеансах NSPK, однако возможен случай одновременного участия

противника в различных сеансах разных протоколов, и описанная выше модель позволяет установить соответствующую атаку, если она действительно существует.

8. Заключение

За пределами настоящего исследования остались следующие вопросы.

1) Предложенный метод представляет собой по сути переборный алгоритм, связанный с построением всевозможных вариантов исполнения анализируемого протокола с участием противника (предполагая, что противник может быть одним из законных участников какого-либо протокола). Данный метод обладает свойством полноты: если атака действительно существует, то он её найдет, причем для поиска атаки достаточно построить конечный фрагмент СП Σ_P. Можно ли установить верхнюю границу на размер указанного выше фрагмента? Можно ли оптимизировать процесс поиска возможной атаки, т.е. строить описанный выше фрагмент аналогично тому как строится фрагмент модели Крипке при верификации свойств, темпоральными формулами, на основе метода on-the-fly, с использованием автоматов Бюхи?

2) Если анализируемый протокол не допускает атаки, то можно ли это доказать, не производя явное построение СП Σ_P, а используя например метод инвариантов (называемый также методом Флойда, или методом Хоара), который позволяет доказывать корректность программ без построения соответствующего множества их состояний?

Решению данных проблем будут посвящены дальнейшие исследования.

Список литературы

[1] Kerberos: The Network Authentication Protocol.
MIT Kerberos. 10 September 2015. Retrieved 31 October 2015
`http://web.mit.edu/kerberos/`

[2] Cervesato I., Jaggard A.D., Scedrov A., Tsay J.-K., Walstad C.,
Breaking and fixing public-key Kerberos,
Information and Computation Volume 206, Issues 2-4, (2008), 402-424.

[3] Burrows M., Abadi M., Needham R.,
A Logic of Authentication,
ACM Transactions on Computer Systems, 8(1), (1990) 18-36.

[4] Javier Thayer, Jonathan Herzog, and Joshua Guttman,
Презентация “Strand Spaces”,
http://www2.imm.dtu.dk/courses/02913/F05/

[5] Jerry den Hartog,
Страница курса Verification of Security protocols
https://www.win.tue.nl/~jhartog/CourseVerif/

[6] Proceedings of Joint Workshop on Foundations of Computer Security and Automated Reasoning for Security Protocol Analysis (FCS-ARSPA '06)
Information and Computation Volume 206, Issue 2, (2008).
https://www.sciencedirect.com/journal/information-and-computation/vol/206/issue/2

[7] Veronique Cortier, Steve Kremer
Formal Models and Techniques for Analyzing Security Protocols
Now Publishers Inc., Hanover, United States (2014).

New mathematical model of authentication protocols and verification method based on this model

Mironov Andrew M.

Authentication protocols are distributed algorithms designed to provide authentication of agents and the transfer of confidential information (cryptographic keys, etc.) in an insecure environment. They are used, for example, in electronic payments, electronic voting procedures, database access systems, etc. On the reason of the large financial and social damage in the case of the incorrect execution of such protocols, it is necessary to use mathematical methods to justify their correctness and security. In the present work, a new mathematical model of such authentication protocols is introduced, which provides a possibility to describe both the protocols and their properties. It is shown a possibility to solve problems of verification of authentication protocols.

Keywords: authentication protocols, distributed algorithms, verification

Порождение семейства ортогональных многочленов дискретной переменной для заданного множества узлов

Парфенов Д.В.

Предложен и обоснован вычислительно эффективный метод синтеза семейства дискретных многочленов комплексного аргумента с единичной весовой функцией, удовлетворяющих условию ортогональности на заданном произвольном конечном множестве несовпадающих узлов. Приведён детальный алгоритм расчёта коэффициентов и табуляции значений многочленов на языке GNU Octave/Mathworks Matlab. Дана оценка сложности, показано, что она минимально возможная.

Ключевые слова: цифровая обработка сигналов, устойчивая интерполяция, ортогональные многочлены дискретной комплексной переменной, произвольные узлы, расчёт коэффициентов, оптимальный алгоритм, вычислительная сложность.

В теории и практике современных вычислений заметная роль отводится дискретным ортогональным многочленам. В частности, в задачах устойчивой аппроксимации, построения квадратур, решения дифференциальных уравнений находят применение известные семейства дискретных ортогональных многочленов: Хана, Мейкснера, Кравчука, Шарлье и другие [1]. Многочисленные полезные свойства обеспечивают широкое применение ортогональных многочленов дискретной переменной и в современных алгоритмах цифровой обработки сигналов, например, в цифровой фильтрации, синтезе линейных устройств, при обработке радиосигналов, звука и изображения.

Нередко возникают ситуации, когда конечный набор узлов, на которых следует обеспечить ортогональность, продиктован задачей, нестандартен и фиксирован, а синтезировать соответствующую весовую функцию явным образом нецелесообразно из соображений минимизации вычислительной сложности. В данной работе предлагается простой алгоритм отыскания коэффициентов семейства многочленов степени не вы-

ше D, ортогональных на заданном множестве из $D+1$ различных узлов с единичной весовой функцией, и приводится его вычислительная сложность. Выкладки и программная реализация соответствуют случаю одномерных многочленов комплексной переменной и допускают как упрощение для случая действительной переменной, так и обобщения на многомерный случай и для заданных произвольных весовых функций, выходящие, однако, за рамки данной статьи.

Широко известно (см., например, [1]), что для всех семейств ортогональных многочленов справедлива так называемая трёхчленная рекуррентная формула, связывающая многочлены трёх смежных степеней. В литературе её обычно приводят в виде с тремя коэффициентами, хотя для монарных (т.е. имеющих единичный коэффициент при старшей степени) многочленов существует более простая форма:

$$p_{d+1}(x) = (x - \alpha_d)\, p_d(x) - \beta_d\, p_{d-1}(x). \tag{1}$$

Здесь $p_d(x)$ – ортогональный многочлен степени d переменной x, а α_d и β_d – некоторые комплекснозначные коэффициенты. В дальнейшем переменная x для краткости опущена в обозначениях многочленов. К сожалению, даже в специальной литературе не часто встречаются явные выражения для α_d и β_d, хотя они давно получены (см., например, [2]). Обоснованием приводимого ниже алгоритма служит следующая простая теорема.

Теорема 1. *Для произвольного семейства монарных ортогональных многочленов справедливо соотношение (1), причём:*

$$\alpha_d = \langle p_d, x\, p_d\rangle \,/\, \langle p_d, p_d\rangle\,, \quad \beta_d = \langle p_d, p_d\rangle \,/\, \langle p_{d-1}, p_{d-1}\rangle\,. \tag{2}$$

Доказательство. Введём вспомогательный многочлен $q_d := p_{d+1} - (x - \alpha_d)p_d + \beta_d p_{d-1}$. Действительно, с учётом монарности p_d и p_{d+1}, это многочлен степени d. Вследствие взаимной ортогональности p_{d-1}, p_d и p_{d+1} по условию теоремы, $\langle q_d, p_m\rangle = -\langle p_d, x\, p_m\rangle$, $m = 0, \ldots, d-2$. Но p_d ортогонален всем многочленам степени не выше $d-1$, откуда $\langle q_d, p_m\rangle = 0$. Подстановка (2) в $\langle q_d, p_{d-1}\rangle$ и $\langle q_d, p_d\rangle$ даёт два тождественных нуля. Таким образом, $\langle q_d, p_m\rangle = 0, \quad m = 0, \ldots, d$, но это означает $q_d \equiv 0$ и справедливость (1) при условии (2). □

Теорема 1 лежит в основе организации вычислений предлагаемого алгоритма *одновременного* нахождения коэффициентов и значений ортогональных с единичным весом многочленов на заданном множестве узлов $\{x_m\}_{m=1}^{D+1}$. Первые два монарные многочлена: $p_0 \equiv 1$ и

$p_1 = a_0 + x$, где из требования их взаимной ортогональности находим $a_0 = -(D{+}1)^{-1} \sum_{m=1}^{D+1} x_m$. Остальные многочлены вычисляются последовательно с помощью (2) и (1). Оптимизация учитывает четырёхкратное использование скалярных произведений вида $\langle p_d, p_d \rangle$: при вычислении числителя β_d, знаменателей α_d и β_d (от прошлой итерации) и при нормировке. Нормировка не является обязательной, её введение преследует две цели – уменьшение диапазона значений и повышение точности вычислений с плавающей точкой, что подтверждается экспериментально.

Из соображений компактности и ясности описания алгоритма на Листинге 1 приведён комментированный текст программы на языке систем GNU Octave/Mathworks Matlab[1]. Обозначения в листинге максимально соответствуют использованным в тексте: pdpd := $\langle p_d, p_d \rangle$, pdxpd := $\langle p_d, x\, p_d \rangle$, pd1pd1_inv := $\langle p_{d-1}, p_{d-1} \rangle^{-1}$, pdpd_inv := $\langle p_d, p_d \rangle^{-1}$, alpha_d := α_d, beta_d := β_d. Дополнительно обозначим $\mathbf{x}$ – вектор $D{+}1$ фиксированных различных узлов, на которых следует обеспечить ортогональность. Выходные параметры:

- a – матрица коэффициентов ортонормированных многочленов, где a(i, j) – значение коэффициента при степени $j - 1$ в многочлене степени $i - 1$, $i, j = 1, \ldots, D + 1$,
- s – матрица значений ортонормированных многочленов в узлах $\mathbf{x}$, где s(i, j) – значение многочлена степени $i - 1$ в j-ом узле $i, j = 1, \ldots, D + 1$.

```
a=zeros(D+1,D+1); % треугольная матрица коэффициентов
a(1,1)=1; % единственный ненулевой коэффициент p_0
a0=-sum(x)./(D+1); a(2,1:2)=[a0,1]; % ненулевые коэффициенты p_1
% s - матрица значений многочленов в фиксированных узлах x
s=[ones(1,D+1); a0+x; zeros(D-1,D+1)];

pdpd_inv=1./(D+1);
for d=2:D, % d - степень вычисляемого многочлена
  % находим 2 коэффициента в (1)
  pd1pd1_inv=pdpd_inv; % от предыдущей итерации / входа в цикл
  pdpd=sum(s(d,:).*s(d,:));
  pdpd_inv=1./pdpd;
  pdxpd=sum(s(d,:).*s(d,:).*x);
```

[1]На подобных C++ языках его объём вдвое больше и значительно хуже наглядность.

```
  alpha_d=pdxpd.*pdpd_inv;
  beta_d=pdpd.*pd1pd1_inv;

  % значения вычисляемого многочлена в узлах
  s(d+1,:)=(x-alpha_d).*s(d,:)-beta_d.*s(d-1,:);
  % коэффициенты вычисляемого многочлена
  a(d+1,1)=-alpha_d.*a(d,1)-beta_d.*a(d-1,1); % a_0
  if d>=3, % a_1,...,a_{d-2}
    a(d+1,2:d-1)=
      a(d,1:d-2)-alpha_d.*a(d,2:d-1)-beta_d.*a(d-1,2:d-1);
  end;
  a(d+1,d)=a(d,d-1)-alpha_d.*a(d,d); % a_{d-1}
  a(d+1,d+1)=1.0; % a_d

  % нормировка s(d-1,:) и a(d-1,:)
  % s(d-1,:) и a(d-1,:) не будут использованы для вычисления
  % следующих многочленов; pd1pd1_inv перезапишется в начале
  % следующей итерации
  pd1pd1_inv=sqrt(pd1pd1_inv); % нормирующий множитель
  s(d-1,:)=s(d-1,:).*pd1pd1_inv; % нормируем значения многочлена
  a(d-1,1:d-1)=a(d-1,1:d-1).*pd1pd1_inv; % нормируем коэффициенты
end;
% последний нормированный в цикле многочлен (степени D-2)
% был (D-1)-ым в массивах a и s. Нормируем оставшиеся многочлены
% степени (D-1) и степени D
pdpd_inv=sqrt(pdpd_inv);
s(D,:)=s(D,:).*pdpd_inv; % нормируем значения многочлена
a(D,1:D)=a(D,1:D).*pdpd_inv; % нормируем ненулевые коэффициенты
pdpd_inv=sqrt(1./sum(s(D+1,:).*s(D+1,:)));
s(D+1,:)=s(D+1,:).*pdpd_inv; % нормируем значения многочлена
a(D+1,:)=a(D+1,:).*pdpd_inv; % нормируем коэффициенты
```

Листинг 1.

Фрагмент кода на Листинге 2 иллюстрирует возможный способ разложения вектора $\mathbf{y}$ отсчётов некоторой величины в узлах $\mathbf{x}$ по синтезированным данным способом многочленам.

```
g=zeros(D+1,1); % коэффициенты разложения последовательности
% (y) по ортонормированным многочленам
```

```
for d=1:D+1, g(d)=sum(s(d,:).*y); end;
```

Листинг 2.

Вычисления последних двух строк могут быть выполнены за время не $O(D^2)$, как в приведённом выше иллюстративном примере, а $O(D\ log^2(D))$, если применить методы быстрых полиномиальных преобразований, подобные [3] и [4], но это возможное улучшение касается использования результатов вычислений и не затрагивает существо предлагаемого алгоритма и его временну́ю оценку сложности. Последнюю легко получить из анализа листинга. В главном цикле, выполняющемся $D-1$ раз, на каждой итерации вычисляется $D+1$ значение s многочлена в заданных узлах, таков же порядок сложности нормировки этих отсчётов. Коэффициенты a образуют треугольную матрицу, поэтому объём связанных с ними вычислений уполовинивается: $(D+1)^2/2$. Таким образом, алгоритм имеет квадратическую сложность по D. Очевидно, меньше она быть не может, так как всего вычисляется $(D+1)^2$ значений набора ортогональных многочленов (каждый многочлен в каждом узле). Поскольку в большинстве современных процессоров деление выполняется в несколько раз дольше умножения, число делений минимизировано – всего одно на итерацию.

Список литературы

[1] Никифоров А.Ф., Суслов С.К., Уваров В.Б., Классические ортогональные полиномы дискретной переменной. – М.: Наука, 1985. – 216 с., ил.

[2] Orthogonal Polynomials and Special Functions: Computation and Applications. / Eds. Marcellan F., Van Assche W. – Lecture Notes in Mathematics 1883, Springer, 2006. – 418 p.

[3] Driscoll J., Healy D., Rockmore D., Fast Discrete Polynomial Transforms with Applications to Data Analysis for Distance Transitive Graphs // SIAM J. Comput. Vol. 26, 1996, pp. 1066–1099.

[4] Potts D., Steidl G., Tasche M. - Fast Algorithms for Discrete Polynomial Transforms // Mathematics of Computation, Vol. 67, No 224, Oct. 1998, pp. 1577-1590.

Generation of the ensemble of discrete orthogonal polynomials with a given node set

Parfenov D.V.

A computationally efficient method for generation the ensemble of discrete orthogonal polynomials of complex variable is motivated and proposed. The ensemble keeps orthogonality at a given arbitrary separate node set with unitary weight function. The detailed algorithm implementation for coefficient computation and polynomial sampling is presented in GNU Octave/Mathworks Matlab language. The complexity estimate is the least attainable.

Keywords: digital signal processing, robust interpolation, orthogonal polynomials of discrete complex variable, arbitrary nodes, coefficient computation, optimal algorithm, computational complexity.

Часть 3.
Математические модели

Письмо в редакцию по поводу статьи З.А.Ниязовой "Расшифровка арифметических сумм монотонных конъюнкций"

Быстрыгова А.В.

В статье З.А.Ниязовой "Расшифровка арифметических сумм монотонных конъюнкций" получена точная оценка сложности расшифровки функции, имеющей не более двух нижних единиц. На самом деле, эту оценку можно понизить на 1 запрос, что и демонстрируется в данной работе.

Ключевые слова: точная расшифровка, суммы монотонных конъюнкций, запросы на значение.

Глубокоуважаемая редакция, прошу принять к рассмотрению замечание по работе [1]. В [1], согласно теореме 2, для случая $p = 2$ справедливо $\varphi(n, 2) \geq 2n$ и верно следствие 1, в котором говорится, что $\varphi(n, 2) = 2n$. На самом деле, $\varphi(n, 2) \leq 2n - 1$.

Далее сформулируем несколько определений, а также напомним некоторые из работы [1].

Под $\wedge$ будем понимать операцию логического И, то есть $a \wedge b$ имеет вектор значений (0001).

Пусть даны наборы $a = (a_1, a_2, \ldots, a_n)$ и $b = (b_1, b_2, \ldots, b_n)$. Будем говорить, что набор $a \leq b$, если для любого i, $1 \leq i \leq n$ выполняется неравенство $a_i \leq b_i$.

