ISSN 2220-3397

ТЕОРЕТИЧЕСКИЙ И ПРИКЛАДНОЙ НАУЧНО-ТЕХНИЧЕСКИЙ ЖУРНАЛ

Программная инженерия

ПрИн
2018
Том 9
3

Рисунки к статье С. З. Свердлова
«АВТОМАТИЧЕСКАЯ НАСТРОЙКА РЕЗКОСТИ
ПРИ УМЕНЬШЕНИИ ЦИФРОВЫХ ИЗОБРАЖЕНИЙ»

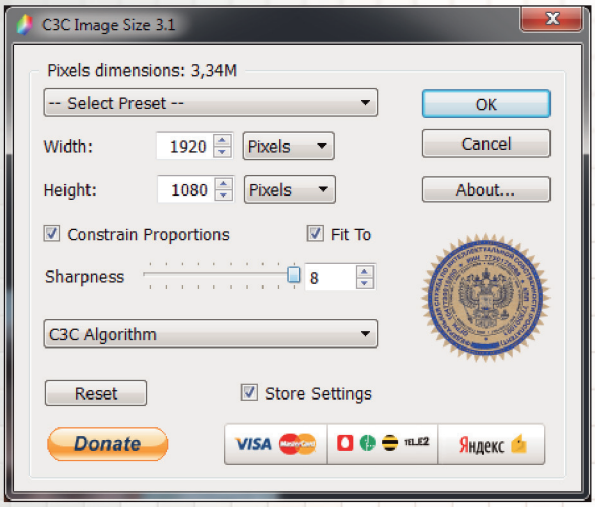
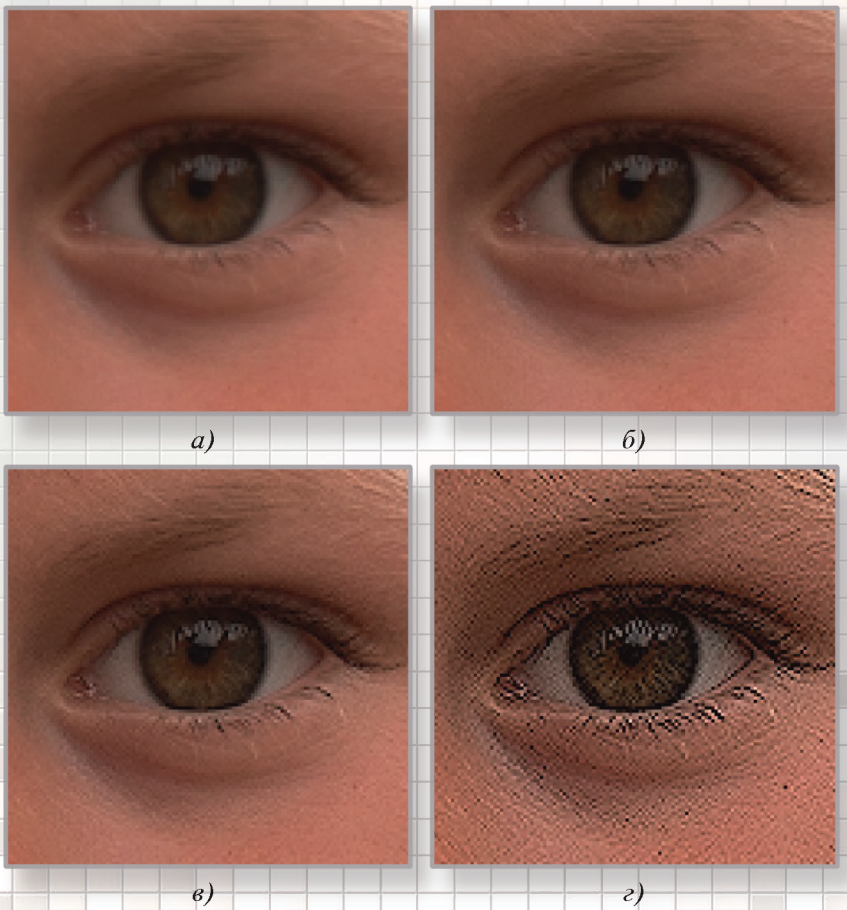


Рис. 1. Программа C3C Image Size. Движок Sharpness (резкость) на значении 8

Рис. 2. Егор. Портрет (исходное разрешение 2592×2592 пикселей)



а) *б)*

в) *г)*

Рис. 3. Фрагмент изображения: *а* – без усиления резкости; *б* – умеренное усиление резкости; *в* – существенное усиление резкости; *г* – чрезмерное усиление резкости

ТЕОРЕТИЧЕСКИЙ И ПРИКЛАДНОЙ НАУЧНО-ТЕХНИЧЕСКИЙ ЖУРНАЛ

Издается с сентября 2010 г. DOI 10.17587/issn.2220-3397 ISSN 2220-3397

Журнал издается при поддержке Отделения математических наук РАН, Отделения нанотехнологий и информационных технологий РАН, МГУ имени М.В. Ломоносова, МГТУ имени Н.Э. Баумана

СОДЕРЖАНИЕ

Журнал зарегистрирован в Федеральной службе по надзору в сфере связи, информационных технологий и массовых коммуникаций. Свидетельство о регистрации ПИ № ФС77-38590 от 24 декабря 2009 г.

Журнал распространяется по подписке, которую можно оформить в любом почтовом отделении (индексы: по каталогу агентства "Роспечать" — **22765**, по Объединенному каталогу "Пресса России" — **39795**) или непосредственно в редакции.
Тел.: (499) 269-53-97. Факс: (499) 269-55-10.
Http://novtex.ru/prin/rus E-mail: prin@novtex.ru
Журнал включен в систему Российского индекса научного цитирования.
Журнал входит в Перечень научных журналов, в которых по рекомендации ВАК РФ должны быть опубликованы научные результаты диссертаций на соискание ученой степени доктора и кандидата наук.

THEORETICAL AND APPLIED SCIENTIFIC AND TECHNICAL JOURNAL

SOFTWARE ENGINEERING

PROGRAMMNAYA INGENERIA

Vol. 9
N 3
2018

Published since September 2010 **DOI 10.17587/issn.2220-3397** **ISSN 2220-3397**

CONTENTS

Information about the jornal is available online at:
http://novtex.ru/prin/eng e-mail:prin@novtex.ru

УДК 004.02 DOI: 10.17587/prin.9.99-108

Г. Н. Калянов, д-р техн. наук, проф., гл. науч. сотр., e-mail: kalyanov@mail.ru,
Институт проблем управления РАН, г. Москва

О теории бизнес-процессов

Проанализировано современное состояние теории бизнес-процессов. Даны краткая историческая справка, источники и основы теории, а также проклассифицированы ее основные направления. По каждому из направлений приведен краткий обзор его основных моделей и методов.

Ключевые слова: *процесс, бизнес-процесс (БП), модель БП, язык моделирования БП, инжиниринг/реинжиниринг БП, верификация БП, требования по автоматизации БП*

Термин "бизнес-процесс" (БП) был введен в начале 1990-х гг. Майклом Хаммером и получил широкое распространение после публикации монографии [1], посвященной новому подходу к реорганизации БП. Однако исследованиями аналогичных объектов, называемых организационными (организационно-управляющими) процессами, деятельностями, работами и т. п., специалисты начали заниматься достаточно давно. В частности, Франк и Лилиан Гилбрет в 1921 г. в своем докладе в ASME (*American Society of Mechanical Engineers*) предложили нотацию карт процессов (*flow process chart*), которая с небольшими модификациями используется и по сегодняшний день. Знаковым этапом развития науки о БП являлся период конца 1960-х — начала 1970-х гг. — именно тогда появились широко используемые в настоящее время языки моделирования SADT-IDEF0, DFD, CFD, кластеры (прообразы классов в объектно-ориентированном подходе), а имена их авторов (Росс, Йодан, ДеМарко, Гейн, Сарсон, Лисков и др.) известны любому специалисту в рассматриваемой области. Однако приоритет безоговорочно принадлежит опубликованному еще до отмены крепостного права в России отечественному изданию [2], в котором был не только описан предшественник перечисленных выше языков моделирования, но предусмотрен и ряд конструкций современных языков моделирования БП класса EML (например, "плавательные дорожки").