Под подкубом набора x будем понимать подкуб, составленный из наборов a, удовлетворяющих отношению $a \leq x$.

Запросом к загаданной функции $f(x_1, x_2, \ldots, x_n)$ является набор $(a_1, a_2, \ldots, a_n)$, а ответ на запрос $f(a_1, a_2, \ldots, a_n)$ равен количеству нижних единиц, которые лежат в подкубе набора $(a_1, a_2, \ldots, a_n)$.

Под 0^i будем понимать набор, где все компоненты кроме i-й равны 1, а i-я равна 0.

Под $0^{i,j,A}, i \neq j$, $i, j \notin A$, будем понимать набор, где все компоненты кроме компонент с номерами i, j и компонент с номерами из множества A равны 1, а остальные переменные равны 0.

Утверждение 1. *Справедливо неравенство* $\varphi(n, 2) \leq 2n - 1, n \geq 2$.

Доказательство. Пусть загадана $f(x_1, x_2, \ldots, x_n)$ с не более, чем двумя нижними единицами. Запросим значение на всех наборах вида $0^i, i = 1, 2, \ldots, n$, соответственно потратив в точности n запросов. Возможны следующие случаи.

1) Ответы на всех запрошенных наборах равны 0.

   Запросим значение функции на наборе $(1, \ldots, 1)$. Если ответ на запрос равен 1, значит $f(x_1, x_2, \ldots, x_n) = x_1 \wedge x_2 \wedge \ldots \wedge x_n$. Иначе, $f(x_1, x_2, \ldots, x_n) \equiv 0$.

   Следовательно, для восстановления вектора значений функции f суммарно было сделано не более $n + 1$ запросов.

2) Ответ хотя бы на одном из наборов равен 1 или 2.

   Иными словами, хотя бы одна нижняя единица у функции точно есть, и она не находится в вершине $(1, \ldots, 1)$.

   Причем, выполнено следующее.

   а) Если $f(0^i) = 0$, то у всех нижних единиц i-я компонента равна 1.

   б) Если $f(0^i) = 2$, то у функции две нижние единицы и их i-я компонента равна 0.

   в) Если $f(0^i) = 1$, то, если у функции ровно одна нижняя единица, ее i-я компонента равна 0, а если нижних единиц две, то i-я компонента этих нижних единиц будет отличаться.

   Обозначим через q количество запросов, на которые ответ не равен 1.

   Если у функции ровно одна нижняя единица, она лежит в подкубе набора 0^i для некоторого i. Если нижних единиц две, тогда хотя бы в одной компоненте они отличаются. Следовательно, на какой-то из заданных 0^i запросов ответ равен 1.

   Рассмотрим подкуб, для всех наборов которого верно следующее: компоненты с номерами из множества $A_0 = \{i | 1 \leq i \leq n, f(0^i) = 0\}$

установлены в 1, а компоненты с номерами из множества $A_2 = \{i | 1 \leq i \leq n, f(0^i) = 2\}$ установлены в 0. В этом подкубе лежат все нижние единицы загаданной функции. Берем любое i, для которого верно равенство $f(0^i) = 1$. Заметим, что в подкубе набора 0^i лежит одна нижняя единица и ее i-я компонента равна 0.

Зададим $n-1-q$ запросов вида $0^{i,j,A_2}, j \in \{1, \ldots, n\} \setminus (A_0 \cup A_2 \cup \{i\})$. Если $f(0^{i,j,A_2}) = 0$, то j-я компонента искомой нижней единицы равна 1. Если $f(0^{i,j,A_2}) = 1$, то j-я компонента искомой нижней единицы равна 0.

Следовательно, задав дополнительно $n - 1 - q$ запросов, мы восстановим значения всех компонент одной нижней единицы. Если первая найденная нижняя единица лежит на нижнем слое рассматриваемого подкуба, то у функции f одна нижняя единица, иначе их две и вторая восстанавливается по первой.

В результате, мы разобрали все случаи и получили, что для восстановления функции было задано $n + (n - 1 - q)$ $(q \geq 0)$ запросов.

□

Список литературы

[1] Ниязова З. А. Расшифровка арифметических сумм монотонных конъюнкций // Интеллектуальные системы. Теория и приложения. — 2015. — Т. 19, вып. 4. — С. 169–195.

Letter to the editor concerning the paper by Z. A. Niyazova "Learning of arithmetic sum of monotone conjunctions"

Bistrigova A.V.

The complexity of learning functions with respect to the number of monotone conjunctions in their representation is studied in the paper "Learning of arithmetic sum of monotone conjunctions" by Z. A. Niyazova. In particular, lower and upper bounds on the number of queries one needs to learn the function, having no more than 2 conjunctions, are presented there. Here, we show that this result can be improved upon by one query.

Keywords: exact learning, sum of monotone conjunctions, membership queries.

Классическая истинность всех абсолютно арифметически реализуемых предикатных формул.

Коновалов А. Ю.

Доказывается, что всякая абсолютно арифметически реализуемая предикатная формула является классически истинной, однако не всякая классически истинная предикатная формула является абсолютно арифметически реализуемой.

Ключевые слова: конструктивная семантика, реализуемость, арифметическая реализуемость, абсолютная реализуемость, формальная арифметика.

В статье [1] автором была определена семантика абсолютной арифметической реализуемости для предикатных формул. В настоящей работе устанавливается связь между абсолютной арифметической реализуемостью и классической истинностью.

Будем считать, что язык формальной арифметики LA содержит функциональные символы для всех примитивно-рекурсивных функций, а также константы для всех натуральных чисел. Атомарные формулы (атомы) языка LA суть выражения $t_1 = t_2$, где t_1 и t_2 — термы. Более сложные формулы языка LA строятся обычным образом из атомов при помощи логических связок $\wedge$, $\vee$, $\rightarrow$, $\neg$ и кванторов $\exists$, $\forall$. Формулы языка LA будем называть арифметическими формулами.

Пусть фиксировано натуральное число $n \geq 1$. *Унифомизацией* формулы $\Phi(x_1, \ldots, x_n, y)$ языка LA, не содержащей параметров, отличных от $x_1, \ldots, x_n, y$, будем называть формулу

$$\Phi(x_1, \ldots, x_n, y) \wedge (\forall z < y)\, \neg\Phi(x_1, \ldots, x_n, z),$$

которую обозначим $\Phi^U(x_1, \ldots, x_n, y)$. Каждая такая формула задает частичную функцию $f : \mathbb{N}^n \rightarrow \mathbb{N}$, где $f(k_1, \ldots, k_n) = k$, если и только если $\mathbb{N} \models \Phi^U(k_1, \ldots, k_n, k)$, т. е. формула $\Phi^U(k_1, \ldots, k_n, k)$ истинна в стандартной интерпретации. Пусть фиксирована геделева нумерация всех формул языка LA. Формулу с геделевым номером k обозначаем Φ_k. Если

k — геделев номер такой формулы LA, которая не содержит параметров, отличных от $x_1, \ldots, x_n, y$, то посредством φ_k^n обозначим n-местную частичную функцию, задаваемую формулой Φ_k^U. Частичные функции, которые могут быть представлены в виде φ_k^n для некоторых n и k, будем называть арифметическими.

Предикатные формулы строятся обычным образом из атомов $P(v_1, \ldots, v_n)$, где P есть n-местная предикатная переменная, а $v_1, \ldots, v_n$ — предметные переменные, при помощи логических констант $\top$ (истина), $\bot$ (ложь), связок $\wedge$, $\vee$, $\to$ и кванторов $\forall$, $\exists$.

Пусть фиксированы примитивно-рекурсивные двухместная функция c, которая взаимно однозначно нумерует все пары натуральных чисел, и одноместные обратные функции p_1 и p_2, так что выполняются соотношения $\mathsf{p}_1(\mathsf{c}(x,y)) = x$ и $\mathsf{p}_2(\mathsf{c}(x,y)) = y$. В выражениях вида $\mathsf{p}_1(t)$, $\mathsf{p}_2(t)$ обычно будем опускать скобки.

Следуя [2], n-местным обобщенным предикатом будем называть всякую функцию типа $\mathbb{N}^n \to 2^{\mathbb{N}}$. Пусть A — предикатная формула, f — отображение, которое каждой предикатной переменной из A ставит в соответствие обобщенный предикат той же валентности. В этом случае отображение f будем называть оценкой формулы A. Временно введем в язык логики предикатов константы для обозначения всех натуральных чисел. Формулы с этими константами будем называть предикатными формулами расширенного языка.

Определим отношение $e\,\mathbf{r}_f^{\mathsf{ar}}\,A$ (натуральное число e арифметически реализует предикатную формулу расширенного языка A при оценке f этой формулы), индукцией по построению формулы A:

- для любого натурального числа e верно $e\,\mathbf{r}_f^{\mathsf{ar}}\,\top$;
- для любого натурального числа e неверно $e\,\mathbf{r}_f^{\mathsf{ar}}\,\bot$;
- $e\,\mathbf{r}_f^{\mathsf{ar}}\,P(a_1, \ldots, a_n) \rightleftharpoons e \in f(P)(a_1, \ldots, a_n)$,
  если P — n-местный предикатный символ;
- $e\,\mathbf{r}_f^{\mathsf{ar}}\,(A \wedge B) \rightleftharpoons \mathsf{p}_1 e\,\mathbf{r}_f^{\mathsf{ar}}\,A$ и $\mathsf{p}_2 e\,\mathbf{r}_f^{\mathsf{ar}}\,B$;
- $e\,\mathbf{r}_f^{\mathsf{ar}}\,(A \vee B) \rightleftharpoons (\mathsf{p}_1 e = 0$ и $\mathsf{p}_2 e\,\mathbf{r}_f^{\mathsf{ar}}\,A)$ или $(\mathsf{p}_1 e = 1$ и $\mathsf{p}_2 e\,\mathbf{r}_f^{\mathsf{ar}}\,B)$;
- $e\,\mathbf{r}_f^{\mathsf{ar}}\,(A \to B) \rightleftharpoons \forall a\,(a\,\mathbf{r}_f^{\mathsf{ar}}\,A \Rightarrow$ определено $\varphi_e^1(a)$ и $\varphi_e^1(a)\,\mathbf{r}_f^{\mathsf{ar}}\,B)$;
- $e\,\mathbf{r}_f^{\mathsf{ar}}\,\exists x A(x) \rightleftharpoons \mathsf{p}_2 e\,\mathbf{r}_f^{\mathsf{ar}}\,A(\mathsf{p}_1 e)$;
- $e\,\mathbf{r}_f^{\mathsf{ar}}\,\forall x\,B(x) \rightleftharpoons \forall k\,($определено $\varphi_e^1(k)$ и $\varphi_e^1(k)\,\mathbf{r}_f^{\mathsf{ar}}\,B(k))$.

Замкнутую предикатную формулу A будем называть абсолютно арифметически реализуемой, если для всякой оценки f этой формулы найдется такое натуральное число e, что имеет место $e\,\mathbf{r}_f^{\mathsf{ar}}\,A$.

Верны следующие теоремы.

Теорема 1. *Всякая абсолютно арифметически реализуемая предикатная формула является классически истинной.*

Теорема 2. *Существует классически истинная предикатная формула, которая не является абсолютно арифметически реализуемой.*

Отметим, что для обычной рекурсивной реализуемости аналог теоремы 1 неверен. (см. [3], [4, теорема 3]).

Список литературы

[1] Коновалов А. Ю. Арифметическая реализуемость и базисная логика // Вестн. Моск. ун-та. Матем. Механ., 2016. №1, стр. 52–56.

[2] Плиско В. Е. Абслолютная реализуемость предикатных формул // Изв. АН СССР. Сер. матем. 1983. **47**, №2. 315–334.

[3] Оревков В. П. Связь конструктивной общезначимости с выводимостью в классическом исчислении предикатов. // Всесоюзный симпозиум по матем. логике (тезисы докладов), Алма-Ата, 1969, стр. 35.

[4] В. Е. Плиско. Неарифметичность класса реализуемых предикатных формул. // Изв. АН СССР. Сер. мат., 1977, т. 41, № 3, стр. 483—502.

All absolute arithmetically realizable predicate formulas are classically true.

Konovalov A. Yu.

It is proved that every absolute arithmetically realizable predicate formula is classically true, but there is a classically true predicate formula that is not absolute arithmetically realizable.

Keywords: constructive semantics, realizability, arithmetic realizability, absolute realizability, formal arithmetic.

Приведенные критериальные системы предполных классов в классах линейных автоматов над конечными полями

Часовских А.А.

Найдены множества всех предполных классов в классах линейных автоматов над конечными полями, являющиеся приведенными критериальными системами в этих классах.

Ключевые слова: конечный автомат, линейный автомат, операции композиции, операции суперпозиции, обратная связь, проблема полноты, предполный класс, критериальная система, приведенная критериальная система, сумматор, задержка.

Мы будем использовать понятия и обозначения, введенные в работах [1] – [3]. Конечное поле, содержащее $k = p^m$ элементов, где p – простое число, а m – натуральное число, обозначим E_k. Кольцо многочленов переменной ξ с коэффициентами из E_k обозначим $E_k[\xi]$, а поле, полученное из E_k путем трансцендентного расширения переменной ξ обозначаем $E_k(\xi)$. Это поле состоит из дробей, числитель и знаменатель которых являются взаимнопростыми многочленами из $E_k[\xi]$. Подкольцо поля $E_k(\xi)$, состоящее из дробей, знаменатели которых не делятся на ξ, обозначим $E'_k(\xi)$. Кольцо формальных степенных рядов переменной ξ над полем E_k обозначим $R_k(\xi)$. Это кольцо содержит подкольцо, состоящее из рядов, коэффициенты которых образуют периодическую (с предпериодом) последовательность, изоморфное кольцу $E'_k(\xi)$.

Входные переменные и переменная, приписанная выходу линейного автомата принимают значения из кольца $R_k(\xi)$. Лиинейным автоматом мы называем отображение $f(x_1, x_2, \ldots, x_n)$, для которого в $E'_k(\xi)$ найдутся такие элементы μ_i, $i = 0, 1, \ldots, n$, что для любых α_i, $\alpha_i \in R_k(\xi)$, $i = 1, 2, \ldots, n$ выполнено равенство:

$$f(\alpha_1, \alpha_2, \ldots, \alpha_n) = \sum_{i=1}^{n} \mu_i \alpha_i + \mu_0. \qquad (1)$$

Переменная x_i этого автомата называется существенной, если $\mu_i \neq 0$, эта переменная называется непосредственной, если $\mu_i(0) \neq 0$. Через $U(f)$ будем обозначать $\{\ \mu_i \mid i = 1, 2, \ldots, n\ \}$, а через $C(f)$ будем обозначать множество с одним элементом: $C(f) = \{\ \mu_0\ \}$.

Множество всех линейных автоматов над полем E_k обозначим $\mathfrak{L}_k$. Множество $\mathfrak{L}_k$ вместе с опрерациями суперпозиции и обратной связи представляет собой класс линейных автоматов [4] над полем E_k с операциями композиции.

Для множества M линейных автоматов полагаем: $U(M) = \cup_{f \in M} U(f)$.

Наша цель найти все предполные классы [5] в классе $\mathfrak{L}_k$ для случая $m > 1$, так как случай простого поля был рассмотрен ранее в работе [6].

В дальнейшем нам понадобятся следующие подмножества линейных автоматов.

$$T_a = \{\ f(x_1, x_2, \ldots, x_n) \mid n \in \mathbb{N},\ f \in \mathfrak{L}_k,\ \text{из}$$

$$f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} \mu_i x_i + \mu_0 \ \text{следует}$$

$$\left. \sum_{i=1}^{n} \mu_i(0) \cdot a + \mu_0(0) = a \ \right\},$$

где $a \in E_k$.

$$V_1 = \{\ f(x_1, x_2, \ldots, x_n) \mid n \in \mathbb{N},\ f \in \mathfrak{L}_k,$$

$$f \ \text{имеет не более одной непосредственной переменной}\ \}.$$

$$V_p = \{\ f(x_1, x_2, \ldots, x_n) \mid n \in \mathbb{N},\ f \in \mathfrak{L}_k,\ \text{и из}$$

$$f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} \mu_i x_i + \mu_0 \ \text{следует}$$

$$\left. \sum_{i=1}^{n} \mu_i(0) = 1 \right\}.$$

Разложим число m в произведение различных простых чисел q_s, $s = 1, 2 \ldots, l$:

$$m = q_1^{r_1} \cdot q_2^{r_2} \ldots q_l^{r_l}. \tag{2}$$

Для каждого s, $s = 1, 2 \ldots, l$, в поле E_k содержится подполе E_{k_s} из $k_s = \frac{k}{q_s}$ элементов [7]. Положим:

$$P_s = \{ \ f(x_1, x_2, \ldots, x_n) \ \mid \ n \in \mathbb{N}, \ \ f \in \mathfrak{L}_k,$$

$$f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} \mu_i x_i + \mu_0,$$

$$\mu_i(0) \in E_{k_s} \ \ i, \ \ i = 1, 2, \ldots, n \quad \}.$$

Для дроби $\mu \in E'_k(\xi)$, $\mu = \frac{u}{v}$, такой, что выполнено $\deg u \leq \deg v$, найдется целое неотрицательное число r и найдутся a, a', b, b', u', v', $s \in \mathbb{N}$, $\quad a, a' \in E_k$, $\quad b, b' \in E_k \setminus \{0\}$, $\quad u', v' \in E_k[\xi]$, $\quad \deg u' < s - 1$, $\deg v' < s - 1$, такие, что имеет место равенство:

$$\mu = \frac{a + \xi u' + a'\xi^s}{b + \xi v' + b'\xi^s}.$$

Тогда положим: $\Psi_0(\mu) = \left(\frac{a}{b}, \frac{a'}{b'}\right)$.