Современная теория бизнес-процессов [3—5], с одной стороны, является одним из направлений теории процессов (модели которой подходят для изучения иерархических систем, включая организационные системы), в свою очередь, представляющей собой "раздел математической теории программирования, изучающий математические модели поведения динамических систем, называемых процессами" [6], где процесс определяется как "модель поведения, которое заключается в исполнении действий". Фактически в теории процессов процесс представлен диаграммой переходов состояний (*state transition diagram* — STD), включающей такие объекты, как состояние, начальное состояние, переход, действие (но не включающей условия перехода между состояниями). Отличия от классической STD (являющейся одной из базовых моделей БП) заключаются в наличии множества переменных процесса, начального условия процесса, использования операторов вместо действий (оператор — "схема действия, приобретающая вид конкретного действия лишь при конкретном выполнении этого оператора"). В качестве источников и предшественников теории процессов необходимо указать следующие направления:

1) теория взаимодействующих последовательных процессов Хоара [7], в которой исследуется модель взаимодействия синхронных параллельных процессов, основанная на передаче сообщений;

2) исчисление взаимодействующих систем Милнера [8], в котором введена денотационная семантика параллельных процессов, потоковый граф с синхронизированными портами;

3) алгебра взаимодействующих процессов [9] — теория первого порядка с равенством, в которой предметные переменные принимают значения в множестве процессов, используются формальные символы, соответствующие операциям над процессами.

С другой стороны, формальный аппарат, лежащий в основе теории БП, базируется на следующих направлениях теории программирования: формальные грамматики и языки; параллельные процессы и методы распараллеливания; теория тестирования программного обеспечения (ПО); методы оптимизации, верификации, анализа и оценки качества ПО; теория баз данных и баз знаний; структурные и объектно-ориентированные методы анализа и проектирования и др.

Необходимость формирования самостоятельной теории БП обусловливается следующими причинами:

- появились новые задачи, не стоявшие ранее и не решаемые в рамках общей теории процессов (например, реинжиниринг БП);
- в отдельных направлениях практически отсутствует формальный аппарат (например, для преобразования моделей БП в модели требований по их автоматизации);
- в других направлениях накопилось много неформализованных методов и моделей, необходима их систематизация.

Основные направления теории бизнес-процессов

Для целей настоящей статьи будем использовать следующие определения из работы [10]:

Операция — элементарное (неделимое) действие, выполняемое на одном рабочем месте.

Функция — совокупность операций, сгруппированных по определенному признаку.

Бизнес-процесс — связанная совокупность функций, в ходе выполнения которой потребляются определенные ресурсы и создается продукт (вещественный или нематериальный результат человеческого труда: предмет, услуга, научное открытие, идея), представляющий ценность для потребителя.

Подпроцесс — бизнес-процесс, являющийся структурным элементом некоторого объемлющего бизнес-процесса и представляющий ценность для внутреннего клиента.

Бизнес-модель — структурированное графическое описание сети процессов и/или функций/операций, связанных с данными, документами, организационными единицами и прочими объектами, отражающими существующую или предполагаемую деятельность предприятия.

В статье рассматриваются следующие разделы теории БП:

- языки моделирования и модели БП;
- технологии моделирования;
- методы структурирования/декомпозиции;
- методы инжиниринга/реинжиниринга;
- методы анализа и верификации;
- методы перехода от моделей БП к требованиям по автоматизации БП.

Языки моделирования и модели БП

Данный раздел теории БП опирается на структурный подход к моделированию БП. Концептуальная модель соответствующего структурного языка приведена в работе [11], она включает четыре базовых компонента — словарь языка, синтаксис языка, комплект абстрактных семантических правил/процедур, аспекты языковой прагматики.

Существует три вида базовых моделей БП — функциональная, информационная, поведенческая (все три могут включать элементы организационно-структурной модели). По типам модели БП разделяют на статические и динамические, по степени формализации — на слабоформализованные и формальные.

Наиболее популярные нотации моделирования и их возможности в части построения базовых видов моделей БП приведены в табл. 1.

В качестве формальной модели БП, реализующей приведенное выше определение БП, в работе [12] предложен смешанный граф. Нижний уровень модели содержит информационные объекты (ИО), представляемые с помощью кортежей $D_i\left(a_i^1, a_i^2, \ldots, a_i^n\right)$, где D_i — идентификатор i-го ИО, a_i^j — j-й атрибут i-го ИО. Бизнес-операция моделируется парой $T_iD_j = (T_i, D_j)$, где T_i — тип операции с ИО. При этом $T_iD_j = \left(T_ia_j^1, T_ia_j^2, \ldots, T_ia_j^k\right)$, однако для ряда операций (например, операции редактирования) могут существовать такие индексы m, что $T_ia_j^m = a_j^m$, т. е. операция может применяться не ко всем атрибутам ИО. Бизнес-функция моделируется кортежем бизнес-операций I_m $((T_{1m}, D_{1l}), \ldots, (T_{km}, D_{kl}))$, где I_m — код должности

Таблица 1

Нотация	Функции	Данные	Поведение	Оргструктура
DFD	+	–	–	– : +
CFD	+	–	+	– : +
IDEF0	+	–	–	– : +
ERD, IDEF1X	–	+	–	–
Структурограммы	–	+	–	–
STD	–	–	+	–
BPMN	– : +	–	+	+
ePC, IDEF3	+	–	–	– : +
Сети Петри, IDEF2	–	–	+	–
Смешанные графы	+	+	+	+

Примечание: "+" означает наличие модели данного вида; "–" — отсутствие модели; сочетание "– : +" — имеется частичная информация по модели данного вида.

исполнителя, T_{1m}, ..., T_{km} — элементы множества $\{T_j\}$, D_{1l}, ..., D_{kl} — элементы множества $\{D_j\}$. При этом, не нарушая общности, можно считать, что внутри бизнес-функции бизнес-операции имеют естественный порядок исполнения.

Формальная модель бизнес-процесса представляет собой граф управления бизнес-функциями G (N, n_0, $n_ф$, E, M, EM, EN, R, ER), где

- N — множество узлов, каждый из которых соответствует бизнес-функции;
- n_0 и $n_ф$ — входной и завершающий узлы, соответственно;
- E — множество управляющих ребер такое, что $i, j \in N \cup \{n_0, n_ф\}$: $(i, j) \in E$, если возможна ситуация, когда за выполнением бизнес-функции i будет выполняться бизнес-функция j;
- M — множество узлов, соответствующих структурным подразделениям предприятия ($M \cap N = \Omega$, где Ω — пустое множество);
- EM — множество ребер подчиненности такое, что $i, j \in M$: $(i, j) \in EM$, если структурное подразделение j подчинено структурному подразделению i;
- EN — множество ребер исполнения бизнес-функции такое, что $i \in M, j \in N$: $(i, j) \in EN$, если бизнес-функция j может быть выполнена в подразделении i;
- R — множество ресурсов предприятия;
- ER — множество взвешенных ребер использования ресурсов такое, что $i \in R$, $j \in N$: $(i, j) \in ER$, если бизнес-функция j использует при своем выполнении ресурс i.

Введенная графовая модель БП имеет следующие основные достоинства.

1. Модель интегрирует три базовых аспекта предприятия — его организационно-штатную структуру, технологии деятельности в разрезе реализуемых ими функций, а также используемые ресурсы в контексте использующих эти ресурсы направлений деятельностей.

2. Модель может служить интегрирующим ядром комплексной технологии проектирования и анализа БП, многие методы, включенные в описанные ранее разделы теории БП, базируются на предложенной модели.

3. Модель является внутренней с позиций пользователя, имеются алгоритмы трансляции моделей, представленные с помощью традиционных языков (DFD, IDEF0 и др.) в графовую модель и соответствующие обратные ретрансляторы.

Основными направлениями исследования в рамках данного раздела теории БП являются:

- унификация — создание универсального структурного языка по аналогии с UML (*Unified Modeling Language*) для объектно-ориентированного подхода к моделированию;
- разработка языков класса EML (*Enterprise Modeling Language*);
- разработка методов и алгоритмов трансляции в низкоуровневые (формализованные) языки моделирования;
- формализация синтаксиса и семантики языков моделирования БП.

Особенно много работ посвящено последнему вопросу. В частности, в работах [11, 13, 14] предлагаются различные виды грамматик, варьируемых от контекстно-зависимых до автоматных (в соответствии с классификацией по Хомскому) и различающихся по типу используемых символов (графовые грамматики, смешанные грамматики, модифицированные традиционные грамматики, например, RV-грамматики).