Для каждого автоморфизма ω поля E_k определим следующие множества:

$$M_\omega^{(1)} = \Big\{ \ \mu \ \mid \ \mu \in E'_k(\xi), \ \ \mu = \frac{u}{v},$$

$$\deg u \leq \deg v, \ \ \Psi_0(\mu) = (\mu(0), \omega(\mu(0))) \ \},$$

$$M_\omega = \Big\{ \ f \ \mid \ f \in \mathfrak{L}_k, \ \ U(f) \subseteq M_\omega^{(1)} \ \Big\}. \tag{3}$$

Множество всех автоморфизмов поля E_k будем обозначать Ω.

Занумеруем все неприводимые приведенные многочлены из $E_k[\xi]$: $p_1, p_2, \ldots$ так, что $p_1 = \xi$.

Если дробь μ, $\mu \in E'_k(\xi)$, $\mu = \frac{u}{v}$ представлена в несократимом виде и для некоторого j, $j \in \{2, 3, \ldots\}$, и v не делится на p_j, то найдется, и притом однозначно многочлен u', $\deg u' \leq \deg(p_j)$, такой, что для некоторого μ' из $E'_k(\xi)$, знаменатель которой не делится на p_j, имеет место равенство:

$$\mu = u' + \xi p_j \mu'.$$

При этом положим: $\Psi_j(\mu) = u'$.

В дальнейшем будем использовать следующие множества линейных автоматов.

$$M_j = \{ \ f(x_1, x_2, \ldots, x_n) \ \mid \ n \in \mathbb{N}, \ \ f \in \mathfrak{L}_k,$$

$$f\left(x_1, x_2, \ldots, x_n\right) = \sum_{i=1}^{n} \mu_i x_i + \mu_0,$$

$$\Psi_j(\mu_i) \in E_k,\ \ i = 1, 2, \ldots, n\ \ \}\,.$$

Положим далее:

$$M_1 = \{\ \ f\left(x_1, x_2, \ldots, x_n\right)\ \ |\ \ n \in \mathbb{N},\ \ f \in \mathfrak{L}_k,$$

$$f\left(x_1, x_2, \ldots, x_n\right) = \sum_{i=1}^{n} \mu_i x_i + \mu_0,$$

$$\mu_i(\xi) - \mu_i(0) \in \xi^2 \cdot E_k'(\xi),\ \ i = 1, 2, \ldots, n\ \ \}\,.$$

Введем некоторые классы одноместныхлинейных автоматов:

$$M_0^{(1)} = \left\{\ \ \mu\ \ |\ \ \mu \in E_k'(\xi),\ \ \mu = \frac{u}{v},\ \ \deg u \le \deg v \right\},$$

$$\tilde{M}_0^{(1)} = \left\{\ \ \mu\ \ |\ \ \mu \in E_k'(\xi),\ \ \mu = \frac{u}{v},\ \ \deg u < \deg v \right\},$$

$$M_1^{(1)} = \{\ \ \mu\ \ |\ \ \mu \in E_k'(\xi),\ \ \mu - \mu(0) \in \xi^2 \cdot E_k'(\xi)\ \ \}\,,$$

$$M_j^{(1)} = \left\{\ \ \mu\ \ |\ \ \mu \in E_k'(\xi),\ \ \mu = \frac{u}{v},\ \ (u, v) = 1,\ \ p_j\ \text{не делит}\ \ v\ \ \right\},$$

$$\tilde{M}_j^{(1)} = \left\{\ \ \mu\ \ |\ \ \mu \in E_k'(\xi),\ \ \mu = \frac{u}{v},\ \ (u, v) = 1,\ \ p_j\ \text{делит}\ \ u\ \ \right\},$$

$j = 2, 3, \ldots,$

$$R_j^e = \left\{\ \ f\ \ |\ \ f \in \mathfrak{L}_k,\ \ f\left(x_1, x_2, \ldots, x_n\right) = \sum_{i=1}^{n} \mu_i x_i + \mu_0,\right.$$

$\forall i, i = 1, 2, \ldots, n,$ если x_i — единственная существенная

переменная функции f, то $\mu_i \in M_j^{(1)}$,

в противном случае: $\left.\mu_i \in \tilde{M}_j^{(1)}\ \ \right\},$

$$R_j^r = \left\{\ \ f\ \ |\ \ f \in \mathfrak{L}_k,\ \ f\left(x_1, x_2, \ldots, x_n\right) = \sum_{i=1}^{n} \mu_i x_i + \mu_0,\right.$$

$\forall i, i = 1, 2, \ldots, n,$ если x_i — единственная непосредственная

переменная функции f, то $\mu_i \in M_j^{(1)}$,

$$\text{в противном случае: } \mu_i \in \tilde{M}_j^{(1)} \ \Big\},$$

$j = 0, 2, 3, \ldots$.

Степенью дроби μ, $\mu = \frac{u}{v}$, из $E_k(\xi)$ будем называть, как принято, число $\deg(\mu)$, равное максимуму из степеней ее числителя и знаменателя. Положим:

$$Q = \{ \ \mu \ | \ \mu \in E'_k(\xi), \ \deg(\mu) = 1, \}. \tag{4}$$

Постое трансцендентное расширение поля E_{k_s}, $s \in \{1, 2, \ldots, l\}$, элементом μ из множества Q обозначим $E_{k_s}(\mu)$. В дальнейшем мы используем множества $B_{\mu,s}$:

$$B_{\mu,s} = \Big\{ \ f \ | \ f \in \mathfrak{L}_k, \ \ f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} \mu_i x_i + \mu_0,$$

$$\forall i, i = 1, 2, \ldots, n, \ \ \mu_i \in E_{k_s}(\mu) \ \Big\}.$$

Нам понадобится множество $\tilde{J}_k$,

$$\tilde{J}_k = \{ \ V_1, V_p, P_s, T_a, M_\omega, M_j, R_i^e, R_i^r, B_{\mu,s} \ |$$

$$s \in \{1, 2, \ldots, l\}, a \in E_k, \omega \in \Omega, j \in \{1, 2, 3, \ldots\},$$

$$i \in \{0, 2, 3, \ldots\}, \mu \in Q\}$$

Следующее утверждение без труда доказывается индуцией по построению.

Лемма 1. *Множество $\tilde{J}_k$ состоит из замкнутых в $\mathfrak{L}_k$ классов, не совпадающих с $\mathfrak{L}_k$.*

Доказательство. Замкнутость классов V_1, V_p, T_a, M_j, R_i^e, R_i^r доказывается также, как и для аналогичных классов в случае простого поля.

Для доказательства замкнутости классов P_s, M_ω и $B_{\mu,s}$ будем использовать замыкание $K^{(1)}$ над подмножествами из $E'_k(\xi)$, которое определено в [1]. Там же показано, что для любого M, $M \subseteq \mathfrak{L}_k$, выполнены соотношения:

$$U(K(M)) \subseteq K^{(1)}(U(M)). \tag{5}$$

Нетрудно видеть, что все три операции оператора замыкания $K^{(1)}$ сохраняют множества $E'_k(\xi) \cap M_\omega^{(1)}$ и $E'_k(\xi) \cap E_{k_s}(\mu)$. Поэтому множества M_ω и $B_{\mu,s}$ являются замкнутыми классами.

Рассмотрим класс P_s. Если $\mu_i \in E'_k(\xi)$ и при этом $\mu_i(0) \in E_{k_s}$, $i = 1, 2$, то

$$(\mu_1 + \mu_2)(0) \in E_{k_s},$$

$$(\mu_1\mu_2)(0) \in E_{k_s},$$

и в случае, если к паре (μ_1, μ_2) применима операция "Об", то есть, если $\mu_2(0) = 0$, то

$$\text{Об}(\mu_1, \mu_2)(0) = \frac{\mu_1}{1 - \mu_2}(0) = \mu_1(0) \in E_{k_s}.$$

Таким образом, согласно соотношению (5), свободный член любого ряда из $U(K(P_s))$ содержится в E_{k_s}, то есть класс P_s замкнут.

Таким образом, все множества из $\tilde{J}_k$ являютяс замкнутыми классами. Для завершения доказательства леммы для каждого класса из рассматриваемого множества приведем пример линейного автомата, не содержащегося в этом классе:

$$x_1 + x_2 \notin V_1 \cup V_p \cup \left(\bigcup_i R_i^e\right) \cup \left(\bigcup_i R_i^r\right),$$

$$1 \notin T_0,$$

$$\xi x \notin \left(\bigcup_{a, a \neq 0} T_a\right) \cup \left(\bigcup_{\omega} M_\omega\right) \cup \left(\bigcup_j M_j\right),$$

пусть b — примитивный элемет поля E_k, тогда

$$bx \notin \left(\bigcup_s P_s\right) \cup \left(\bigcup_{(\mu,s)} B_{\mu,s}\right).$$

Лемма доказана.

В дальнейшем мы выделим из множества $\tilde{J}_k$ некоторое подмножество, являющееся приведенной критериальной системой в $\mathfrak{L}_k$, которое также окажется множеством всех предполные классы в $\mathfrak{L}_k$.

Удалив из множества $\tilde{J}_k$ замкнутые классы семейства $\{ B_q \mid q \in Q \}$, получим множество $\hat{J}_k$.

Лемма 2. *Для любых различных классов Θ и Θ' из множества $\hat{J}_k$ выполнено:*

$$\Theta \not\subset \Theta'. \tag{6}$$

Доказательство. Для каждого Θ, $\Theta \in \hat{J}_k$, укажем такое множество $\hat{\Theta}$, что

$$\hat{\Theta} \subset \Theta, \tag{7}$$

но для любого Θ', $\Theta' \in \hat{J}_k \setminus \{\Theta\}$, выполнено:

$$\hat{\Theta} \not\subseteq \Theta'. \tag{8}$$

Пусть b — примитивный элемент поля E_k. Положим:

$$\hat{V}_1 = \{\ \xi x_1 + \xi x_2 + 1,\ \ bx\ \},$$

$$\hat{V}_p = \{\ \xi x_1 + x_2 + 1,\ \ bx_1 + bx_2 + \cdots + bx_p + x_{p+1}\ \}.$$

Обозначим через b_s элемент поля E_{k_s}, являющийся примитивным элементом этого поля,

$$\hat{P}_s = \{\ b_s x_1 + b_s x_2,\ \ \xi x_1 + \xi x_2,\ \ 1\ \},$$

$s = 1, 2, \ldots, l$.

$$\hat{T}_a = \{\ (p-1)bx_1 + bx_2 + a,\ \ \xi x_1 + \xi x_2 + a\ \},$$

$$\hat{M}_{id} = \left\{\ bx_1 + x_2,\ \ \frac{\xi}{1+\xi^2}x_1 + \frac{\xi}{1+\xi^2}x_2,\ \ 1\ \right\},$$

$$\hat{M}_{\omega} = \left\{\ \frac{b+\omega(b)\xi}{1+\xi}x_1 + \frac{b+\omega(b)\xi}{1+\xi}x_2,\ \ \frac{\xi}{1+\xi^2}x_1 + \frac{\xi}{1+\xi^2}x_2,\ \ 1\ \right\},$$

если $\omega \in \Omega \setminus \{id\}$.

$$\hat{M}_j = \{\ (b + \xi p_j)\, x_1 + (b + \xi p_j)\, x_2,\ \ \xi p_j x_1 + \xi p_j x_2,\ \ 1\ \},$$

$j = 1, 2, \ldots,$

$$\hat{R}_0^e = \left\{\ \frac{\xi}{1+\xi}x,\ \ \frac{b}{1+\xi}x_1 + \frac{b}{1+\xi}x_2,\ \ 1\ \right\},$$

$$\hat{R}_i^e = \left\{\ \xi x,\ \ b\frac{p_i}{p_i(0)}x_1 + p_i x_2,\ \ 1\ \right\},$$

$i = 2, 3 \ldots,$

$$\hat{R}_0^r = \left\{\ \frac{\xi}{1+\xi^2}x_1 + bx_2,\ \ \frac{b}{1+\xi}x_1 + \frac{b}{1+\xi}x_2,\ \ 1\ \right\},$$

$$\hat{R}_i^r = \{\ bx_1 + \xi p_i x_2,\ \ p_i x_1 + p_i x_2,\ \ 1\ \},$$

$i = 2, 3 \ldots,$

Лемма 3. *Пусть* $M \subseteq \mathfrak{L}_k$ *и для любого* Θ, $\Theta \in \hat{J}_k$, *выполнено:*

$$M \not\subseteq \Theta. \tag{9}$$

Тогда для любого j, $j = 0, 1, 2, \dots$ справедливо:

$$U\left(K(M)\right) \not\subseteq M_j^{(1)}. \tag{10}$$

Доказательство. Рассмотрим подмножество M множества $\mathfrak{L}_k$ такое, что $\forall \Theta, \Theta \in \hat{J}_k$, справедливо (9). Соотношение (10) для $j = 1$ вытекает из определения класса M_1.

Поэтому будем рассматривать значения j из множества $\{0, 2, 3, \dots\}$. Для замыкания множества $U(M)$ по операциям сложения и умножения будем использовать обозначение $S^{(1)}\left(U(M)\right)$. По лемме 12 из работы [1] для каждого j, $j = 0, 2, 3 \dots$, соотношение (10) следует из существования μ_j, $\mu_j \in S^{(1)}(U(M))$, такого, что $\mu_j \notin \tilde{M}_j^{(1)}$ и $\mu_j(0) = 0$.

Если $U\left(M\right) \not\subseteq M_j^{(1)}$, то (10) выполнено. В противном случае, пусть $j = 0$. Из $M \not\subseteq R_0^e$ следует,что в $U(M)$ найдется μ_0', не содержащаяся в $\tilde{M}_0^{(1)}$.

Для некоторых a' и b' из E_k имеем: $\Psi_0(\mu_0') = (a', b')$, $b' \neq 0$. Если $a' = 0$, то дробь μ_0' искомая и соотношение (10) имеет место. Если $a' \neq 0$, то обозначим через $g'(z)$ ненулевой многочлен из $E_p[z]$ с минимальной степенью, для которого $g'(a') = 0$. Если при этом $g'(b') \neq 0$, то искомым является элемент $g'(\mu')$ из $S^{(1)}(U(M))$.

В противном случае, из соотношений $M \not\subseteq P_s$, $s = 1, 2, \dots, l$, следует, что для некоторого примитивного элемента a поля E_k в $U(K(M))$ найдется элемент μ такой, что для некоторого b, $b \in E_k$, справедливо: $\Psi_0(\mu) = (a, b)$. Если $b = 0$, то через r обозначим такое натуральное число, что $a^r = a'$. Тогда дробь $\mu_0' - \mu^r$ — искомая. Если же $b \neq 0$ и b не сопряжено с a, рассмотрим ненулевой многочлен $g(z)$ над E_p степени m такой, что $g(a) = 0$. Если $g(b) \neq 0$, то дробь $g(\mu)$ — искомая. В противном случае, через ω обозначим автоморфизм поля E_k, переводящий элемент a в элемент b. Из соотношения $M \not\subseteq M_\omega$ следует, что в $U(M)$ найдется такое μ'', что для некоторого натурального i выполнено равенство $\Psi_0\left(\mu''\right) = \left(a^i, c\right)$ и $c \neq b^i$. Тогда дробь $\mu^i - \mu''$ является искомой. Таким образом, случай $j = 0$ разобран.