Табл. 2 отражает состояние дел в данном разделе теории БП.

Технологии моделирования

Современные технологии моделирования БП базируются на следующих основных принципах.

- Интеграция моделей различных видов, например, DFD-технология [15] (интегрирующая диаграммы DFD, CFD, ERD, STD и спецификации процессов в различных нотациях), схема Захмана [16] и развивающая ее модель "3D-предприятие" [17], онтологическая модель Бунге—Ванда—Вебера [18] и др.

Таблица 2

Языковые аспекты	Формальные методы	Формализованные методы	Неформальные методы
Нотации	Смешанные графы, сети Петри, STD (конечные автоматы), ERD (реляционная алгебра), структурограммы данных, таблицы и деревья решений, языки программирования	DFD, CFD, IDEF0, IDEF3, EPC, BPMN (BPML), модель Чена, FLOW-формы, структурные карты (схемы) и др.	Структурированный естественный язык, модели "верхнего уровня" (VACD, цепочка добавленной стоимости и др.)
Синтаксис	RV-грамматики, графовые грамматики, смешанные грамматики	RV-грамматики, графовые грамматики, смешанные грамматики	Не определяется
Семантика	Определяется нотацией	Частично определяется нотацией (статическая семантика)	Не определяется

- "Трансляция" статических моделей в динамические (прежде всего, в сети Петри), теория такого подхода предложена в работе [19], в качестве примеров коммерческих реализаций можно привести продукты Design/IDEF—Design/CPN, реализующие переход от IDEF0 к сети Петри, а также продукты CPN-AMI и INCOME, реализующие переход от DFD к сети Петри.
- "Трансляция" моделей простейших видов в более "развитые", в частности, переход от IDEF0 или DFD к вышеописанному смешанному графу [12].
- Переход от функциональной модели к информационной, соответствующие методы строятся по следующей схеме "функциональная модель — предварительная информационная схема — концептуальная информационная схема — ER-модель".
- Формализованное построение "правильной" модели, например, метод нормализации ERD с использованием нормальных форм Кодда [20], заключающейся в преобразовании схемы к наиболее простой третьей нормальной форме (3НФ). Отметим, что на практике схемы 1НФ и 2НФ имеют тенденцию возникать при попытке описать несколько реальных сущностей в одной схеме (заказ и книга, проект и сотрудник). Модель 3НФ является наиболее простым способом представления данных, отражающим здравый смысл. Построив 3НФ, можно фактически выделить базовые сущности предметной области.

Методы структурирования/декомпозиции

Базовые принципы структурирования приведены в работе [21] и кратко сводятся к перечисленным далее.

- Структурирование должно осуществляться в соответствии с деятельностями (группами однотипных БП) и бизнес-процессами предприятия, а не в соответствии с его организационно-штатной структурой.
- Верхний уровень модели должен отражать только контекст системы.
- На втором уровне модели должны быть отражены основные деятельности предприятия и их взаимосвязи.
- Каждая из деятельностей, в свою очередь, должна быть детализирована на бизнес-процессы (желательно, единственного уровня).
- Дальнейшая детализация бизнес-процессов осуществляется посредством бизнес-функций (обычно 2—3 уровня).
- Общее число уровней в модели не должно превышать 6—7.
- При структурировании данных необходимо выполнять "правило накопителей" (если информационный объект используется единственным БП, то он не должен присутствовать на более высоком уровне, при этом на втором уровне модели вводятся базовые накопители — Сотрудники (данные о сотрудниках), Оборудование, Произведенная продукция и т. п.).

При структурировании немаловажную роль играют различные классификации БП (наиболее часто используемые из них приведены в работе [21]), фактически, на их основе строятся различные методики структурирования.

В качестве примера ниже приведена классификация, апробированная автором в ряде проектов по моделированию и автоматизации БП для отечественных предприятий.

- Основными бизнес-процессами являются процессы, ориентированные на производство товара или оказание услуги, являющихся целевыми объектами создания предприятия и обеспечивающих получение дохода (например, для жирового комбината — процессы производства масла и майонеза).
- Сопутствующими бизнес-процессами являются процессы, ориентированные на производство товара или оказание услуги, являющихся результатами сопутствующей основному производству производственной деятельности предприятия и также обеспечивающих получение дохода (для жирового комбината — процессы производства мыла и глицерина).
- Вспомогательными бизнес-процессами являются процессы, предназначенные для жизнеобеспечения основных и сопутствующих процессов, которые ориентированы на поддержку отражающих их специфику направлений деятельности (для жирового комбината, например, это процессы ремонта и технического обслуживания маслосбойного оборудования).
- Обеспечивающими бизнес-процессами являются процессы, предназначенные для жизнеобеспечения основных и сопутствующих процессов и ориентированные на поддержку их универсальных черт (для любого предприятия — процесс финансового обеспечения деятельности, процесс управления кадрами, процесс юридического обеспечения и т. п.).
- Бизнес-процессы управления — это процессы, охватывающие весь комплекс функций управления на уровне каждого бизнес-процесса и предприятия в целом. Примерами таких процессов могут являться процессы стратегического, оперативного и текущего планирования, процессы формирования и выполнения управляющих воздействий.
- Наконец, бизнес-процессами развития являются процессы совершенствования производимого товара или услуги, процессы развития технологий, процессы модификации оборудования, а также инновационные процессы.

В предложенном в работе [22] подходе процесс декомпозиции определяется как случайный процесс, доказывается возможность интерпретации процесса декомпозиции БП моделью ветвящегося процесса Гальтона—Ватсона [23], строится соответствующая вероятностная модель процесса декомпозиции, оценивается ожидаемое число элементов функциональной модели БП, время проектирования вариантов БП, трудоемкость проектирования, глубина декомпозиции и т. п. Следует особо отметить, что автору

подхода методами теории вероятности удалось обосновать "магическое число 7", ограничивающее число уровней модели, а также число функциональных объектов на каждой из диаграмм.

В работе [24] предложен метод построения модели БП, в котором в качестве необходимого условия разбиения сквозного БП на цепочку взаимодействующих подпроцессов рассматривается смена объекта управления процесса, а в качестве достаточного условия выступает перегруппировка потоков управления, доказано соответствие критериям "правильной" декомпозиции модели бизнес-процесса, разработанным на основе анализа онтологической модели Бунге—Ванда—Вебера.

В работе [12] введены метрики оценки качества декомпозиции, базирующиеся на общих свойствах "хорошей" системы — сцепление (механизм взаимодействия между компонентами БП) и связность (механизм внутренней структурной организации компонент), предложено и ранжировано множество типов сцепления и связности (при этом статистические данные показывают, что более 70 % реальных БП соответствуют введенным типам), разработан метод оценки качества, обеспечивающий для любого БП определение типа его сцепления и связности, а также проектирование БП с заданным типом сцепления и связности.

В работе [25] предложены:

- критерий глубины детализации функциональной модели, обеспечивающий необходимый и достаточный уровень детализации;
- критерии верификации полноты состава информационных объектов модели предметной области;
- метод выявления ограниченного состава информационных сущностей, позволяющий ограничить размерность корпоративной модели данных при сохранении ее логической целостности.

На примере нескольких процессов банковской деятельности проведена оценка количественных характеристик сцепления БП по информационным сущностям, установлено, что зависимость сцепления функциональной модели от характеристик сущностей носит нелинейный характер (при этом для различных проектов форма кривых аналогична). Установлено также следующее:

- 10...15 % сущностей с наибольшими коэффициентами использования оказывают наибольшее влияние на суммарное сцепление модели (компактная группа сущностей, определяемых как общесистемные);
- около 80 % сущностей имеют малые коэффициенты использования, определяющие их локальную значимость для отдельных БП.

Интерес представляют и методы оценки качества БП на базе метрик качества программного обеспечения, разработанные магистрантами Московского физико-технического института (МФТИ) в рамках подготовки магистерских диссертаций. Их идея заключается в формировании комплексного метода оценки на основе объемных и топологических метрик (таких как мера Холстеда, цикломатическая мера сложности Мак-Кейба, мера Хенри—Кафуры и др.) и процедур экспертного оценивания.

С практической позиции существенную помощь при структурировании БП обеспечивает комплекс методов разработки регламентов и методик БП, предложенный в работе [26].

Методы инжиниринга/реинжиниринга

В работе [27] предложен подход к инжинирингу предприятий на основе применения интеллектуальных технологий. Представлена эволюция этого направления в аспектах реинжиниринга и управления БП с использованием динамических интеллектуальных систем, имитационного моделирования и систем управления знаниями. Показана интеграционная парадигма инжиниринга предприятий, использующая методы стратегического инжиниринга на основе когнитивного и интеллектуального анализа данных, развития сервисно-ориентированных архитектур предприятий с использованием многоагентных технологий и онтологического инжиниринга предприятий.

В работах [12, 28] предложена формализованная методология инжиниринга/реинжиниринга, интегрирующим ядром которой является вышеприведенная графовая модель БП. Центральное место в методологии занимает метод проектирования вариантов БП на основе аппарата формальных грамматик БП, позволяющий, с одной стороны, расширить число анализируемых вариантов выполнения бизнес-процесса вплоть до их полного перебора, с другой стороны, автоматически отсечь большую часть вариантов, неприемлемых по ряду объективных и субъективных критериев. Соответствующая грамматика является параллельной атрибутной порождающей грамматикой БП и представляет собой следующую упорядоченную девятку объектов: $G = (V_T, V_N, V_0, P, A_s, M_s, A_n, M_n, C)$, где

- V_T — множество терминальных символов;
- V_N — множество нетерминальных символов;
- $V_0 \subseteq V_N$ — множество начальных символов;
- P — множество порождающих правил;
- A_s — конечное множество синтезируемых атрибутов;
- M_s — множество методов синтеза атрибутов;
- A_n — конечное множество наследуемых атрибутов;
- M_n — множество методов наследования атрибутов;
- C — множество символов, определяющих параллелизм.

Первые четыре объекта G определяют традиционным образом порождающую грамматику. Следующие четыре объекта определяют множество свойств (атрибутов), характеризующих символы порождаемых цепочек, и правила обработки этих свойств. Последний символ предназначен для обеспечения возможности порождения подцепочек бизнес-функций, которые могут (но не обязательно должны) выполняться параллельно.

Таким образом, грамматика БП представляет собой порождающую грамматику с дополнительными механизмами передачи информации и обработки некоторых не контекстно-свободных аспектов синтаксиса языка моделирования бизнес-процесса (например, требования ограничений по ресурсам). Неформально атрибутную грамматику можно определить как порождающую грамматику, нетерминальным и терминальным символам которой приписывают таблицы значений атрибутов. Атрибуты подобной грамматики разделяют на два типа: синтезируемые и наследуемые. Синтезируемые атрибуты вычисляют по методам, связанным с нетерминалами, входящими в левую часть порождающего правила, тогда как наследуемые атрибуты — по методам, связанным с терминалами и нетерминалами, входящими в правую часть порождающего правила. Другими словами, значение синтезируемого атрибута некоторого нетерминального символа определяется символами, расположенными в дереве вывода под этим нетерминалом. Такое восходящее вычисление выражается в том, что методы вычисления атрибутов, ассоциированные с правилами вывода, указывают, как вычислять атрибуты в левой части правила по заданным атрибутам символов правой части. Значение наследуемого атрибута ограничивается деревом вывода над соответствующим символом.

В настоящее время в рамках данного раздела теории БП ведутся работы по формализации свойств "хороших" (согласно [1]) процессов, таких как горизонтальное и вертикальное уплотнение, версионность и т. д.

Методы анализа и верификации

В настоящем разделе описаны методы тестирования БП, методы статического анализа потоков данных, методы динамического анализа на базе сетей Петри и др., т. е. методы анализа и верификации, предназначенные, прежде всего, для выявления и локализации ошибок в БП.

На практике обнаружение и локализация ошибок в БП осуществляется во время его функционирования в реальных экономических условиях, что может привести и, как правило, приводит к плачевным результатам. Поэтому актуальной является задача выявления ошибок на стадиях планирования (проектирования) и создания БП, т. е. до того, как он начнет реально функционировать.

При решении поставленной задачи целесообразно использовать результаты исследований, лежащие в основе теории тестирования и отладки компьютерных программ, как наиболее близких к БП объектов. Подобие БП и программ заключается в следующем:

- в основе обоих объектов лежит понятие алгоритма;
- оба объекта имеют схожие этапы жизненного цикла;
- оба объекта могут выполняться как последовательно, так и параллельно;
- оба объекта адекватно моделируются с использованием графовых моделей.

Однако БП является гораздо более сложным и непредсказуемым объектом, чем обычная компьютерная программа, в том числе и параллельная. Если последняя содержит ряд управляющих параллелизмом механизмов, таких как механизмы синхронизации ветвей, то выполнение БП, вообще говоря, непредсказуемо. В то же время даже для последовательных программ задача тестирования является трудноразрешимой. Известно, что полное и исчерпывающее ее тестирование практически невозможно, так как требует огромных трудозатрат. Таким образом, перефразировав известное утверждение Дейкстры (касающееся ПО), можно сказать, что любой бизнес-процесс содержит хотя бы одну ошибку — просто пока еще не были созданы условия для ее проявления.

В работах [12, 29] выделяется класс наиболее типичных для БП ошибок, связанных с информационными ресурсами (ошибок в потоках данных) и предлагается комплекс методов и средств их обнаружения и локализации. Примерами таких ошибок являются:

- создание информационных объектов (ИО) и/или их атрибутов, не используемых в дальнейшей деятельности;
- отсутствие и/или неполнота ИО и/или их атрибутов;
- дублирование ИО и/или их атрибутов и, как следствие, их несогласованность и противоречивость и др.

Специфика данных ошибок обусловливается наличием регламентов доступа к атрибутам ИО, запрещающих или ограничивающих доступ при выполнении ряда бизнес-операций. Так, например, такой атрибут сотрудника, как его зарплата, на ряде предприятий доступен только руководству и сотрудникам бухгалтерии.

В рамках метода тестирования [12, 29] предложена модель потоков данных БП, основанная на отношениях определения и использования ИО при различных масках (определяющих, например, права доступа к данным). Предложенный метод тестирования позволяет:

- обеспечить обнаружение специфических для БП ошибок в потоках данных, связанных с их обработкой под различными масками, обеспечивающими регламенты доступа, не обнаруживаемых другими известными методами тестирования;
- обеспечить выявление всех тех ошибок, обнаружение которых может проводиться с помощью традиционных критериев, основанных на анализе графовых моделей объектов.

Метод статического анализа потоков данных [30] обеспечивает автоматическое обнаружение ошибок в "статической семантике" БП. Метод основан на введении специальной дисциплины взаимодействия состояний ИО на любом этапе выполнения БП. Состояние ИО определяется последним обращением к нему и задается следующим образом:

$$S_i(A = (a_1, a_2, ..., a_m)) = (i, Q = (q_1, q_2, ..., q_m), D = (d_1, d_2, ..., d_m)),$$

где i — номер узла графа бизнес-процесса;

q_j — тип обращения к j-му атрибуту ИО: $q_j \in \{w, r, n\}$, где w — определение атрибута, r — использование атрибута, n — отсутствие обращения к атрибуту;

d_j — элемент маски доступа к j-му атрибуту ИО: $d_j \in \{W, R, N\}$, где W — разрешение доступа на определение атрибута, R — разрешение доступа на использование атрибута, N — запрещение доступа к атрибуту.

Для корректной работы БП по крайней мере должны удовлетворяться следующие пять правил, касающихся последовательности состояний ИО.

1. Последовательность не должна содержать цепочек $...d^i q^i...$, в которых $d^i = N$, а $q^i \in \{w, r\}$.
2. Последовательность не должна содержать цепочек $...d^i q^i...$, в которых $d^i = R$, а $q^i = w$.
3. $\forall i$ такого, что $(q^i = r) \wedge (d^i \neq N)\ \exists j < i$ такое, что $(q^j = w) \wedge (d^j = W)$.
4. $\forall i, j$ таких, что $(q^i = w) \wedge (d^i = W) \wedge (q^j = w) \wedge \wedge (d^j = W)\ \exists i < k < j$ такое, что $(q^k = r) \wedge (d^k \neq N)$.
5. $\forall i$ такого, что $(q^i = w) \wedge (d^i = W) \wedge (\neg\exists\ j > i$ такого, что $(q^j = w) \wedge (d^j = W))\ \exists k > i$ такое, что $(q^k = r) \wedge (d^k \neq N)$.