Пусть $j \in \{2, 3, \dots\}$. Множество $U(K(M))$, как было сказано ранее, содержит элемент μ такой, что $a = \mu(0)$ является примитивным элементом поля E_k. Имеем соотношение: $\mu \in M_j^{(1)} \setminus \tilde{M}_j^{(1)}$. Пусть $\Psi_j(\mu) = u$

таково, что $u = a$. Ввиду соотношения $M \not\subseteq M_j$ в M содержится элемент μ'_j такой, что $\Psi_j(\mu') = u'$, $u' \notin E_k$. Если $\mu'(0) = 0$, то дробь μ' — искомая. В противном случае, найдется натуральное число r такое, что $a^r = \mu'(0)$. Тогда дробь $\mu' - \mu^r$ является искомой. Если $u \neq a$, но u делится на p_j, то рассмотрим дробь μ'', такую, что $\mu'' \in U(M) \setminus \tilde{M}_j^{(1)}$, которая существует ввиду соотношения $M \not\subseteq R_j^e$. Найдется натуральное число r' такое, что $\mu''(0) = a^{r'}$. Тогда искомая дробь: $\mu'' - \mu^{r'}$.

Осталось рассмотреть случай, когда многочлен u не является элементом поля E_k и не делится на многочлен p_j. В этом случае если бы многочлены u и u^k давали бы одинаковые остатки от деления на многочлен $\xi \cdot p_j$, то многочлен $u^k - u$ делился бы на p_j. Тогда в поле Γ_j, являющемся алгебраическим расширением поля E_k элементом z таким, что $p_j(z) = 0$, нашелся бы элемент, не содержащийся в поле E_k, порядок которого делил число k. Но [7], в поле Γ_j все элементы, не содержащиеся в E_k, имеют порядок больше k. Отсюда следует, что дробь $\mu^k - \mu$ в последнем рассматриваемом случае является искомой.

Лемма 3 доказана.

Лемма 4. *Если Θ – предполный класс в $\mathfrak{L}_k$, не содержащийся ни в одном из замкнутых классов системы $\hat{J}_k$, то*

$$E_k \not\subseteq E_p\left(U(\Theta)\right). \tag{11}$$

Доказателство. Рассмотрим предполный класс Θ в $\mathfrak{L}_k$, не содержащийся ни в одном из классов множества $\hat{J}_k$. Если соотношение (11) не выполнено, то по теореме Люрота [8] поле $E_p(U(\Theta))$ является простым расширением поля E_k. Поэтому найдется $\mu \in E_k(\xi)$, что справедливо равенство:

$$E_p(U(\Theta)) = E_k(\mu). \tag{12}$$

Не ограничивая общности рассуждений, будем предполагать, что $\mu \in \xi \cdot E'_k(\xi)$. Отсюда и из равенства (12) следует, что $U(\Theta)$ содержится в $K^{(1)}\left(\{\mu, E_k\}\right)$, где оператор замыкания $K^{(1)}$ определен в [1].

Предположим, что $\deg \mu > 1$. Тогда $\mu = \xi\frac{u}{v}$, $u, v \in E_k[\xi]$, $(u, v) = 1$. Если $\deg u \geq 1$, то для некоторого i, $i \in \{1, 2, \dots\}$, получаем включение $\Theta \subseteq M_i$, что противоречит предположению. Если $u \in E_k \setminus \{0\}$ и $\deg v = k' > 1$, то для некоторых a_1, b_0, b_k из $E_k \setminus \{0\}$ имеем: $\mu = \xi\frac{a_1}{b_0 + \xi v' + b_k\xi^k}$, $\deg v' < k' - 1$. Поэтому $\Psi_0(\mu) = (0, 0)$ и $\Theta \subseteq M_{Id}$, где через Id обозначен тождественный автоморфизм поля E_k. Снова получаем противоречие.

Таким образом, $\deg(\mu) = 1$, и, как нетрудно видеть,

$$E_k(\mu) = E_k(\xi).$$

Отсюда, из леммы 3 и теорем 2 и 4 работы [1] вытекает полнота Θ в $\mathfrak{L}$, что противоречит предположению о том, что Θ — предполный класс в $\mathfrak{L}$.

Лемма 4 доказана.

Лемма 5. *Пусть Θ – предполный класс в $\mathfrak{L}_k$, не содержащийся ни в одном из замкнутых классов системы $\hat{J}_k$, и пусть*

$$k_0 = \max_{k'} \{E_{k'} \subseteq E_p(U(\Theta))\}, \tag{13}$$

$k_0 = p^{m_0}$, $m : m_0 = m_1$ и $E_p(a) = E_k$. Тогда для некоторых ω_i, $\omega_i \in E_p(U(\Theta))$, $i = 0, 1, \ldots, m_1 - 1$, выполнено равенство:

$$\xi = \sum_{i=0}^{m_1-1} \omega_i \cdot a^i. \tag{14}$$

Доказательство. Рассмотрим Θ – предполный класс в $\mathfrak{L}_k$, такой, что $\forall \Theta' \in \hat{J}_k$ выполнено: $\Theta \neq \Theta'$. Из леммы 4 следует, что

$$K(\Theta \cup \{a \cdot x\}) = \mathfrak{L}_k.$$

Поэтому

$$E_p(U(\Theta) \cup \{a\}) = E_k(\xi).$$

Согласно [9], $E_k(\xi)$ является линейным пространством над $E_p(U(\Theta))$, порожденным элементами множества $\{a^0, a, a^2, ..., a^{m_1-1}\}$. Отсюда следует равенство (14). Лемма 5 доказана.

Лемма 6. *Пусть Θ – предполный класс в $\mathfrak{L}_k$, не содержащийся ни в одном из замкнутых классов системы $\hat{J}_k$ и выполнено* (13)*. Тогда найдется μ, $\mu \in U(\Theta)$, такое, что*

$$E_{k_0}(\mu) = E_p(U(\Theta)), \tag{15}$$

$$\deg(\mu) = 1 \tag{16}$$

и E_{k_0} – максимальное подполе в E_k.

Доказательство. По лемме 5 в $E_p(U(\Theta))$ найдутся такие ω_i, $i = 0, 1, \ldots, m_1 - 1$, что многочлен $z - \sum_{i=0}^{m_1-1} \omega_i \cdot a^i$ имеет корень $z = \xi$.

Пусть $\omega_i = \frac{u_i}{v_i}$ – несократимые дроби, $u_i \in E_k[\xi]$, $v_i \in E_k[\xi]$, $i = 0, 1, ..., m_1 - 1$. Среди коэффициентов ω_i, $i = 0, 1, \ldots, m_1 - 1$ найдется такой ω_{i_0}, который зависит от ξ, иначе бы ξ, согласно равенству (14), содержалась бы в E_k.

Многочлен $\omega_{i_0} \cdot v_{i_0}(z) - u_{i_0}(z)$ переменной z имеет корень $z = \xi$. Нетрудно видеть, что он делится на многочлен $z - \sum_{i=0}^{m_1-1} \omega_i \cdot a^i$, так как в противном случае элементы поля a^0, a^1,..., a^{m_1-1}, были линейно зависимы над полем $E_p(U(\Theta))$.

Через $\tilde{v}$ обозначим наименьшее общее кратное многочленов v_0, v_1, $\ldots$, v_{m_1-1}.

Многочлен $f(\xi)$, $f(\xi) = u_{i_0} \cdot v_{i_0} - u_{i_0} \cdot v_{i_0}$ переменной ξ делится на многочлен $g(\xi)$, $g = \tilde{v}\left(z - \sum_{i=0}^{m_1-1} \omega_i \cdot a^i\right)$ этой же переменной [8]. Поэтому $\deg_\xi(f(\xi)) \geq \deg_\xi(g(\xi))$. С другой стороны $\deg_\xi(f(\xi)) \leq \deg_\xi(g(\xi))$. Отсюда, $\deg_\xi(f(\xi)) = \deg_\xi(g(\xi))$. Поэтому для некоторого многочлена из $h(z)$, $h(z) \in E_k[z]$, получаем:

$$f(\xi) = h(z) \cdot g(\xi).$$

Разделив многочлен $\omega_{i_0} \cdot v(z) - u(z)$ переменной z на $h(z)$, получим многочлен первой степени от z с коэффициентами из $E_k(\omega_{i_0})$. Отсюда следует, что найдутся такие дроби ω_i', $\omega_i' \in E_{m_1}(\omega_{i_0})$, $i = 0, 1, \ldots, m_1 - 1$, что

$$\xi = \sum_{i=0}^{m_1-1} \omega_i' \cdot a^i.$$

Таким образом,

$$(E_k(\xi) : E_p(U(\Theta))) = (E_k(\xi) : E_{k_0}(\omega_{i_0})) = m_1 \tag{17}$$

и при этом

$$E_{k_0}(\omega_{i_0}) \subseteq E_p(U(\Theta)).$$

Отсюда для $\mu = \omega_{i_0}$ следуетравенство (15).

Из равенства (17), леммы 4 и равенств

$$(E_k(\xi) : E_{k_0}(\mu)) = (E_k(\xi) : E_k(\mu)) \cdot (E_k : E_{k_0}),$$

$$(E_k : E_{k_0}) = m_1$$

получаем:

$$(E_k(\xi) : E_k(\mu)) = 1.$$

Далее, принимая во внимание, что

$$(E_k(\xi) : E_k(\mu)) = \deg(\mu),$$

получаем (16).

Если E_{k_0} не является максимальным подполем в E_k, то для некоторого максимального подполя $E_{k'}$ поля E_k имеем:

$$E_{k_0}(\mu) \subset E_{k'}(\mu) \subset E_k(\xi),$$

$$E_{k_0}(\mu) \neq E_{k'}(\mu) \neq E_k(\xi),$$

поэтому Θ не является предполным классом в $\mathfrak{L}_k$, что противоречит условию леммы.

Лемма 6 доказана.

Следствие 1. *Если Θ — предполный класс в $\mathfrak{L}_k$, не содержащийся ни в одном из замкнутых классов системы $\hat{J}_k$, то для некоторых μ и s, $\mu \in Q$, $s \in \{1, 2, \ldots, l\}$, выполнено: $\Theta = B_{\mu,s}$.*

Отсюда получаем следующее утверждение.

Теорема 1. *Множество замкнутых классов $\tilde{J}_k$ является критериальной системой [5] в $\mathfrak{L}_k$, то есть для любого подмножества M множества $\mathfrak{L}_k$ его полнота в $\mathfrak{L}_k$ равносильна невключению в каждый замкнутый класс множества $\tilde{J}_k$.*

Доказательство. Если множество линейных автоматов M не является полным в $\mathfrak{L}_k$, и не содержится ни в одном из замкнутых классов множества $\hat{J}$, то M содержится в некотором предполном классе, которое, согласно следствию 1, совпадает с некоторым $B_{\mu,s}$. Поэтому любое множество, не являющееся полным, содержится в некотором классе множества $\tilde{J}$.

С другой стороны, множество линейных автоматов M, являющееся полным, не может содержаться ни в одном из классов множества $\tilde{J}$, так как каждый класс этого множества по лемме 1 замкнут и не совпадает с $\mathfrak{L}_k$.

Теорема 1 доказана.

Далее из множества $\tilde{J}$ выделим подмножество, являющееся приведенной критериальной системой, то есть системой замкнутых классов,

удаляя из которой любой класс, получаем множество классов, не являющееся критериальной системой.

С учетом разложения (2), как известно [7], в поле E_k содержится l максимальных подполей: E_{k_i}, $i = 1, 2, \ldots, l$, при этом $k_i = \frac{k}{q_i}$.

Автоморфизм ω поля E_k будем называть минимальным, если он не является тождественным и сохраняет некоторое максимальное подполе поля E_k.

Пару (μ, s), где $\mu \in Q$, $s \in \{1, 2, \ldots, l\}$, будем называть допустимой, если $\mu(0) \notin E_{k_s}$, а также, если $\mu \in M_0^{(1)}$ и $\Psi(\mu) = (a, b)$, то $\omega(a) \neq b$ для любого минимального автоморфизма ω поля E_k, сохраняющего подполе E_{k_s}.

Нетрудно видеть, что для любой допустимой пары (μ, s) поле $E_{k_s}(\mu)$ является собственным подполем поля $E_k(\xi)$.

Положим:

$$J'_k = \{\ B_{\mu,s} \mid \text{пара } (\mu, s) \text{ допустима } \},$$

$$J_k = \hat{J}_k \cup J'_k.$$

Теорема 2. *Множество замкнутых классов J_k является приведенной критериальной системой в $\mathfrak{L}_k$.*

Для доказательства этой теоремы нам понадобятся некоторые вспомогательные утверждения.

Лемма 7. *Для любого замкнутого класса Θ из множества $\tilde{J}_k \setminus J_k$ найдется такой Θ' из J_k, что выполнено включение:*

$$\Theta \subseteq \Theta'.$$

Доказательство леммы 7.

Рассмотрим Θ, $\Theta \in \tilde{J}_k \setminus J_k$. Тогда для некоторых μ и s, $\mu \in Q$, $s \in \{1, 2, \ldots, l\}$, выполнено:

$$\Theta = B_{\mu,s}.$$

При этом имеет место один из следующих двух случаев.

Случай 1. $\mu(0) \in E_{k_s}$.

Случай 2. $\mu \in M_0^{(1)}$ и для некоторого минимального автоморфизма ω поля E_k, сохраняющего элементы подполя E_{k_s}, выполнено: $\Psi(\mu) = (a, \omega(a))$.

Заметим, что в случае 1 выполнено включение $\{\mu, E_{k_s}\} \subset P_s$, поэтому $B_{\mu,s} \subseteq P_s$, а в случае 2 имеет место: $\{\mu, E_{k_s}\} \subset M_\omega$, поэтому $B_{\mu,s} \subseteq M_\omega$.

Лемма 7 доказана.

Из последней леммы вытекает следующее утверждение.

Следствие 2. *Множество J_k является критериальной системой замкнутых классов в $\mathfrak{L}_k$.*

Далее продолжим обоснование приведенности критериальной системы J_k.

Лемма 8. *Если пары (μ, s) и (μ', s') являются допустимыми и*

$$E_{k_{s'}}\left(\mu'\right) \subseteq E_{k_s}\left(\mu\right), \tag{18}$$

то

$$s = s', \tag{19}$$

$$E_{k_s}\left(\mu\right) = E_{k_{s'}}\left(\mu'\right) \tag{20}$$

и для некоторых b_j, $b_j \in E_{k_s}$, $j = 1, 2, 3, 4$, выполнены соотношения:

$$\mu' = \frac{b_1 + b_2\mu}{b_3 + b_4\mu}. \tag{21}$$

Доказательство. Пусть для допустимых пар (μ, s) и (μ', s') имеет место включение (18). Тогда $E_{k_{s'}} \subseteq E_{k_s}$, а из максимальности подполей E_{k_s} и $E_{k_{s'}}$ в E_k следует, что $E_{k_{s'}} = E_{k_s}$. Поэтому равенство (19) справедливо.

Отсюда, из равенства

$$\left(E_k(\xi) : E_{k_s}(\mu')\right) = \left(E_k(\xi) : E_{k_s}(\mu)\right) = q_s$$

и включения (18) получаем равенство (20).

Для некоторых взаимнопростых многочленов $u(\xi)$ и $v(\xi)$ из $E_{k_s}[\xi]$ выполнено:

$$\mu' = \frac{u(\mu)}{v(\mu)}.$$

Если степень дробь $\eta(\xi) = \frac{u(\xi)}{v(\xi)}$ равна r, то

$$\left(E_{k_s}(\mu) : E_{k_s}(\mu')\right) = \left(E_{k_s} : E_{k_s}\right) \cdot r,$$

откуда и из равенства (20) получаем $r = 1$, поэтому выполнено равенство (21).

Лемма 8 доказана.

Следствие 3. *Множество замкнутых классов J'_k является приведенным.*

Из этого следствия и леммы 2 получаем приведенность каждого из множеств J'_k и $\hat{J}_k$. Теперь нужно доказать приведенность объединения этих множеств.

Лемма 9. *Для любого Θ, $\Theta \in J'_k$, и любого Θ', $\Theta' \in \hat{J}_k$, выполнено:*

$$\Theta \not\subseteq \Theta'$$

и

$$\Theta' \not\subseteq \Theta.$$

Доказательство. Рассмотрим класс $B_{\mu,s}$ для допустимой пары (μ, s). Через b обозначим примитивный элемент поля E_{k_s}. Для множества $\hat{B}_{\mu,s}$,

$$\hat{B}_{\mu,s} = \{\ \mu x,\ \ bx_1 + bx_2\ \ , 1\ \}$$

имеем: $\hat{B}_{\mu,s} \subset B_{\mu,s}$, но для любого Θ', $\Theta' \in \hat{J}_k$, выполнено: $\hat{B}_{\mu,s} \not\subseteq \Theta'$. Таким образом, замкнутый класс $B_{\mu,s}$ не содержится ни в каком классе из множества $\hat{J}_k$.