Нарушения перечисленных правил вызывают ошибки при выполнении бизнес-процесса (или, по крайней мере, являются симптомами ошибок) и могут происходить по вышеуказанным причинам.

Динамический анализ на базе сетей Петри [31] является наиболее развитым направлением в данном разделе теории БП. На практике обычно применяют сложные и развитые сети Петри. Модификации, как правило, касаются следующих трех моментов:

- введение иерархии (иерархические сети Петри);
- определение различий в маркерах, каждый из которых имеет свои уникальные характеристики (цветные/раскрашенные сети Петри);
- введение многоместных (содержащих несколько маркеров) позиций, как последовательных, так и параллельных (сети Петри с многоместными позициями).

Из исследований на этом направлении необходимо отметить работы [19, 32], позволяющие решать ряд дополнительных задач помимо решения традиционных задач в рамках классической теории сетей Петри (достижимость и т. д.).

Методы перехода от моделей БП к требованиям по автоматизации БП

Одним из первых методов перехода от моделей БП к требованиям по их автоматизации являлся метод перехода от функциональной модели БП в виде иерархии DFD-диаграмм к структурным картам (моделям этапа проектирования системы автоматизации БП), предложенный в работе [33]. При этом исследования по оценке качества построенных моделей не проводились.

В работе [34] предложен метод формирования и анализа требований к системе автоматизации, основанный на модификации приведенного выше графа БП. При этом исследование модели требований (редуцированного графа) и ее последующая оценка происходит по следующим направлениям: расчет метрик сцепления и связанности, оценка соответствия модели БП, оценка по ряду критериев качества моделирования.

Следует отметить, что данный раздел теории БП в настоящее время является наименее развитым.

Табл. 3 систематизирует состояние дел в теории БП по ее пяти представленным выше разделам.

Таблица 3

Раздел теории БП	Формальные методы	Формализованные методы	Неформальные методы
Технологии моделирования	Трансляционные технологии, нормальные формы Кодда	Интеграционные технологии	Йодан—Де Марко, Гейн—Сарсон, SADT и др.
Методы декомпозиции	Ветвящиеся процессы, метрики сцепления и связности, критерии глубины детализации и полноты состава	Онтологическая модель Бунге—Ванда—Вебера	Базируются на лучших практиках и личном опыте аналитика
Реинжиниринг	Порождающие грамматики БП	Онтологический подход	BPR, CPI/TQM, BSP и др.
Анализ и верификация моделей	Тестирование, статический анализ, сети Петри, критерии качества	Функционально-стоимостной анализ	Применяются общие методы системного анализа
Переход к моделям требований к системе автоматизации	Отсутствуют	DFD-структурные карты, редуцирование графа	Базируются на лучших практиках и личном опыте проектировщика

Параллельные бизнес-процессы

Рассмотренные в статье модели и методы практически не затрагивают вопросы параллельной работы БП. Для решения задач, связанных с параллелизмом, были выделены следующие классы БП, основанные на известной классификации ЭВМ Флинна:

- ОКОД (одиночный поток команд и одиночный поток данных) — последовательные БП;
- МКОД (множественный поток команд и одиночный поток данных) — конвейерные БП;
- ОКМД (одиночный поток команд и множественный поток данных) — синхронные (векторные и матричные) БП;
- МКМД (множественный поток команд и множественный поток данных) — асинхронные БП.

В работах [35, 36] выделен и исследован специальный класс конвейерных БП, который существенно шире множества "классических" конвейеров, и позволяет при этом вычислять основные характеристики процесса (в том числе и те, которые нельзя вычислить с помощью имитационного моделирования). Модель конвейерного процесса имеет перечисленные далее основные отличия от классической модели конвейера.

- Процесс описывается не параллельно-последовательной схемой, а ациклическим ориентированным графом, вершинами которого могут быть функциональные операции или спусковые функции.
- Числовые характеристики процесса определяются не аналитическими, а рекуррентными выражениями.

Разработана теория моделирования конвейерных бизнес-процессов, включающая в себя следующие основные компоненты.

- Набор функциональных примитивов, позволяющих описывать конвейерные БП.
- Формальное математическое описание модели в виде набора рекуррентных отношений, позволяющих однозначно описать основные характеристики процесса.
- Набор характеристик, определяющий функционирование конвейера: производительность, время начала стационарного режима работы и т. п.
- Структурный метод анализа модели, позволяющий в статике (без имитационного моделирования) вычислять основные характеристики процесса: производительность; время выхода в стационарный режим работы; характеристическую операцию конвейера, определяющую производительность конвейера.
- Теорема, доказывающая, что на любом цикле работы конвейера его фаза изменяется по закону некоторого индуцированного им линейного конвейера.

В части синхронных БП из соответствующего раздела теории программирования в качестве основы могут быть использованы результаты из следующих ее подразделов:

1) векторные и матричные структуры данных и методы их размещения и обработки, языки программирования векторных вычислений [37];

2) методы тестирования, анализа и верификации векторных программ [38].

Для реализации методов первого подраздела необходимо соответствующее расширение информационных моделей БП (в частности, ERD), вместе с тем расширение соответствующих языков построения функциональных моделей не является целесообразным ввиду специфики рассматриваемого объекта и усложнения понимаемости его модели.

Что касается второго подраздела, то решение соответствующих задач обеспечивается перечисленными выше методами и моделями для последовательных БП.

Сложнее выглядит ситуация для класса асинхронных БП. В соответствующем разделе теории программирования выделяются следующие методы для решения задач параллельного программирования:

- синхронизирующие примитивы (семафоры Дейкстры, условные интервалы Хоара и др.);
- методы распараллеливания последовательных программ;
- параллельные грамматики и языки;
- методы параллельной трансляции и т. д.

Подобные исследования, проводимые как отечественными, так и зарубежными специалистами, автору не известны.

Список литературы

1. **Hammer M., Champy J.** Reengineering the Corporation: A Manifesto for Business Revolution. N. Y.: Harper-Collins, 1993. 288 p.
2. **Добролюбов Н. А.** Руководство к наглядному изучению административного порядка течения бумаг в России. Москва, 1858. 20 с.
3. **Калянов Г. Н.** Модели и методы теории бизнес-процессов // Открытое образование. 2015. № 6. С. 4—9.
4. **Калянов Г. Н.** Формальные методы теории бизнес-процессов // Современные информационные технологии и ИТ-образование. 2015. Том 1, № 11. С. 628—632.
5. **Калянов Г. Н.** Теория бизнес-процессов: формальные модели и методы // Экономика, статистика и информатика. 2016. № 4. С. 19—21.
6. **Миронов А. М.** Теория процессов. Переславль-Залесский: Университет г. Переславля, 2008. 345 с.
7. **Hoare C. A. R.** Communicating Sequential Processes. Prentice Hall, 1985. 238 p.
8. **Milner R.** A Calculus of Communicating Systems. Number 92 in Lecture Notice in Computer Science. Springer Verlag, 1980. 238 p.
9. **Bergstra J., Bethke I.** Process Algebra for Synchronous Communicaition // Information and Control. 1984. No. 60 (1/3). P. 109—137.
10. **Калянов Г. Н.** Моделирование, анализ, реорганизация и автоматизация бизнес-процессов. М.: Финансы и статистика, 2006. 240 с.
11. **Калянов Г. Н.** Концептуальная модель DFD-технологии // Открытое образование. 2017. № 4. С. 21—26.
12. **Калянов Г. Н.** Теория и практика реорганизации бизнес-процессов. М.: СИНТЕГ, 2000. 212 с.
13. **Zhang D.-Q., Zhang K., Cao J.** A context-sensitive graph grammar formalism for the specification of visual languages // The Computer Journal. 2001. Vol. 44, No. 3. P. 186—200.
14. **Афанасьев А. Н., Шаров О. Г., Войт Н. Н.** Анализ и контроль диаграмматических моделей при проектировании сложных автоматизированных систем. Ульяновск, УлГТУ, 2016. 87 с.
15. **Калашян А. Н., Калянов Г. Н.** Структурные модели бизнеса: DFD-технологии. М.: Финансы и статистика, 2003. 256 с.