С другой стороны, обозначив через b примитивный элемент поля E_k, заметим, что класс $B_{\mu,s}$ не содержит ни одну из функций следующих функций:

$$bx,$$

$$bx_1 + (1-b)x_2,$$

$$(\mu - \mu(0))x + a, \quad \forall a \in E_k,$$

$$\frac{b\xi}{1+\xi^2}x_1 + \frac{\xi}{1+\xi^2}x_1,$$

$$b\xi p_i x_1 + \xi p_i x_2, \quad i = 1, 2, \ldots,$$

$$bp_i x_1 + p_i x + 2, \quad i = 1, 2, \ldots.$$

Для обоснования этого достаточно двух свойств класса $B_{\mu,s}$:

1) $E_k \not\subseteq U(B_{\mu,s})$;

2) Если $\mu \in U(B_{\mu,s})$, $\mu' \in U(B_{\mu,s})$ и $\frac{\mu}{\mu'} \in E'_k(\xi)$, то $\frac{\mu}{\mu'} \in U(B_{\mu,s})$.

При этом,

$$bx \in V_1,$$

$$bx_1 + (1-b)x_2 \in V_p,$$

$$(\mu - \mu(0))x \in P_s, \quad \forall s, s \in \{1, 2, \dots, l\},$$

$$(\mu - \mu(0))x + a \in T_a, \quad a \in E_k,$$

$$\frac{b\xi}{1+\xi^2}x_1 + \frac{\xi}{1+\xi^2}x_1 \in M_\omega \cap R_0^e \cap R_0^r, \quad \forall \omega \in \Omega,$$

$$b\xi p_i x_1 + \xi p_i x_2 \in M_i, \quad \forall i \in \{1, 2, \dots\},$$

$$bp_i x_1 + p_i x_2 \in R_i^e \cap R_i^r, \quad \forall i \in \{2, 3, \dots\}.$$

Таким образом, лемма 9 доказана.

Используя приведенность множества J_k и ее критериальность, согласно следствию 2, нетрудно получить следующий результат.

Теорема 3. *Множество J_k состоит из предполных в $\mathfrak{L}_k$ классов и каждый предполный в $\mathfrak{L}_k$ класс содержится в множестве J_k.*

Решение задачи нахождения всех предполных классов в $\mathfrak{L}_k$ позволяет построить полиномиальный алгоритм проверки полноты конечных систем линейных автоматов над полем E_k.

Пусть имеется множество M, $M \subset \mathfrak{L}_k$, $|M| < \infty$, причем каждый автомат f из M задан набором коэффициентов при переменных и константной частью $C(f)$.

Шаг 1. Проверяем включения:

$$M \subset V_1,$$

$$M \subset V_p,$$

$$M \subset P_s, \quad s = 1, 2, \dots l,$$

$$M \subset T_a, \quad a \in E_k.$$

Если хотя-бы одно из эих включений выполнено, то множество M не является полным в $\mathfrak{L}_k$, алгоритм заканчивает работу.

Шаг 2. Проверяем, верно ли включение:

$$U(M) \subset M_0^{(1)}.$$

Если нет, то переходим к шагу 4.

Шаг 3. Находим множество

$$\Psi_0(M) = \{\ \Psi_0(\mu)\ \mid\ \mu \in U(M)\ \}.$$

Пусть a — какой-либо примитивный элемент поля E_k. Каждый сопряженный к a элемент b поля E_k порождает автоморфизм ω поля E_k такой, что $\omega(a) = b$. При этом сопряженные к a элементы ищутся как корни минимального многочлена, корнем которого является a. Таким образом, количество рассматриваемых автоморфизмов ограничено числом m [8]. Для каждого автоморфизма ω проверяем включение

$$\Psi_0(M) \subseteq M_\omega.$$

Если хотя бы одно из включений выполнено, то M не является полным в $\mathfrak{L}_k$, алгоритм заканчивает работу.

Шаг 4. Находим наибольший общий делитель D_1 числителей дробей из множества U',

$$U' = \{\ \mu - \mu(0)\ \mid\ \mu \in U(M)\ \}.$$

Если $\deg(D_1) > 1$, то множество M не является полным в $\mathfrak{L}_k$ и алгоритм заканчивает работу.

Шаг 5. Находим наибольший общий делитель D_2 числителей дробей из множества U^e,

$$U^e = \{\ \mu\ \mid\ \text{в}\ M\ \text{содержит автомат, для разложения (1) которого}$$

$$\text{найдется}\ \ i\ \ \text{такое, что}\ \ \mu_i = \mu\ \ \text{и}$$

$$x_i\ \ \text{не является единственной существенной переменной}\ \ f\}.$$

Находим наибольший общий делитель D_3 числителей дробей из множества U^r,

$$U^r = \{\ \mu\ \mid\ \text{в}\ M\ \text{содержит автомат, для разложения (1) которого}$$

$$\text{найдется}\ \ i\ \ \text{такое, что}\ \ \mu_i = \mu$$

$$\text{и}\ \ x_i\ \ \text{не является единственной непосредственной переменной}\ \ f\}.$$

Наименьшее общее кратное знаменателей дробей из $U(M)$ обозначим K_1. Для $i = 2, 3$ наибольший общий делитель D_i и K_1 обозначим D'_i. Если хотя бы для одного i, $i \in \{2, 3\}$, выполнено: $D_i/D'_i \notin E_k$, то $K(M) \neq \mathfrak{L}_k$ и алгоритм заканчивает работу.

Шаг 6. Этот шаг выполняется путем перебора всех допустимых пар (μ, s). Количество таких пар ограничено сверху числом $l \cdot k^3$. Две пары (μ, s) и (μ', s) назовем эквивалентными, если $B_{\mu,s} = B_{\mu',s}$.

Далее, выбирая по одному представителю (μ, s) из каждого класса эквивалентности, выполняем проверку включения $M \subset B_{\mu,s}$. Эта проверка может быть выполнена следующим образом. Сначала выражаем ξ несократимой дробью $\frac{u}{v}$, где u и v — многочлены из кольца $E_k[\mu]$, то есть многочлены переменной μ с коэффициентами из E_k. Далее, для μ' из M проверка включения $\mu' \in B_{\mu,s}$ заключается в подстановке дроби $\frac{u}{v}$ вместо ξ в μ', приведению полученного выражения к виду $\frac{u'}{v'}$, где $u', v' \in E_k[\mu]$ и $v'(0) = 1$. Если при этом оказалось, что все коэффициенты u' и v' содержатся в E_{k_s}, то имеем: $\mu' \in B_{\mu,s}$.

Понятно, что при реализации алгоритма проверку на эквивалентность пар можно не проводить.

Если нашлась такая допустимая пара (μ, s), что $U(M) \subseteq B_{\mu,s}$, то M не является полным в $\mathfrak{L}_k$. Алгоритм заканчивает работу с отрицательным результатом. В противном случае, алгоритм заканчивает работу с положительным результатом.

Для упрощения реализации алгоритма и оптимизации времени ее работы проиндексируем элементы поля E_k числами $0, 1, \ldots, k-1$. Для элементов поля E_k, используя индексы, составим таблицы сложения, умножения, возведения в степень до степени $k - 1$.

Выберем некоторый элемент b поля E_k, являющийся примитивным для этого поля. Для упрощения реализации алгоритма и оптимизации времени его работы проиндексируем элементы поля E_k: элемент 0 индексируем числом 0, каждый ненулевой элемент a индексируем степенью i такой, что $1 \leq i \leq k - 1$ и $b^i = a$. Для элементов поля E_k, используя индексы, составим таблицы сложения, умножения, возведения в степень до степени $k - 1$, таблицу принадлежности максимальным подполям P_s, $s = 1, 2, \ldots, l$. Для каждого элемента сопряженного с b составим также таблицу индексов степеней этого элемента от 1 до $k - 1$.

Также заранее заготовим выражения ξ через каждую из дробей μ первой степени, для которой найдется s, что μ входит в допустимую пару (μ, s).

В качестве параметров для оценки времени работы этого алгоритма выберем следующие:
r — количество функций в множестве M, проверяемом на полноту;
n — максимальное количество переменных в функциях множества M;
d — максимальная степень дробей из множества $U(M)$, при этом, как

принято, $deg\left(\frac{u}{v}\right) = \max(\deg u, \deg v)$;
k — как и ранее, количество элементов конечного поля;
l — количество максимальных подполей в поле E_k.

Несложный анализ приводит к следующей теореме.

Теорема 4. *Полученный алгоритма проверки полноты конечных подмножеств линейных автоматов может быть реализован с временной сложностью* $O\left(rnd^2k^3l\right)$.

Доказательство.

Нетрудно видеть, что шаги 1-3 алгоритма могут быть реализованы с временной сложностью $O(rnk)$, шаги 4 и 5 — с временной сложностью $O\left(rnd^2\right)$, а шаг 6, с временной сложностью $O\left(rnd^2k^3l\right)$.

Теорема 4 доказана.

Список литературы

[1] Часовских, А.А. Проблема полноты для класса линейно-автоматных функций / А. А. Часовских // Дискретная математика. — 2015. — Т. 27, № 2. — С. 134–151.

[2] Часовских, А.А. Критериальные системы в классах линейно-автоматных функций над конечными полями / А. А. Часовских // Интеллектуальные системы. Теория и приложения. — 2015. — Т. 19, вып. 3. — С. 195–207.

[3] Часовских, А.А. Проблема полноты в классах линейных автоматов / А. А. Часовских // Интеллектуальные системы. Теория и приложения. — 2018. — Т. 22, вып. 2. — С. 151–154.

[4] Гилл, А. Линейные последовательные машины / А. Гилл. — М.: Наука, 1974. — 288 с.

[5] Кудрявцев, В. Б. Введение в теорию автоматов / В. Б. Кудрявцев, С. В. Алешин, А. С. Подколзин. — М.: Наука, 1985. — 320 с.

[6] Часовских, А. А. Условия полноты линейно-р-автоматных функций / А. А. Часовских // Интеллектуальные системы. Теория и приложения. — 2014. — Т. 18, вып. 3. — С. 203–252.

[7] Лидл, Р. Конечные поля: в 2 т. / Р. Лидл, Г. Нидеррайтер. — М.: Мир, 1988. — 2 т.

[8] Ван дер Варден, Б. Л. Алгебра / Б. Л. Ван дер Варден. — М.: Наука, 1976. — 648 с.

[9] Зарисский, О. Коммутативная алгебра: в 2 т. / О. Зарисский, П. Самюэль. —М.: ИЛ, 1963. — 2 т.

Reduced criterial system of are precomplete classes in linear automata classes over finite fields

Chasovskikh A.A.

The sets of all precomplete classes in the classes of linear automata over finite fields are found, which are reduced criteria systems in these classes.

Keywords: finite automaton, linear automaton, composition operations, superposition operations, feedback, completeness problem, precomplete class, criterial system, reduced criterial system, adder, delay.

Часть 4.
Материалы семинара «Теория автоматов»

Доклады семинара «Теория автоматов»

В третьем и четвертом кварталах 2018 года на научном семинаре «Теория автоматов» под руководством академика Валерия Борисовича Кудрявцева состоялось 11 докладов.

19 сентября 2018 года

Оптимизация аппаратных реализаций криптографических алгоритмов

аспирант Курганова Е. А.

В последнее время в сферах цифровой обработки сигналов, высокоскоростной передачи данных и криптографии все чаще возникает ситуация, когда программная реализация устройства не может обеспечить необходимую пропускную способность. Поэтому для многих современных цифровых устройств используются интегральные схемы. Помимо этого криптографические стандарты динамично меняются. Также из-за угрозы практической реализации квантовых вычислений в группу риска попадает подавляющее большинство стандартов с открытым ключом. В связи с этим возникает задача построения «криптографического конструктора», из которого можно оперативно порождать криптографические процессоры с эффективной поддержкой необходимой функциональности. Также с этой задачей неразрывно связана еще одна — создание кремниевого компилятора, т.е. программы, которая преобразует высокоуровневое описание алгоритма в схему в технологической библиотеке.

Аппаратные реализации моделируются схемами из функциональных элементов в расширенном базисе — конъюнкция, дизъюнкция, отрицание и задержка. Главным параметром производительности таких реализаций является глубина схемы, т.е. длина максимального простого пути схемы. Также в качестве дополнительного параметра часто рассматривают сложность — общее количество элементов схемы. И в том, и в другом случае элементы отрицания не учитываются при вычислении.

Первая часть доклада посвящена построению современных криптопроцессоров. Рассматриваются аппаратные реализации нескольких широко используемых симметричных шифров (DES, AES, ZUC, ГОСТ Р 34-12.2015), после чего производится сравнение скорости их работы. Приводится аппаратная реализация асимметричного шифра NTRUEncrypt,

обладающего устойчивостью к квантовым атакам, и кратко рассматривается реализация классического асимметричного шифра RSA. Данные реализации также сравниваются по глубине и сложности.

Во второй части доклада рассматривается построение аппаратных реализаций некоторых преобразований, часто используемых в криптографии. Приводится алгоритм для аппаратной реализации системы булевых функций, оптимизированной по сложности, а также результаты применения этого алгоритма к S-блокам (блокам подстановки).

19 сентября 2018 года

Свойства графов автоматов и других разложимых графов

аспирант Ищенко Р. А.

В докладе описываются свойства графов (диаграмм Мура) групповых и дефинитных автоматов, а также приводятся оценки хроматических чисел графов в зависимости от их толщины, охвата и древесности.

Автор рассматривает диаграммы Мура автоматов с точки зрения теории графов. В докладе рассказывается о критериях, позволяющих определить класс автомата по его диаграмме Мура, а также о том, в каких случаях ребра ориентированного графа можно разметить таким образом, чтобы образованная диаграмма Мура соответствовала групповому или дефинитному автомату. Приводятся описания алгоритмов, осуществляющих такую разметку.

Результаты во второй части доклада относятся к классической задаче теории графов: определению хроматического числа и других свойств графа в зависимости его разложения на более “простые” подграфы. В докладе приводятся оценки хроматического числа графа в зависимости от его толщины, охвата и древесности.

Доклад может быть интересен широкому кругу специалистов в теории автоматов и теории графов.

26 сентября 2018 года

Компьютерное моделирование логических процессов

профессор Подколзин А. С.

Главным средством изучения логических процессов на сегодняшний день является их компьютерное моделирование. Доклад посвящен исследованию техники такого моделирования. Это исследование позволило поднять уровень обучения компьютерных решателей задач до пограничного слоя между теоремами и алгоритмами и вплотную приблизиться к анализу источников саморазвития решателей. В процессе обучения возникла версия решателя, позволяющая не только получать ответ, но и показывать ход рассуждений по шагам.

3 октября 2018 года

Жизнь после описания сложности задачи удовлетворения ограничениям

с.н.с. Жук Д. Н.

В 2017 году была описана сложность задачи удовлетворения ограничениям на конечном множестве в зависимости от языка ограничений, что являлось основной открытой проблемой в данной области на протяжении 20 лет.

В докладе будет рассмотрен как этот результат, так и некоторые вариации и обобщения, которые до сих пор остаются открытыми проблемами и к которым сейчас приковано основное внимание. В частности, будет рассмотрено обобщение, где помимо кванторов существования допускаются также кванторы всеобщности (Quantified CSP), задача удовлетворения ограничениям на бесконечном множестве, задача удовлетворения ограниченям с обещанием (Promise CSP) и некоторые другие.

17 октября 2018 года

Решётка всех клонов на трёхэлементном множестве, задаваемых бинарными предикатами

Моисеев С. В.

В 2016-ом году автором была описана решётка всех клонов трёхзначной логики, которые могут быть заданы как классы сохранения некоторого множества бинарных предикатов. Оказалось, что существует ровно 2,079,040 таких клонов. В докладе будет рассмотрен этот результат, а также множество других фактов, выявленных в ходе работы над основным результатом.

24 октября 2018 года

О графовой модели криптографических протоколов

доцент Миронов А. М.

Криптографические протоколы — это распределенные алгоритмы, предназначенные для обеспечения безопасной передачи информации в небезопасной среде. Они используются, например, в электронных платежах, электронных процедурах голосования, системах доступа к базам данных, и т.д. Учитывая большой финансовый и социальный ущерб в случае неправильной работы таких протоколов, необходимо использовать математические методы для обоснования их корректности и безопасности. В докладе была предсталена новая математическая модель таких протоколов, позволяющая описывать как сами протоколы, так и их свойства. Было показано, как на базе данной модели можно решать задачи верификации криптографических протоколов.

31 октября 2018 года

Обучение устройств с дискретным управлением

аспирант Голиков К. А.

Сегодня во многих аспектах нашей жизни появляются автономные роботы, автомобили без водителя, самодвижущиеся устройства, дроны доставки, интеллектуальные алгоритмы, которые, обладая некоторыми

когнитивными функциями, для успешного взаимодействия с людьми, объектами и средой, должны уметь уточнять свою модель окружения в зависимости от меняющихся условий. В докладе будет рассмотрен один из подходов обучения систем и адаптации обучения к меняющимся условиям среды на примере задачи управления роботами с дискретным управлениям.

7 ноября 2018 года

О языках, устойчивых относительно операций выпадения, вставки

м.н.с. Дергач П. С.

В первой части доклада рассматривается два оператора замыкания языков — оператор вставки и оператор выпадения. Для них доказывается, что только регулярные языки могут быть замкнутыми. Приводится критериальное описание замкнутых классов в терминах регулярных выражений. Излагаются результаты об автоматной сложности таких языков. Решается проблема описания базисов возникающих классов, описываются все предполные классы, приводятся решения проблем полноты и выразимости.

Вторая часть доклада носит анонсирующий характер и посвящена переносу этих результатов на другие операторы замыкания. В частности, рассматривается оператор замыкания, заменяющий одну букву на две соседние, и обратный к нему. Приводятся некоторые результаты о свойствах возникающих замкнутых классов.

14 ноября 2018 года

О полиномиальной полноте конечных квазигрупп

с.н.с. Галатенко А. В.

Конечной квазигруппой называется конечное множество Q с бинарной операцией f такой, что для любых a и b из Q уравнения $f(x,a)=b$ и $f(a,y)=b$ однозначно разрешимы. Таблица Кэли квазигрупповой операции представляет из себя латинский квадрат. Квазигруппа называется полиномиально полной, если система из функции f и всех констант из Q полна относительно операции суперпозиции.

Одним из активно изучаемых приложений конечных квазигрупп является построение криптографических примитивов. При этом полиномиальная полнота — одно из желательных свойств, обеспечивающих стойкость. В докладе планируется рассмотреть критерии и алгоритмы проверки полиномиальной полноты.

21 ноября 2018 года

Распознавание лиц

с.н.с. Мазуренко И. Л.

В последние годы автоматическое распознавание лиц получило широкое применение на практике: эта технология работает в системах безопасности, используется для идентификации личности в мобильных телефонах, банковских системах, на транспорте, активно применяется в индустрии развлечений. В докладе будет дан обзор современного состояния науки и техники в этой области компьютерного зрения, а также приведены результаты совместной разработки по распознаванию лиц кафедры МаТИС и Московского исследовательского центра компании Хуавей. В завершающей части доклада будет сделана попытка формулировки сложных нерешенных математических и инженерных задач в области глубокого машинного обучения.

5 декабря 2018 года

Обобщенная реализуемость для языка арифметики и логики предикатов

м.н.с. Коновалов А. Ю.

Понятие реализуемости было введено в 1945 г. американским математиком С. К. Клини. В докладе предполагается рассмотреть модификации этого понятия, связанные с заменой в определении реализуемости класса всех частично-рекурсивных функций на другие классы функций.

Обучение систем с дискретным управлением

Голиков К.А.

В докладе изложена работа по созданию алгоритма обучения системы с дискретным управлением действовать и достигать целей. Обучение происходит на основе проб и ошибок. Весь опыт системы сохраняется в Базе Данных. Оптимизация алгоритма производится по двум критериям: точность достижения поставленных целей и максимальное сокращение времени обучения. Сокращение времени обучения реализуется, главным образом, уменьшением количества пробных действий с помощью методов прогнозирования и интерполяции по опытным данным.

Ключевые слова: позиционирование, алгоритм обучения, робот, интерполяция, аппроксимация.

Пусть есть некоторый **Алгоритм** – обучаемый субъект, принимающий решения, который автономно исследует оптимальное поведение с помощью попыток и ошибок, взаимодействует с устройством для решения проблемы управления.

Пусть есть **Система** – это то, с чем может взаимодействовать алгоритм в процессе обучения. Предполагается, что система – это некоторое физическое устройство или виртуальная модель устройства (через систему можно опосредованно взаимодействовать с иными системами и окружающей средой).

Алгоритм работает в дискретном времени: каждый такт времени он может узнавать часть данных о состоянии системы, а также выбирать действие, которое будет производиться системой, на основе полученных данных и предположений: любые произведённые системой действия сохраняются в БД, на основе этих данных делаются прогнозы.

У системы есть **приводы**, обеспечивающие действия системы по своим внутренним неизвестным законам, зависящим от внутреннего состояния системы и изменяющихся во времени. У системы есть **входы**, алгоритм подаёт на них дискретные значения - указания приводам действовать. У системы есть **выходы**, с них алгоритм может считывать

вещественные значения, получать частичные данные о состоянии - *обратную связь*. У системы есть фиксированное **начальное положение**. Из этого начального положения алгоритм учится достигать *целей*. Цели характеризуются системой и задачей, которую система предназначена решать.

Мы изначально *отказываемся от возможности аналитически построить точную или приближённую физическую модель системы, полагаемся только на опыты и обратную связь*. Это принципиальное ограничение нашего исследования.

АВТОМАТНАЯ МОДЕЛЬ СИСТЕМЫ

Будем рассматривать **систему**, как чёрный ящик, который работает по тактам.

Входы – m штук (m-чётно) бинарных значений (1 или 0), означают включён ли привод в текущий такт или выключен

+ аппаратная кнопка $reset$ (1 или 0) - возвращение системы в начальное положение.

Выходы – y штук вещественных чисел, описывают известную часть состояния системы

- функция выходов $f(c_1, \ldots, c_m, reset) = (x_1, ..., x_y)$
- функция перехода $g(c_1, \ldots, c_m, reset, q_1, \ldots, q_i, ...)$ – состояние внутренних переменных, их количество и структура связей неизвестны.

Кнопка reset возвращает систему в известное начальное положение (абсолютно точно) из любого состояния.

Количество приводов чётно, потому что любое действие должно быть обратимо. В систему приводы добавляются по два антагониста, работающих в разных направлениях. *Известно, как нумеруются входы приводов, для каждого управления можно получить обратное.* Если произвести действие системой, подавая на входы C до некоторого конечного состояния, тогда, выполняя инверсное управление C', из этого конечного состояния можно вернуться в начальное положение почти точно. Всегда точно вернуться обратно (без reset) нельзя, потому что внутренние законы функционирования приводов системы зависят от состояния системы и времени - со временем они гладко изменяются в небольших пределах. Таким образом, от действия к действию конечное состояние системы может смещаться. Следовательно, запомненное в базе данных действие, успешно достигшее цели, на практике не всегда её достигает (корректирующие операции будут обсуждены в конце статьи).

Каждый такт на входы системы можно подавать значения включены или выключены конкретные приводы, т.е. усилие привода - одна из неизвестных внутренних переменных, напрямую управлять ей нельзя. Можно варьировать *время запуска и длительность работы каждого привода*. Т.к. в приводах и внутренних механизмах системы есть определённое трение и прочие лаги, то для того, чтобы активация входов привела к фактическому изменению состояния системы, приводы должны быть активны в течение какого-то времени. Одного такта активации недостаточно, чтобы привод развил силу, способную преодолеть величину трения, сдвинуть систему из одного состояния в другое. Единицы нужно группировать в последовательности, такие входы приводят к результатам.

Кроме того, самым дорогим ресурсом при обучении - является *время работы системы*. Чтобы производить как можно больше действий, нужно чтобы сами действия были бы короткими. Поэтому вводится n – *максимальное число тактов времени действия системы*. Назовём **эпизодом** обучения изменение состояния системы от известного начального положения в некоторое *конечное положение* с последующим возвратом в начальное положение (обычно с помощью кнопки reset) за время меньшее или равное n.

Определим понятие **управление** - это матрица, у которой число строк m(число приводов) и число столбцов n(максимальная длительность эксперимента), эта матрица определяется последовательностями времени работы приводов в течение отдельного эпизода обучения. При этом наложим дополнительное ограничение на управление - общее число последовательностей единиц во всех строках матрицы должно быть не больше k, где $k << n$. В таком случае движение получится не очень длинное, более прямолинейное – сложнее сгенерировать управление, ведущее систему к цели окольными путями.

Решения задачи будем искать в виде **управлений**.

ИНТЕРПРЕТАЦИЯ ДЛЯ КОНКРЕТНОЙ ЗАДАЧИ

Обучение алгоритма отрабатывается в *виртуальной среде* для задачи **позиционирования разных роботов**. Применяя алгоритм для разных систем показываем широту возможностей алгоритма и достаточность описанных выше ограничений для успешного решения задачи.

В рамках задачи позиционирования количество приводов m, для которого исследуем поведение системы маленькое от 4 до 10. k для управлений - в пределах 4-8 одновременно активных приводов, а число тактов

эпизода $n = 10000$. Выходов у системы ровно 4, понимаются как координаты *базовой точки* системы (x,y) и вектор *моментальной скорости* этой точки $v = (v_x, v_y)$.

Цели интерпретируются как точки, в которые нужно привести базовую точку системы с заданной наперёд *точностью* быстрейшим образом. А *конечным положением* для эпизода является обязательно статичное положение системы - полная остановка базовой точки. Система пришла в целевую точку (выполнила задание), если её базовая точка находится в малой окрестности целевой точки (в соответствии с точностью), и скорость базовой позиции $v = (0., 0.)$. Цели выбираются произвольным образом, они больше нужны для обучения, чем для работы системы. Пробуя попадать в конкретные цели, система в итоге учится попадать в любые точки. При этом, чем больше вокруг новой точки, в которую хочется спозиционироваться, располагается выученных конечных положений, тем точнее действие будет произведено обученной системой.

Для уменьшения времени обучения целевыми точками лучше покрывать область, в которой в дальнейшем будет производиться полезная работа. Например, если система - это робот манипулятор, который должен научиться закручивать шурупы в отверстия на определённых координатах. Нужно, во-первых, расположить робота так, чтобы все отверстия лежали в рабочем пространстве манипулятора, во-вторых, закрепить банку с шурупами в начальном позиции робота, в-третьих, целевые точки задать так, чтобы они покрывали область детали с отверстиями, в которые нужно вкручивать шурупы.

ПРИМЕРЫ СИСТЕМ

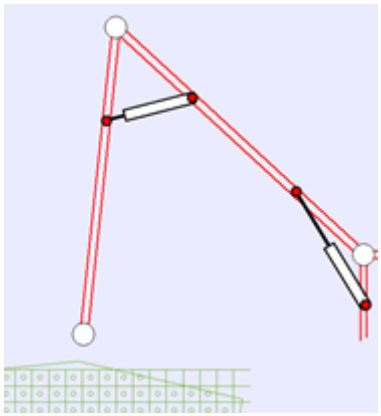


Рис. 1.

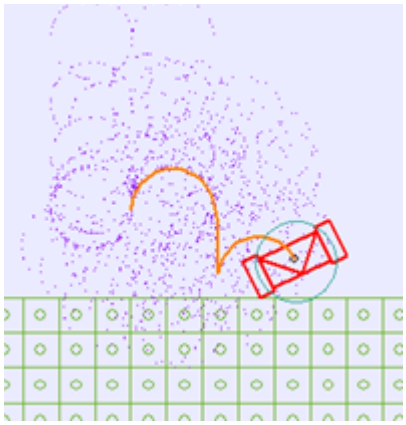


Рис. 2.

1. **Манипулятор** с 2-3мя вращательными и призматическими сочленениями (рис.1). Базовая точка – точка центра захвата. Привод – действие: либо выдвигающее, либо сдвигающее пневмо-цилиндр сочленения.

2. **Вездеход** с 2мя гусеницами (рис.2). Базовая точка – центр робота. Привод - вращение гусеницы либо вперёд, либо назад.

СЛОЖНОСТИ В РЕШЕНИИ ЗАДАЧИ

Подчеркнём отдельно те сложности, которые необходимо преодолеть при решении задачи в описанных выше ограничениях:

1. За минимальное время произвести системой *ценные действия*, отражающие возможности и принципы функционирования неизученной системы. Получить управления, хорошо подходящие для интерполяции и предсказаний поведения системы. В рамках задачи позиционирования: получить равномерную плотную решётку конечных положений покрывающую целевые точки.

2. Получить метод интерполяции по сохранённым в БД управлениям, достаточно точный, а также метод выбора действий, которые ценно будет произвести для уточнения алгоритма интерполяции, чтобы минимизировать возможность промахов.

3. В условиях отсутствия времени на переобучение при изменяющихся законах функционирования приводов системы построить алгоритм адаптации к этим изменениям.

1. РАВНОМЕРНАЯ РЕШЁТКА КОНЕЧНЫХ ПОЛОЖЕНИЙ

Случайная генерация управлений не даёт выборки действий, по которым может получиться хорошая интерполяция. Получается много конечных положений системы рядом с начальным положением и совсем мало в остальном рабочем пространстве робота. Кроме того, эти сгенерированные управления – описывают неоптимальные траектории (рис.3), сильно отличаются по содержащимся последовательностям активных приводов, по ним нельзя построить достаточно точную интерполяцию за адекватное время обучения.

Написан двухэтапный алгоритм построения равномерной решётки конечных положений в нужной области, он обеспечивает возможность сравнимости и усреднения управлений. На первом этапе варьированием числа активных приводов и их длительностей получается *разряжённая решётка действий системы, конечные положения которых покрывают всё рабочее пространство робота.* Это даёт понимание того, на что в принципе способна система, где располагаются *цели в координатах*

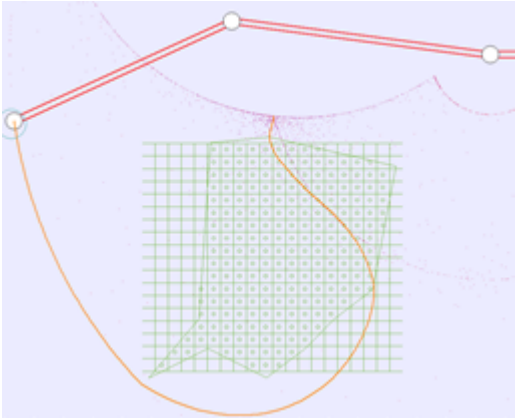


Рис. 3.

управлений. Второй этап - построение плотной по возможности равномерной решётки, покрывающей все целевые точки. Идея в следующем: длительности работы мускулов увеличиваются с некоторым шагом, одновременно производятся смещения моментов времени старта последовательностей работы приводов относительно друг друга в создаваемом управлении. Подбор управлений - это подбор значений аргументов неизвестной функции, чтобы выдерживать равные дистанции между её значениями. "Адаптивный"подбор изменений управлений осуществляется методами Монте-Карло и МНК на основе имеющиеся в БД управлений выполненных действий.

2. ИНТЕРПОЛЯЦИЯ ПО ПРОИЗВЕДЁННЫМ ДЕЙСТВИЯМ

После построения плотной решётки конечных положений. Приближение к целевым или желаемым точкам осуществляется методом Стохастического Градиентного спуска. Не всякие два управления подходят для градиентного спуска. Если управления очень непохожи друг на друга, то среднее арифметическое не существует – всё разное. Если есть среднее арифметическое, значит разные лишь длительности работы одних и тех же приводов с сохранением их порядка активации.

На рис.4 показана ситуация после окончания работы алгоритма получения плотной решётки. На нём видно два кластера управлений. Внутри кластеров получение промежуточных управлений возможно, а управления из разных кластеров между собой усреднить нельзя - результат усреднения не приведёт в конечное положение между выбранными.

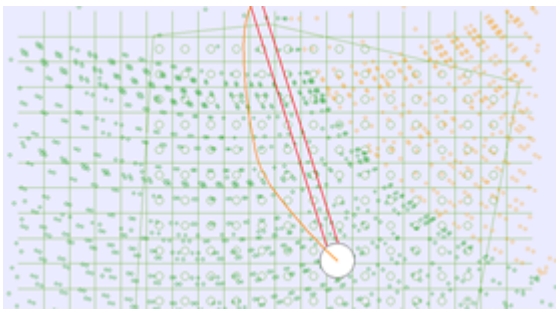


Рис. 4.

Интерполяции строятся локальные по ближним управлениям для ближних конечных положений, попадающие в один кластер. В этом случае построение интерполяции не очень затратная по вычислительной мощности операция, а точность приближения у неё довольно высокая.

3. АДАПТАЦИЯ К ИЗМЕНЕНИЯМ В ДЕЙСВИИ ПРИВОДОВ

В реальном мире при функционировании системы в среде могут возникать

• *системные изменения* – изменения в приводах происходят постепенно и медленно, вносят ограниченные смещения, действуют длительно в течение всего эпизода переобучения (атмосферное давление, износ подшипников и пр.), к ним нужно *уметь приспособляться*,

• *мгновенные изменения* – сильное смещение, действует один такт времени, неповторяется (порыв ветра, задевание недопустимого объекта), нужно *уметь отфильтровывать этот шум*.