16. **Zachman J. A.** A Framework for Information Systems Architecture // IBM Syst. J. 1987. Vol. 26, No. 3. P. 276—292.
17. **Зиндер Е. З.** "3D-предприятие" — модель трансформирующейся системы // Директор ИС. 2000. № 4. С. 16—19.
18. **Wand Y., Weber R.** An Ontological Model of an Information System // IEEE Transactions on Software Engineering. 1990. Vol. 16, No. 11. P. 1282—1292.
19. **Юдицкий С. А.** Сценарный подход к моделированию поведения бизнес-систем. М.: СИНТЕГ, 2001. 112 с.
20. **Дейт К. Дж.** Введение в системы баз данных. М.: Вильямс, 2006. 782 с.
21. **Калянов Г. Н.** Консалтинг: от бизнес-стратегии к корпоративной информационно-управляющей системе. М.: Горячая линия — Телеком, 2011. 210 с.
22. **Циперман Г. Н.** Стохастическая модель процесса идентификации сервисов информационной системы // Труды ИСП РАН. 2014. Т. 26, Вып. 5. С. 7—28.
23. **Ватутин В. А.** Ветвящиеся процессы и их применение. М.: МИАН, 2008, 108 с.
24. **Фёдоров И. Г.** Методология создания исполняемой модели и системы управления бизнес-процессами. М.: МЭСИ, 2015. 156 с.
25. **Тудер И. Ю.** Коллективное моделирование предметной области большой размерности. Автореферат диссертации на соискание ученой степени кандидата технических наук. М.: РосНИИ ИТиАП, 2002. 24 с.
26. **Елиферов В. Г., Репин В. В.** Бизнес-процессы: регламенты и методики. М.: ИНФРА-М, 2005. 319 с.
27. **Тельнов Ю. Ф.** Реинжиниринг бизнес-процессов: компонентная методология. М.: Финансы и статистика, 2004. 212 с.
28. **Калянов Г. Н.** Формальные методы поддержки реорганизации бизнес-процессов // Экономика, статистика и информатика. 2013. № 3. С. 161—165.
29. **Калянов Г. Н.** Тестирование информационных потоков // Труды 13-й Российской научно-практической конференции "Реинжиниринг бизнес-процессов на основе современных информационных технологий. Системы управления процессами и знаниями". Москва, 2010. С. 146—151.
30. **Калянов Г. Н.** Формальные методы инжиниринга и верификации бизнес-процессов // Труды 16-й научно-практической конференции "Инжиниринг предприятий и управление знаниями". Москва, 2013. С. 144—149.
31. **Питерсон Д.** Теория сетей Петри и моделирование систем. М.: Мир, 1984. 264 с.
32. **Mendling J., Reijers H., Van der Aalst W.** Seven Process Modeling Guidelines // Information and Software Technology. 2010. Vol. 52, No. 2. P. 127—136.
33. **Гейн К., Сарсон Т.** Системный структурный анализ: средства и методы. М.: Эйтекс, 1992. 400 с.
34. **Калянов Г. Н.** Системный анализ требований к КИС // Труды 15-й Российской научно-практической конференции "Реинжиниринг бизнес-процессов на основе современных информационных технологий. Системы управления знаниями". Москва, 2012. С. 96—98.
35. **Куприянов Б. В.** Моделирование конвейерных бизнес-процессов // Сборник трудов "Управление большими системами". Москва. 2010. Вып. 28. С. 230—273.
36. **Куприянов Б. В.** Метод эффективного анализа модели рекурсивного конвейерного процесса // Автоматика и телемеханика. 2017. № 3. С. 63—79.
37. **Разбегин В. П., Калянов Г. Н., Куприянов Б. В.** Система программирования векторных вычислений // Программирование. 1985. № 4. С. 25—32.
38. **Калянов Г. Н.** Тестирование программ параллельных вычислительных систем с общим управлением // Программирование. 1986. № 2. С. 41—47.

On the Theory of Business Processes

G. N. Kalyanov, kalyanov@mail.ru, Institute of Control Sciences Russian Academy of Sciences, Moscow, 117997, Russian Federation

Corresponding author:

Kalyanov Georgyi N., Professor, Institute of Control Sciences Russian Academy of Sciences, Moscow, 117997, Russian Federation
E-mail: kalyanov@mail.ru

Received on December 14, 2017
Accepted on December 21, 2017

The current state of the theory of business processes is analyzed. The brief historical background, sources and fundamentals of the theory are classified. For each of these areas a brief overview of the main models and methods is provided.

The necessity of forming an independent theory of BP is due to the following reasons:

• there are new tasks, not put before and not solved in the general theory of processes (e. g., reengineering BP);

• the formal apparatus absent in some areas (for example, to convert models to model the requirements for their automation);

• the other directions have a lot of non-formalized methods and models, there is the need for their systematization.

This article discusses the following parts of the theory:

• modeling languages and BP-models,

• BP modeling technology,

• methods of structuring/decomposition of BP,

• methods of engineering/re-engineering of BP,

• methods of analysis and verification of BP,

• methods of transition from BP-models to requirements for automation of BP.

The theory of business processes is an integral part of the general theory of processes, which is a branch of mathematical theory of programming, studying mathematical models of the behavior of dynamical systems.

The formal apparatus underlying the theory of BP was based on the theory of programming: formal grammar and languages; parallel processes and methods of paralleling; the theory of software testing; methods of optimization, verification, analysis and evaluation of quality; the theory of databases and knowledge bases; structural and object-oriented analysis and design etc.

Keywords: *process, business process, model, verification, engineering /reengineering, requirements to automation*

For citation:

Kalyanov G. N. On the Theory of Business Processes, *Programmnaya Ingeneria,* 2018, vol. 9, no. 3, pp. 99—108.

DOI: 10.17587/prin.9.99-108

References

1. **Hammer M., Champy J.** *Reengineering the Corporation: A Manifesto for Business Revolution.* N. Y.: Harper-Collins, 1993, 288 p.

2. **Dobrolyubov N. A.** *Rukovodstvo k naglyadnomu izucheniyu administrativnogo poryadka techeniya bumag v Rossii* (A guide to the graphic study of the administrative order of the flow of securities in Russia), Moscow, 1858, 20 p. (in Russian).

3. **Kalyanov G. N.** Modeli i metody teorii biznes-processov (Models and methods of the theory of business processes), *Otkrytoe obrazovanie*, 2015, no. 6, pp. 4—9 (in Russian).

4. **Kalyanov G. N.** Formal'nye metody teorii biznes-processov (Formal methods of the theory of business processes), *Sovremennye informacionnye tekhnologii i IT-obrazovanie*, 2015, vol. 1, no. 11, pp. 628—632 (in Russian).

5. **Kalyanov G. N.** Teoriya biznes-processov: formal'nye modeli i metody (The theory of business processes: formal models and methods), *Ekonomika, statistika i informatika*, 2016, no. 4, pp. 19—21 (in Russian).

6. **Mironov A. M.** *Teoriya processov* (Process theory), Pereslavl'-Zalesskij: Universitet g. Pereslavlya, 2008, 345 p. (in Russian).

7. **Hoare C. A. R.** *Communicating Sequential Processes*, Prentice Hall, 1985. 238 p.

8. **Milner R.** *A Calculus of Communicating Systems.* Number 92 in Lecture Notice in Computer Science. Springer Verlag, 1980, 238 p.

9. **Bergstra J.** Process Algebra for Synchronous Communicaition, *Information and Control,* 1984, no. 60 (1/3), pp. 109—137.

10. **Kalyanov G. N.** *Modelirovanie, analiz, reorganizaciya i avtomatizaciya biznes-processov* (Modeling, analysis, reorganization and automation of business processes), Moscow, Finansy i statistika, 2006, 240 p. (in Russian).

11. **Kalyanov G. N.** Konceptual'naya model' DFD-tekhnologii (A conceptual model of DFD-technology*), Otkrytoe obrazovanie*, 2017, no. 4, pp. 21—26 (in Russian).

12. **Kalyanov G. N.** *Teorija i praktika reorganizacii biznes-processov* (Theory and practice of the reorganization of business processes), Moscow, SINTEG, 2000, 212 p. (in Russian).

13. **Zhang D.-Q., Zhang K., Cao J.** A context-sensitive graph grammar formalism for the specification of visual languages, *The Computer Journal*, 2001, vol. 44, no. 3, pp. 186—200.