Если бы условия менялись сильно и непредсказуемым образом, действовали бы длительно, то, очевидно, прогностические методы к такому окружению были бы неприменимы. Законы движения и изменения в приводах задаются разностными дифференциальными уравнениями, проводя эксперименты с переобучением управления системой важно выяснить границы применимости адаптации для разных видов уравнений.

Параметры внутреннего состояния системы делятся на три вида:

• задаваемые (вес груза, угол установки системы, . . .),

• наблюдаемые (температура помещения, износ подшипников, . . .),

• ненаблюдаемые (изменение вязкости рабочего тела пневматики, трение, влажность воздуха, плотность среды, . . .)

Внутренние параметры переходятся в тот или иной вид, исходя из технической возможности задать параметры, предусмотреть и установить необходимые датчики на физическое устройство и пр.

Итак, *память обширная*, предоставляет возможности по построению адекватных предсказаний, *но её данные со временем устаревают и перестают верно отображать обстановку вещей*. Например, у манипулятора, из-за засорения локтевого подшипника, появляется неодинаковое для любой точки рабочей области смещение влево на несколько миллиметров. Подавая управление, ранее точно приводившее захват в цель, сейчас приводит его в точку левее цели.

ОБУЧЕНИЕ АДАПТАЦИИ

Для того, чтобы переобучаться времени нет, уточнение действий производятся "на лету". После успешной фазы обучения происходит эксплуатация обученной системы. База Данных постоянно дополняется. Так как записи действий в БД сохраняются навсегда, важно не запутаться в одинаковых действиях со смещёнными конечными положениями. Для этого выделяется первое *эталонное действие с определённым управлением*, при повторениях в указанную запись добавляются *фактические смещения базовой точки* системы.

Процедура «Удержание на траектории»

Во время повторения запомненной траектории в каждый момент времени производится слежение за величиной и направлением отклонения от эталонной траектории, в управление подмешивается сначала выбранная наугад инверсная часть, с каждым тактом она подбирается точнее, чтобы компенсировать выявленное смещение.

Данная процедура позволяет уменьшить величину конечного отклонения насколько это возможно для необученного алгоритма с подкреплением. Для того чтобы полностью нивелировать смещение, необходимо использовать статистические методы.

Процедура «Изучение смещений»

1. Разделить область позиционирования (конечных положений) на несколько небольших регионов.

2. Для каждого региона регистрировать направление отклонений, коллекционировать их, похожие собирать в кластеры, задавать вес больше для актуальных смещений.

3. По данным кластера построить обратное преобразование: по смещению найти инверсное управление, которое нужно подмешать к эталонному управлению, чтобы вернуться на траекторию и попасть в итоге в цель.

4. Для каждого исполняемого действия, как можно раньше выявлять самый похожий кластер.

Функцию аппроксимации полного эпизода - возвращающую управление по начальной и конечной позициям системы - обучить дольше и затратнее[4], чем несколько простых функций аппроксимаций, каждая из которых решает простую подзадачу, а все работают вместе, дополняя друг друга.

Список литературы

[1] Яблонский С.В. Введение в дискретную математику. — М.: Высшая школа, 2006.

[2] Саттон Р.С., Барто Э.Г. Обучение с подкреплением — 2-е изд. — М.:БИНОМ.Лаборатория знаний, 2014.

[3] Богачёв К.Ю. Практикум на ЭВМ. Методы приближения функций — М.: Мех.-мат. МГУ, 1998.

[4] Окуловский Ю.С. Интеллектуальные алгоритмы калибровки робототехнических систем — [Электронный ресурс] // Екатеринбург: УрГУ, 2010. URL: http://elar.urfu.ru/bitstream/10995/21944/1/Okulovskii_GK_P1047.pdf (дата обращения: 13.11.2018).

Learning systems with descrete control

Golikov K.A.

The report outlines the work to create a discrete-control system learning algorithm for acting and achieving goals. Learning is based on trials and misses. The entire experience of the system is stored in the Database. The algorithm is optimized by two criteria: the accuracy of achieving the goals and the maximum reduction in training time. The reduction in training time is implemented mainly by reducing the number of trials using prediction methods and interpolation by experimental data.

Keywords: positioning, learning algorithm, robot, interpolation, approximation.

О свойствах языков, устойчивых относительно операций выпадения, вставки

Дергач П.С., Кудрявцев В.Б.

В статье изучаются операции выпадения/вставки, продвижением которых занимался В. И. Левенштейн. Вводятся операторы замыкания относительно этих операций. Для оператора вставки доказывается существование, конечность и единственность базиса в замкнутых классах, а для оператора выпадения — несуществование для бесконечного класса и существование, конечность и единственность — для конечного. Исследуется автоматная сложность замкнутых классов. Решаются проблемы полноты, предполноты и выразимости.

Ключевые слова: операции выпадения и вставки; замкнутый класс; регулярный язык; базис; автоматная сложность; проблемы полноты/предполноты/выразимости.

Введение

Данная статья является продолжением статьи [1] , которая, в свою очередь, основана на статье [2]. В ней изучаются языки, устойчивые относительно операций выпадения, вставки — по отдельности для каждой из операций. В [1] было показано, что такие языки регулярны, был найден их канонический вид. В этой статье изучаются и успешно решаются следующие три проблемы:

- существование, нахождение и количество базисов у возникающих замкнутых классов;
- разбиение замкнутых классов в конечное объединение регулярных множеств, имеющих линейную от длины выражения автоматную сложность;

- полнота, предполнота и выразимость для классов регулярных языков относительно операторов выпадения, вставки.

Основные определения и результаты

Напомним основные определения.

Определение 1. *Множество конечных слов в алфавите A обозначаем через A^*. Пустое слово Λ по умолчанию тоже лежит в A^*.*

Определение 2. *Множество натуральных чисел обозначаем через $\mathbb{N}$, а множество целых неотрицательных чисел — через $\mathbb{N}_0$.*

Определение 3. *Множество всех регулярных языков в алфавите A обозначаем через $R(A)$.*

Замечание 1. *Понятие регулярного языка подробно изложено в [3].*

Определение 4. *Пусть $\alpha \in A^*$, $P \subseteq A^*$, $k \in \mathbb{N}$, $* \in \{in, out\}$. Тогда*

$$[\alpha]_*^0 := \{\alpha\},$$

$$[\alpha]_{in}^k := \{\alpha_1 a \alpha_2 \mid \alpha_1\alpha_2 \in [\alpha]_{in}^{k-1}, a \in A, \alpha_1, \alpha_2 \in A^*\},$$

$$[\alpha]_{out}^k := \{\alpha_1\alpha_2 \mid \alpha_1 a \alpha_2 \in [\alpha]_{out}^{k-1}, a \in A, \alpha_1, \alpha_2 \in A^*\},$$

$$[\alpha]_* := \bigcup_{k=0}^{\infty} [\alpha]_*^k,$$

$$[P]_* := \bigcup_{\alpha \in P} [\alpha]_*.$$

Замечание 2. *Возникающий при этом оператор in будем далее называть оператором вставки, а оператор out — оператором выпадения.*

Определение 5. *Пусть $\alpha_1, \alpha_2 \in A^*$. Если $\alpha_2 \in [\alpha_1]_{in}$ или $\alpha_1 \in [\alpha_2]_{in}$, то говорим, что эти слова сравнимы. Если же это не так, то говорим, что эти слова несравнимы.*

Определение 6. *Пусть $P \subseteq A^*$. Говорим, что $L \subseteq P$ — базис относительно оператора $* \in \{in, out\}$ в P, если*

- $P = [L]_*$,
- *для любого* $L_1 \subset L$ *верно* $P \neq [L_1]_*$.

Определение 7. *При фиксированном входном алфавите* A *и выходном алфавите* $B = \{0, 1\}$ *обозначаем через* $K(A, B)$ *множество абстрактных конечных инициальных автоматов в этих алфавитах. Для* $n \in \mathbb{N}$ *через* $Avt(A, n)$ *обозначаем множество событий* $P \subseteq A^*$, *представимых автоматами из* $K(A, B)$ *с не более чем* n *состояниями.*

Замечание 3. *Зависимость* $Avt(A, n)$ *от* A *далее, для простоты, будем опускать и писать просто* $Avt(n)$.

Замечание 4. *Понятия автомата и представимости события автоматом подробно изложены в [3].*

Определение 8. *Пусть* $P \subseteq A^*$. *Говорим, что класс* P — *полный относительно оператора* $* \in \{in, out\}$ *в* P, *если*

$$[P]_* = A^*.$$

Проблемой полноты относительно одного из операторов $* \in \{in, out\}$ *называем проблему определения по произвольному* $P \in R(A)$ *полноты (или не полноты) этого класса относительно оператора *.*

Определение 9. *Пусть* $P \subset A^*$. *Говорим, что класс* P — *предполный относительно оператора* $* \in \{in, out\}$ *в* P, *если*

- $[P]_* \neq A^*$,
- *для любого* $P \subset P_1 \subseteq A^*$ *верно* $[P_1]_* = A^*$.

Проблемой предполноты относительно одного из операторов $* \in \{in, out\}$ *называем проблему определения по произвольному* $P \in R(A)$ *предполноты (или не предполноты) этого класса относительно оператора *.*

Определение 10. *Проблемой выразимости относительного одного из операторов* $* \in \{in, out\}$ *называем проблему определения по произвольным* $P_1, P_2 \in R(A)$ *выполнимости свойства*

$$[P_1]_* \subseteq P_2.$$

Утверждение 1. *Пусть $P \subseteq A^*$, $P = [P]_{in}$. Тогда в P существует базис относительно оператора вставки, он конечен и единственен.*

Утверждение 2. *Пусть $P \subseteq A^*$, $P = [P]_{out}$. Если P бесконечно, то в P не существует базиса относительно оператора выпадения. Если же P конечно, то такой базис существует, конечен и единственен.*

Утверждение 3. *Для любого $\alpha \in A^*$ верно*

$$[\alpha]_{in} \in Avt(t+1),$$

где t — длина слова α.

Утверждение 4. *Для любых $s \in \mathbb{N}_0$, $\alpha_1, \ldots, \alpha_{s+1} \in A^*$ и непустых $A_1, \ldots, A_s \subseteq A$ верно*

$$[\alpha_1]_{out} \cdot (A_1)^* \cdot [\alpha_2]_{out} \ldots (A_s)^* \cdot [\alpha_{s+1}]_{out} \in Avt(t_1 + \ldots + t_{s+1} + 2),$$

где t_i — длины слов α_i.

Утверждение 5. *Проблема полноты относительно операторов выпадения, вставки алгоритмически разрешима.*

Утверждение 6. *Проблема предполноты относительно операторов выпадения, вставки алгоритмически разрешима.*

Утверждение 7. *Проблема выразимости относительно операторов выпадения, вставки алгоритмически разрешима.*

Доказательство вспомогательных утверждений

Лемма 1. *Пусть $P \subseteq A^*$. Тогда*

$$P = [P]_{in} \Longleftrightarrow P = \bigcup_{\alpha \in T} [\alpha]_{in},$$

где T — произвольное конечное множество слов в алфавите A.

Доказательство.

Доказательство этого утверждения приведено в [1].

Лемма 2. *Пусть* $P \subseteq A^*$. *Тогда*

$$P = [P]_{out} \iff P = \bigcup_{i=1}^{k} [\alpha_{i,1}]_{out} \cdot (A_{i,1})^* \cdot [\alpha_{i,2}]_{out} \dots (A_{i,s(i)})^* \cdot [\alpha_{i,s(i)+1}]_{out},$$

где $k \in \mathbb{N}$, $s(i) \in \mathbb{N}_0$, $\alpha_{i,j} \in A^*$, $A_{i,j} \subset A$.

Доказательство.

Доказательство этого утверждения приведено в [1].

Лемма 3. *Для любых* $P_1, P_2 \subseteq A^*$ *и* $* \in \{in, out\}$ *верно, что*

- $[P_1 \cdot P_2]_* = [P_1]_* \cdot [P_2]_*$,
- $[P_1 \cup P_2]_* = [P_1]_* \cup [P_2]_*$.

Доказательство.

Первая часть утверждения доказана в [1]. Докажем вторую часть.

Если $\alpha \in [P_1 \cup P_2]_*$, то $\alpha \in [\beta]_*$ для некоторого $\beta \in P_1 \cup P_2$. Пусть это, без ограничения общности, P_1. Тогда $\alpha \in [P_1]_* \subseteq [P_1]_* \cup [P_2]_*$.

В другую сторону, пусть $\alpha \in [P_1]_* \cup [P_2]_*$. Тогда, без ограничения общности, $\alpha \in [P_1]_*$, то есть $\alpha \in [\beta]_*$ для некоторого $\beta \in P_1 \subseteq P_1 \cup P_2$. Значит $\alpha \in [P_1 \cup P_2]_*$.

■

Лемма 4. *Для любого* $P \subseteq A^*$ *верно, что*

- $[P^*]_{in} = A^*$,
- $[P^*]_{out} = A_1^*$, *для некоторого* $A_1 \subseteq A$.

Доказательство.

Доказательство этого утверждения приведено в [1].

Лемма 5. *Пусть* $P \subseteq A^*$ *предполно относительно какого-то из операторов* $* \in \{in, out\}$. *Тогда оно получается из множества* A^* *удалением одного слова.*

Доказательство.

В самом деле, пусть найдется предполное множество, отличающееся от A^* хотя бы на два слова. Назовем их α_1 и α_2. Из второго свойства определения 9 следует, что тогда $\alpha_1 \in [\alpha_2]_*$ и $\alpha_2 \in [\alpha_1]_*$. Но это означает, что $\alpha_1 = \alpha_2$.

■

Доказательство основных утверждений

Утверждение 1. *Пусть $P \subseteq A^*$, $P = [P]_{in}$. Тогда в P существует базис относительно оператора вставки, он конечен и единственен.*

Доказательство.

Из леммы 1 мы знаем, что

$$P = \bigcup_{\alpha \in T} [\alpha]_{in}, \tag{1}$$

где T — произвольное конечное множество слов в алфавите A. Можно считать, что слова из T попарно несравнимы, так как в противном случае большее по длине сравнимое слово можно было бы просто выкинуть из T, сохранив при этом равенство (1). Это то множество, очевидно, и будет базисом в P относительно оператора вставки. В самом деле, первое свойство из определения 6 выполнено в силу (1), а второе — в силу того, что слова из T попарно не сравнимы и ни одно из них не пораждается замыканием остальных.

Осталось доказать единственность базиса. Но легко заметить, что множество T — это множество минимальных по вложению относительно вставки слов из P. Значит каждое слово из T должно быть в произвольном базисе, так как его нельзя получить замыканием других слов из P. Утверждение доказано.

■

Утверждение 2. *Пусть $P \subseteq A^*$, $P = [P]_{out}$. Если P бесконечно, то в P не существует базиса относительно оператора выпадения. Если же P конечно, то такой базис существует, конечен и единственен.*

Доказательство.

Из леммы 2 мы знаем, что

$$P = \bigcup_{i=1}^{k} [\alpha_{i,1}]_{out} \cdot (A_{i,1})^* \cdot [\alpha_{i,2}]_{out} \ldots (A_{i,s(i)})^* \cdot [\alpha_{i,s(i)+1}]_{out}, \tag{2}$$

где $k \in \mathbb{N}$, $s(i) \in \mathbb{N}_0$, $\alpha_{i,j} \in A^*$, $A_{i,j} \subset A$.

Если P бесконечно, то хотя бы одно из множеств $A_{i,j}$ в (2) непусто. Очевидно, что если бы базис в P существовал (назовем его L), то он был

бы бесконечен, так как замыкание относительно выпадения переводит конечные множества в конечные. Поэтому в L обязательно было бы слово α, лежащее в, без ограничения общности, множестве

$$[\alpha_{1,1}]_{out} \cdot (A_{1,1})^* \cdot [\alpha_{1,2}]_{out} \dots (A_{1,s(1)})^* \cdot [\alpha_{i,s(1)+1}]_{out}, \tag{3}$$

где $s(1) > 0$. Но тогда, засчет итерации непустого множества $A_{i,j}$ в (3), получаем, что в P существует слово $\beta \neq \alpha$, для которого выполнено $\alpha \in [\beta]_{out}$. Это означает (в силу базисности L), что существует $\gamma \in L$ такое, что $\beta \in [\gamma]_{out}$. Но тогда $\alpha \in [\gamma]_{out}$, а это противоречит базисности L, так как $\alpha \neq \gamma$ и нарушено второе свойство из определения 6.