14. **Afanasev A. N., Sharov O. G., Voit N. N.** *Analiz i kontrol' diagrammaticheskih modelej pri proektirovanii slozhnyh avtomatizirovannyh sistem* (Analysis and control diagrammatically models in the design of complex automated systems), Ul'yanovsk, UlGTU, 2016, 87 p. (in Russian).

15. **Kalashan A. N., Kalyanov G. N.** *Strukturnye modeli biznesa: DFD-tekhnologii* (Structural business model: DFD-technology), Moscow, Finansy i statistika, 2003, 256 p. (in Russian).

16. **Zachman J. A.** A Framework for Information Systems Architecture, *IBM Syst. J.*, 1987, vol. 26, no. 3, pp. 276—292.

17. **Zinder E. Z.** "3D-predpriyatie" — model' transformiruyushchejsya sistemy ("3D-enterprise" model of transformed system), *Direktor IS,* 2000, no. 4, pp. 16—19 (in Russian).

18. **Wand Y., Weber R.** An Ontological Model of an Information System, *IEEE Transactions on Software Engineering*, 1990, vol. 16, no. 11, pp. 1282—1292.

19. **Juditshkyi S. A.** *Scenarnyj podhod k modelirovaniyu povedeniya biznes-sistem* (Scenario approach to modelling for business systems), Moscow, SINTEG, 2001, 112 p. (in Russian).

20. **Date K.** *Vvedenie v sistemy baz dannyh* (Introduction to database systems), Moscow, Willams, 2006, 782 p. (in Russian).

21. **Kalyanov G. N.** *Konsalting: ot biznes-strategii k korporativnoj informacionno-upravlyayushchej sisteme* (Consulting: from business strategy to corporate information management system), Moscow, Goryachaya liniya — Telekom, 2011, 210 p. (in Russian).

22. **Tshiperman G. N.** Stohasticheskaya model' processa identifikacii servisov informacionnoj sistemy (A stochastic model of the process of identification of services of the information system*), Trudy ISP RAN*, 2014, vol. 26, no. 5, pp. 7—28 (in Russian).

23. **Vatutin V. A.** *Vetvyashchiesya processy i ih primenenie* (Branching processes and their applications), Moscow, MIAN, 2008, 108 p. (in Russian).

24. **Fedorov I. G.** *Metodologiya sozdaniya ispolnyaemoj modeli i sistemy upravleniya biznes-processami* (The methodology for creating executable models and systems of control of business processes), Moscow, MESI, 2015, 156 p. (in Russian).

25. **Tuder I. Y.** *Kollektivnoe modelirovanie predmetnoj oblasti bol'shoj razmernosti* (Collective domain modeling of large dimension), Avtoreferat dissertacii na soiskanie uchenoj stepeni kandidata tekhnicheskih nauk, Moscow, RosNII ITiAP, 2002, 24 p. (in Russian).

26. **Eliferov V. G., Repin V. V.** *Biznes-processy: reglamenty i metodiki* (Business processes: regulations and methods), Moscow, INFRA-M, 2005, 319 p. (in Russian).

27. **Telnov J. F.** *Reinzhiniring biznes-processov: komponentnaya metodologiya* (Reengineering business processes: a component methodology), Moscow, Finansy i statistika, 2004, 212 p. (in Russian).

28. **Kalyanov G. N.** *Formal'nye metody podderzhki reorganizacii biznes-processov* (Formal methods to support the reengineering of business processes), *Ekonomika, statistika i informatika*, 2013, no. 3, pp. 161—165 (in Russian).

29. **Kalyanov G. N.** Testirovanie informacionnyh potokov (Testing of information flows*), Trudy 13-j Rossijskoj nauchno-prakticheskoj konferencii "Reinzhiniring biznes-processov na osnove sovremennyh informacionnyh tekhnologij. Sistemy upravleniya processami i znaniyami"*, Moscow, 2010, pp. 146—151 (in Russian).

30. **Kalyanov G. N.** Formal'nye metody inzhiniringa i verifikacii biznes-processov (Formal methods of engineering and verification of business processes), *Trudy 16-j nauchno-prakticheskoj konferencii "Inzhiniring predpriyatij i upravlenie znaniyami",* Moscow, 2013, pp. 144—149 (in Russian).

31. **Piterson D**. *Teoriya setej Petri i modelirovanie sistem* (The theory of Petri nets and modeling of systems), Moscow, Mir, 1984, 264 p. (in Russian).

32. **Mendling J., Reijers H., Van der Aalst W.** Seven Process Modeling Guidelines*, Information and Software Technology*, 2010, vol. 52, no. 2, pp. 127—136.

33. **Gain K., Sarson T.** *Sistemnyj strukturnyj analiz: sredstva i metody* (Structural system analysis: tools and methods), Moscow, Ejteks, 1992, 400 p. (in Russian).

34. **Kalyanov G. N.** Sistemnyj analiz trebovanij k KIS (System requirements for KIS), *Trudy 15-j Rossijskoj nauchno-prakticheskoj konferencii "Reinzhiniring biznes-processov na osnove sovremennyh informacionnyh tekhnologij. Sistemy upravleniya znaniyami"*, Moscow, 2012, pp. 96—98 (in Russian).

35. **Kupriyanov B. V.** Modelirovanie konvejernyh biznes-processov (Modelling pipelined business processes), *Sbornik trudov "Upravlenie bol'shimi sistemami"*, Moscow, 2010, vol. 28, pp. 230—273 (in Russian).

36. **Kupriyanov B. V.** Metod ehffektivnogo analiza modeli rekursivnogo konvejernogo processa (The method of effective analyze of the recursively conveyor process model), *Avtomatika i telemekhanika*, 2017, no. 3, pp. 63—79 (in Russian).

37. **Razbegin V. P., Kalyanov G. N., Kupriyanov B. V.** Sistema programmirovaniya vektornyh vychislenij (Programming system for vector computing), *Programmirovanie*, 1985, no. 4, pp. 25—32 (in Russian).

38. **Kalyanov G. N.** Testirovanie programm parallel'nyh vychislitel'nyh sistem s obshchim upravleniem (Test the software in parallel SIMD computer systems), *Programmirovanie,* 1986, no. 2, pp. 41—47 (in Russian).

УДК 681.5:519.7:519.767.4:004.58:004.823:811.93:007.51 DOI: 10.17587/prin.9.109-122

М. Ф. Степанов, д-р техн. наук, доц., проф., e-mail: mfstepanov@mail.ru,
Саратовский государственный технический университет имени Ю. А. Гагарина,
А. М. Степанов, канд. техн. наук, ст. науч. сотр., e-mail: amstepanov@mail.ru,
Институт проблем точной механики и управления Российской академии наук, г. Саратов

Адаптивный пользовательский интерфейс системы автоматизированного анализа и синтеза алгоритмов управления

Многообразие видов и форм математических моделей компонентов систем автоматического управления осложняет создание систем автоматизации анализа и синтеза алгоритмов управления. На основе принципа разделения декларативного описания математических моделей и их процедурной интерпретации предложен подход к построению адаптивного пользовательского интерфейса, реализующего автоматическую генерацию экранных форм визуализации математических моделей.