Допустим теперь, что P конечно. Тогда его базисом будет множество слов из P, являющихся в нем максимальными по вложению. Очевидно, это множество конечно, так как конечно само P. Кроме того, любой базис в P обязан содержать это множество. Значит такой базис единственен (в силу второго свойства из определения 6). Утверждение доказано.

■

Утверждение 3. *Для любого $\alpha \in A^*$ верно*

$$[\alpha]_{in} \in Avt(t+1),$$

где t — длина слова α.

Доказательство.

Пусть $\alpha = a_{i_1} \dots a_{i_t}$. Очевидно, что

$$[\alpha]_{in} = A^* \cdot a_{i_1} \cdot A^* \cdot a_{i_2} \dots A^* \cdot a_{i_t} \cdot A^*. \tag{4}$$

Опишем автомат, задающий это множество. В первом состоянии он всегда переходит в себя, кроме перехода во второе состояние по букве a_{i_1}. Во втором состоянии автомат переходит в себя, кроме перехода в третье состояние по букве a_{i_2}. И так далее. В предпоследнем состоянии он переходит в себя, кроме перехода в последнее состояние по букве a_{i_t}. Наконец, в последнем состоянии он переходит всегда в себя. Автомат принимает те и только те, слова которые приводят его в последнее состояние. Очевидно, что этот автомат искомый и в нем $t+1$ состояний. Неформально говоря, автомат следит за тем, какой максимальный префикс слова α уже встретился как подслово во входном слове. Если этот префикс — все слово, то автомат его принимает. В противном случае, слово автоматом не принимается. Утверждение доказано.

■

Следствие 1. *Пусть $P \subseteq A^*$, $P = [P]_{in}$. В силу леммы 1 получаем, что P представимо в виде конечного объединения множеств вида (4), каждое из которых имеет автоматную сложность, линейно зависящую от своей длины t.*

Утверждение 4. *Для любых $s \in \mathbb{N}_0$, $\alpha_1, \ldots, \alpha_{s+1} \in A^*$ и непустых $A_1, \ldots, A_s \subseteq A$ верно*

$$[\alpha_1]_{out} \cdot (A_1)^* \cdot [\alpha_2]_{out} \ldots (A_s)^* \cdot [\alpha_{s+1}]_{out} \in Avt(t_1 + \ldots + t_{s+1} + 2), \quad (5)$$

где t_i — длины слов α_i.

Доказательство.

Ограничимся неформальным описанием функционирования автомата. Составим слово $\alpha_1\alpha_2 \ldots \alpha_{s+1}$. Для каждой из позиций в этом слове у автомата будет свое состояние. Таких позиций ровно $t_1 + \ldots t_{s+1} + 1$. Кроме того, у автомата будет одно дополнительное последнее состояние, играющее роль тупиковой ловушки. Находясь в нетупиковом основном состоянии q и получив на вход букву a автомат определяет, в какой самой левой из позиций выражения (5) он все еще может оказаться, стартуя из соответствующей позиции q (при этом всегда вставая слева от итерации, если находится на границе между словами) и двигаясь слева направо. При движении автомат пропускает все буквы слов α_i, отличные от a и пропускает итерацию A_i, если $a \notin A_i$. Если он встретил букву a в каком-то α_i, то он делает еще один шаг вправо и останавливается за найденной буквой. Если же он встретил a в итерации, то он остается слева от нее. Если автомат прошел целиком все выражение (5) и не встретил букву a, то он переходит в тупиковое состояние. Иначе новое состояние автомата будет соответствовать найденной позиции, в которой он остановился. Из тупикового состояния автомат уже никогда не выходит. Автомат будет принимать те и только те слова, которые еще не загнали его в тупиковое состояние. Очевидно, что этот автомат принимает множество (5) и у него $t_1 + \ldots t_{s+1} + 2$ состояний. Утверждение доказано.

■

Следствие 2. *Пусть $P \subseteq A^*$, $P = [P]_{out}$. В силу леммы 2 получаем, что P представимо в виде конечного объединения множеств вида (5), каждое из которых имеет автоматную сложность, линейно зависящую от своей длины $t = t_1 + \ldots + t_{s+1}$.*

Утверждение 5. *Проблема полноты относительно операторов выпадения, вставки алгоритмически разрешима.*

Доказательство.

Заметим, что множество $P \subseteq A^*$ является полным относительно вставки тогда и только тогда, когда содержит пустое слово. В самом деле, пустое слово из других слов получить вставкой нельзя, а из пустого слова, в свою очередь, можно получить вставкой любое слово. Ясно, что проблема принадлежности пустого слова регулярному языку алгоритмически разрешима.

Для доказательства части утверждения относительно оператора выпадения вспомним, что любое регулярное множество можно эффективно представить в виде

$$P = \bigcup_{i=1}^{k} \alpha_{i,1} \cdot (P_{i,1})^* \cdot \alpha_{i,2} \cdot (P_{i,2})^* \dots (P_{i,s(i)})^* \cdot \alpha_{i,s(i)+1}, \tag{6}$$

где $k \in \mathbb{N}$, $s(i) \in \mathbb{N}_0$, $\alpha_{i,j} \in A^*$, $P_{i,j} \in R(A)$. Заметим, что в силу лемм 3 и 4 множество $[P]_{out}$ можно теперь эффективно представить в виде

$$\bigcup_{i=1}^{k} [\alpha_{i,1}]_{out} \cdot (A_{i,1})^* \cdot [\alpha_{i,2}]_{out} \dots (A_{i,s(i)})^* \cdot [\alpha_{i,s(i)+1}]_{out}, \tag{7}$$

где $k \in \mathbb{N}$, $s(i) \in \mathbb{N}_0$, $\alpha_{i,j} \in A^*$, $A_{i,j} \subset A$. В самом деле, $A_{i,j}$ здесь — это просто буквы, встречающиеся в записи выражения для $P_{i,j}$. Осталось заметить, что множество (7) равно A^* если и только если хотя бы одно из $A_{i,j}$ в нем равно A. В одну сторону этот факт очевиден, а в другую сторону доказывается построением длинного слова, заведомо не лежащего ни в одном из конечных объединений множеств выражения (5). Попросту говоря, мы идем по выражениям для этих множеств слева направо и подбираем буквы входного слова, чтобы алгоритм движения автомата, описанный в доказательстве предыдущего утверждения, обязательно сошел с каждого из них в тупиковое состояние. Это можно сделать, так как всегда можно правильно выбрать очередную букву, чтобы перешагнуть как через конечные позиции в $\alpha_{i,j}$, так и (в случае необходимости) через итерацию какого-то из $A_{i,j}$, ведь все они не равны A. Закончив с одним выражением из (5) мы переходим к следующему, приписывая новые буквы справа от уже найденных. Таким образом, найден эффективный критерий проверки регулярного множества на полноту относительно операции выпадения. Утверждение доказано.

■

Утверждение 6. *Проблема предполноты относительно операторов выпадения, вставки алгоритмически разрешима.*

Доказательство.

В силу леммы 5 класс может быть предполным относительно вставки или выпадения только если он получен из A^* удалением одного слова. Но из пустого слова вставкой можно получить любое другое. Значит предполным относительно оператора вставки будет единственное множество $A \setminus \{\Lambda\}$. И можно эффективно проверить, равны ли два регулярных множества. Относительно оператора выпадения нужно заметить, что какое бы мы слово не удалили из A^*, оставшееся множество при замыкании относительно оператора выпадения даст все A^*, то есть такие множества будут полными, а не предполными. Значит предполных классов относительно операции выпадения не существует. Утверждение доказано.

■

Утверждение 7. *Проблема выразимости относительно операторов выпадения, вставки алгоритмически разрешима.*

Доказательство.

Мы должны научиться эффективно проверять верность равенства

$$[P_1]_* \subseteq P_2$$

для произвольных $P_1, P_2 \in R(A)$ и $* \in \{in, out\}$. Для этого воспользуемся той же идеей, что и в доказательстве утверждения 5. Получив представление P_1 в виде (6) и используя леммы 3 и 4 можно эффективно получить представление множества $[P_1]_{out}$ в виде (6) и множества $[P_1]_{in}$ в виде (1). Далее остается только прверить на вложение пару регулярных множеств. Утверждение доказано.

■

Список литературы

[1] П. С. Дергач. *О языках, устойчивых относительно операций выпадения, вставки.* Интеллектуальные системы, 2018. Т.22, вып. 2, М., Сс. 39-52.

[2] В. И. Левенштейн. *О Двоичные коды с исправлением выпадений и вставок символа 1.* Пробл. передачи информ., 1965. Т.1, вып. 1, М., Сс. 12-25.

[3] В. Б. Кудрявцев, С. В. Алешин, А. С. Подколзин. *Введение в теорию автоматов.* Издательство "Наука", М., 1985.

[4] П. С. Дергач. *О каноническом регулярном представлении S-тонких языков.* Интеллектуальные системы, 2014. Т.18, вып. 1, М., Сс. 211-242.системы, 2014. Т.18, вып. 1, М., Сс. 211-242.

[5] П. С. Дергач. *О проблеме вложения допустимых классов.* Интеллектуальные системы, 2015. Т.19, вып. 2, М., Сс. 143-174.

[6] П. С. Дергач, Э. С. Айрапетов. *О прогрессивном разбиении некоторых подмножеств натурального ряда.* Интеллектуальные системы, 2015. Т.19, вып. 3, М., Сс. 79-86.

[7] П. С. Дергач. *О двух размерностях спектров тонких языков.* Интеллектуальные системы, 2015. Т.19, вып. 3, М., Сс. 155-174.

[8] П. С. Дергач, Э. С. Айрапетов. *О прогрессивном разбиении последовательности натуральных чисел, имеющей пропуск длины 2.* Интеллектуальные системы, 2016. Т.20, вып. 2, М., Сс. 67-86.

[9] П. С. Дергач. *О проблеме проверки однозначности алфавитного декодирования в классе регулярных языков с полиномиальной функцией роста.* Интеллектуальные системы, 2016. Т.20, вып. 2, М., Сс. 147-202.

[10] П. С. Дергач, Е. Д. Данилевская. *О покрытиях и разбиениях натуральных чисел, имеющих два последовательных пропуска длины 1.* Интеллектуальные системы, 2017. Т.21, вып. 1, М., Сс.192-237.

[11] П. С. Дергач. *О структуре вложения прогрессивных множеств сложности два.* Интеллектуальные системы, 2017. Т.21, вып. 2, М., Сс.117-162.

[12] П. С. Дергач, Ж. И. Раджабов. *О длине минимальной алфавитной склейки для класса линейных регулярных языков.* Интеллектуальные системы, 2017. Т.21, вып. 3, М., Сс.120-130.

[13] Д. Е. Александров. *Эффективные методы реализации проверки содержания сетевых пакетов регулярными выражениями.* Интеллектуальные системы, 2014. Т.18, вып. 1, М., Сс. 37-60.

[14] Д. Н. Бабин. *Частотные регулярные языки.* Интеллектуальные системы, 2014. Т.18, вып. 1, М., Сс. 205-210.

[15] Д. Е. Александров. *Об оценках автоматной сложности распознавания классов регулярных языков.* Интеллектуальные системы, 2014. Т.18, вып. 4, М., Сс. 161-190.

[16] В. М. Дементьев. *О звездной высоте регулярного языка и цикли-ческой сложности минимального автомата.* Интеллектуальные системы, 2014. Т.18, вып. 4, М., Сс. 215-222.

[17] И. Е. Иванов. *О сохранении периодических последовательностей автоматами с магазинной памятью с однобуквенным магазином.* Интеллектуальные системы, 2015. Т.19, вып. 1, М., Сс. 145-160.

[18] А. А. Петюшко. *О контекстно-свободных биграмных языках.* Интеллектуальные системы, 2015. Т.19, вып. 2, М., Сс. 187-208.

[19] И. Е. Иванов. *Нижняя оценка на максимальную длину периода выходной последовательности автономного автомата с магазинной памятью.* Интеллектуальные системы, 2015. Т.19, вып. 3, М., Сс. 175-194.

[20] В. А. Орлов. *О конечных автоматах с максимальной степенью различимости состояний.* Интеллектуальные системы, 2016. Т.20, вып. 1, М., Сс. 213-222.

[21] А. М. Миронов. *Основные понятия теории вероятностных автоматов.* Интеллектуальные системы, 2016. Т.20, вып. 2, М., Сс. 283-330.

[22] А. А. Петюшко, Д. Н. Бабин. *Классификация Хомского для матриц биграммных языков.* Интеллектуальные системы, 2016. Т.20, вып. 2, М., Сс. 331-336.

[23] С. Б. Родин. *О связи линейно реализуемых автоматов и автоматов с максимальной вариативностью относительно кодирования состояний.* Интеллектуальные системы, 2016. Т.20, вып. 2, М., Сс. 337-348.

[24] С. Б. Родин. *О свойствах кодирования состояний автомата.* Интеллектуальные системы, 2017. Т.21, вып. 1, М., Сс. 97-111.

[25] Р. А. Ищенко. *Графы групповых автоматов*. Интеллектуальные системы, 2017. Т.21, вып. 2, М., Сс. 111-116.

[26] И. Е. Иванов. *Об автоматных функциях с магазинной памятью*. Интеллектуальные системы, 2018. Т.22, вып. 1, М., Сс. 39-110.

[27] П. А. Пантелеев. *Об обощении теоремы Мура*. Интеллектуальные системы, 2018. Т.22, вып. 1, М., Сс. 151-154.

[28] И. Ю. Самоненко. *О количестве регулярных языков, представимых в групповых гиперавтоматах*. Интеллектуальные системы, 2018. Т.22, вып. 2, М., Сс. 113-121.

[29] А. А. Часовских. *Проблема полноты в классах линейных автоматов*. Интеллектуальные системы, 2018. Т.22, вып. 2, М., Сс. 151-154.

Сведения об авторах

Дергач Петр Сергеевич
Младший научный сотрудник МГУ имени М. В. Ломоносова
e-mail: dergachpes@mail.ru.
Кудрявцев Валерий Борисович
Заведующий кафедрой МаТИС МГУ имени М. В. Ломоносова
e-mail: ilaky@bk.ru.

On the properties of languages that are stable to the drop/paste operations
Dergach P.S., Kudryavtsev V.B.

The article is devoted to the drop and paste operations, which have been promoted by V.I. Levenshtein. Closure operators are introduced for these operations. For the paste operator the existence, finiteness and uniqueness of the basis in closed classes are proved, and for the drop operator, non-existence for the infinite class and existence, finiteness and uniqueness for the finite are proved. The automata complexity of closed classes is investigated. The problems of completeness,precompleteness,expressibility are solved.

Keywords: drop and paste operations; closed class; regular language; basis; automata complexity; problems of completeness,precompleteness,expressibility.

К сведению авторов публикаций в журнале «Интеллектуальные системы. Теория и приложения»

В соответствии с требованиями ВАК РФ к изданиям, входящим в перечень ведущих рецензируемых научных журналов и изданий, в которых могут быть опубликованы основные научные результаты диссертаций на соискание ученой степени доктора и кандидата наук, статьи в журнал «Интеллектуальные системы. Теория и приложения» предоставляются авторами в следующей форме:

1. Статьи, набранные в пакете LaTeX, предоставляются к загрузке через WEB-форму http://intsysjournal.org/generator_form .
2. К статье прилагаются файлы, содержащие название статьи на русском и английском языках, аннотацию на русском и английском языках (не более 50 слов), список ключевых слов на русском и английском языках (не более 20 слов), информация об авторах: Ф.И.О. полностью, место работы, должность, ученая степень и/или звание (если имеется), контактные телефоны (с кодом города и страны), e-mail, почтовый адрес с индексом города (домашний или служебный).
3. Список литературы оформляется в едином формате, установленном системой Российского индекса научного цитирования.
4. За публикацию статей в журнале «Интеллектуальные системы. Теория и приложения» с авторов (в том числе аспирантов высших учебных заведений) статей, рекомендованных к публикации, плата не взимается. Оттиски статей авторам не предоставляются. Журнал распространяется по подписке, экземпляры журнала рассылаются подписчикам наложенным платежом. Условия подписки публикуются в каталоге НТИ «Роспечать», индекс журнала 64559.
5. Доступ к электронной версии последнего вышедшего номера осуществляется через НЭБ «Российский индекс научного цитирования». Номера, вышедшие ранее, размещаются на сайте http://intsysjournal.org, и доступ к ним бесплатный. Там же будут размещены аннотации всех публикуемых статей.

Подписано в печать: 10.12.2018
Дата выхода: 25.12.2018
Тираж: 200 зкз.
Цена свободная
Свидетельство о регистрации СМИ: ПИ № ФС77–58444 от 25 июня 2014 г., выдано Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций(Роскомнадзор).