Ключевые слова: *лингвистическое обеспечение представления данных, представление данных и знаний, язык ИНСТРУМЕНТ-П, автоматическая генерация пользовательского интерфейса, автоматизация решения задач*

Введение

Проблемы создания человеко-машинного интерфейса диалогового взаимодействия пользователя с ЭВМ возникли с вместе с созданием ЭВМ. Различие способов представления данных обусловило необходимость посредника между человеком и машиной — диалогового интерфейса. Однако вначале пользователь был вынужден сам приспосабливаться к имеющимся в тот период развития вычислительной техники средствам ввода—вывода данных. Вопросы эффективности человеко-машинного взаимодействия в процессе решения задач даже не ставились. С ростом технических возможностей ЭВМ, с появлением возможности непосредственного взаимодействия пользователя с ЭВМ, минуя посредничество операторов, за счет использования пользовательских терминалов (от консоли до графических мониторов) стало возможным подстраивать человеко-машинный интерфейс в целях повышения эффективности решения задач. Исследования в этом направлении (например, [1]) выявили слабое звено в паре человек—ЭВМ. Таким звеном является человеко-машинный интерфейс, поскольку потеря эффективности при передаче информации от человека к машине происходит при необходимости преобразования ее из одной системы понятий, пользовательской (человеческой) в другую — машинную [2]. Возросшие возможности ЭВМ позволили организовывать человеко-машинный интерфейс в форме, удобной для человека. Однако расширение сфер применения компьютеров, а также включение в ряды пользователей ЭВМ широких слоев населения привели к необходимости подстройки интерфейса под каждую предметную область, под каждого пользователя. Возникло понимание необходимости адаптации к требованиям пользователя [1]. Информационные системы 1980—1990-х гг. были весьма ограничены по классам решаемых задач. Как следствие, первоначально понятие адаптации включало требования к учету предпочтений пользователя, обусловленных языковыми (лингвистическими) особенностями и уровнем квалификации пользователя. При этом все варианты реализации человеко-машинного интерфейса предусматривались разработчиком на стадии создания информационной системы. Адаптация к требованиям пользователя сводилась к запоминанию предпочитаемых конкретным пользователем форм ведения диалогового взаимодействия. Это позволяло в следующих сеансах работы исключить необходимость со стороны пользователя выбирать форму взаимодействия. Освободившееся время позволило пользователю уделить больше внимания решаемым задачам, а следовательно, повысить эффективность их решения на ЭВМ.

На современном этапе развития вычислительной техники и информатики появились новые аспекты проблемы адаптации человеко-машинного интерфейса. С ростом сложности решаемых с помощью ЭВМ задач на первый план вышли проблемы расширения классов решаемых задач, адаптации к классам задач конкретного пользователя. При этом ин-

тенсивность развития проблемных областей, расширение сфер применимости их методов привели к ситуации, когда разработчики информационных систем не успевают расширять классы решаемых задач. Возникло направление открытых систем [3], возможности которых каждый пользователь может настроить (расширить) на свой класс решаемых задач, используя встроенные средства.

Первый принцип построения открытых систем состоит в создании среды, включающей программные и аппаратные средства, службы связи, интерфейсы, форматы данных и протоколы. Такая среда опирается на развивающиеся, доступные и общепризнанные стандарты, она обеспечивает переносимость, взаимодействие и масштабируемость приложений и данных [4].

Второй принцип открытых систем состоит в использовании методов функциональной стандартизации, а именно в построении и использовании профиля — согласованного набора базовых стандартов, необходимых для решения конкретной задачи или класса задач [4].

Реализация принципов построения открытых систем осуществляется на основе технологии, включающей ряд этапов [5], именуемой технологией открытых систем (ТОС). При этом, как правило, реализуется один из двух вариантов: создается новая система; модернизируется имеющаяся система. Однако в большинстве своем современные системы по-прежнему предоставляют пользователям весьма ограниченные возможности настройки на конкретную проблемную область. Как следствие, появилось большое число программных средств, заявленных как открытые системы, однако ориентированных на весьма узкие предметные области. Широкие предметные области, например, автоматическое управление техническими объектами, интенсивно развиваются и характеризуются большим набором подлежащих решению задач, для каждой из которых разработаны и продолжают разрабатываться новые методы (процедуры) их решения, использующие нарастающее разнообразие математических моделей проектируемых систем. Разумеется, разработчики средств автоматизации не в состоянии предвидеть и воплотить компоненты пользовательского интерфейса, отражающие будущие нововведения. Как следствие, необходимы открытые системы автоматизации, снабженные адаптивным пользовательским интерфейсом, разработка которых актуальная, однако трудная задача.

Современные средства автоматизации разработки систем автоматического управления (САУ) весьма разнообразны как по классам решаемых с их помощью задач, так и по способам реализации вычислительного процесса. К таким средствам относятся, в частности, системы Экспресс-Радиус 2.1 [6], МВТУ [7], ГАММА [8—10], Инструмент-3м-И [11—14]. Среди зарубежных систем наиболее известна система MATLAB [15]. Указанные средства автоматизации предназначены для инженеров — разработчиков САУ. Требования к этим средствам сформулированы в работе [10], где предполагается, в частности, что инженер — разработчик САУ не участвует в создании средств автоматизации, а в задачах синтеза законов (алгоритмов) управления использует в качестве целей управления принятые инженерные показатели, такие как допустимые ошибки по регулируемым переменным, время регулирования, запасы устойчивости и т. д. Этим требованиям удовлетворяют, в частности, системы ГАММА-2РС и Инструмент-3м-И. Система Инструмент-3м-И содержит элементы искусственного интеллекта. Необходимость присутствия таких элементов вызвана тем, что инженеру — разработчику САУ зачастую трудно выбрать наиболее подходящую для конкретной задачи процедуру синтеза закона (алгоритма) управления. Кроме того, каждая из таких процедур (программ) имеет достаточно много разнообразных условий применимости. Их проверка может составлять самостоятельную задачу, которая осложняется необходимостью учитывать все условия до начала решения задачи, поскольку в противном случае либо решение задачи не будет получено, либо результаты ее решения могут стать неадекватными. Однако инженер, занятый текущей практической работой, может просто не знать о необходимости проверки всех условий применимости, а иногда не иметь представления и об их существовании.

Отличительной особенностью предметной области автоматического управления является большое разнообразие видов математического описания компонентов САУ [16]. При этом различные компоненты одной и той же конкретной САУ могут быть описаны различными математическими моделями. Большое разнообразие форм и видов математических моделей компонентов САУ, сложность задач проектирования и исследования систем автоматического управления как с алгоритмической, так и с вычислительной точек зрения, обусловила необходимость широкого применения средств автоматизации для решения таких задач [17, 18].

Проблема создания универсальных (охватывающих широкий класс задач) средств автоматизации решения задач управления осложняется рядом факторов. К ним относится расширение классов управляемых объектов, которые описываются все более сложными математическими моделями. Как следствие, это приводит к развитию собственно теории автоматического управления, к созданию новых методов для решения новых задач, а также к расширению категорий заинтересованных в использовании этих методов инженеров-разработчиков, а следовательно, снижение, в среднем, их квалификации собственно в теории автоматического управления.

Большинство методов теории автоматического управления (ТАУ) характеризуются достаточно ограниченными областями их применимости, которые задаются рядом условий, накладываемых на вид математической модели объекта управления, на параметры, форму и вид цели управления (критерия

точности и качества управления), на форму и вид синтезируемой модели управляющего устройства и т. д. На современном этапе развития ТАУ не удается создать универсальные средства, обеспечивающее автоматизацию решения широкого класса задач проектирования и исследования систем управления объектами произвольной физической природы. Вместе с тем востребованность в таких средствах издавна привлекала многих ученых-исследователей и инженеров-практиков и привела к созданию многочисленных пакетов программ, позволяющих автоматизировать решение ряда задач, представляющих собой "*...отдельные островки в безбрежном океане задач...*" [19].

Однако пользователя — разработчика систем автоматического управления больше привлекают открытые системы, обеспечивающие гибкость настройки (адаптации) классов решаемых задач за счет включения создаваемых пользователем процедур решения новых задач, использующих новые математические модели новых классов объектов управления и компонентов систем управления. Здесь термин "*новый*" означает "*неизвестный ранее*", который не был *предусмотрен разработчиком системы автоматизации*. Однако "прокрустово ложе" предопределенных форм визуализации данных пользовательского интерфейса существенно ограничивает возможности.

Для разрешения отмеченной выше проблемы необходимо решить следующие задачи.

- Задача 1. Разработать гибкие средства описания математических моделей различных видов и форм, включая задаваемые пользователем — разработчиком систем управления оригинальные (новые) математические модели компонентов систем управления в привычном векторно-матричном виде.
- Задача 2. Разработать адаптивные средства организации пользовательского интерфейса, обеспечивающие возможности визуализации (отображения, редактирования) произвольных задаваемых пользователем математических моделей компонентов систем управления в привычных для пользователя — разработчика систем управления представлениях, включая формульные математические описания.

В данной работе рассмотрен подход к решению поставленных задач. Соответствующие средства реализованы в системе ГАММА-3 [20].

1. Математическая модель предметной области теории управления

Концептуально метод решения отмеченной выше проблемы базируется на разделении знаний предметной области теории управления на декларативное описание математических моделей компонентов систем управления и их процедурную интерпретацию проектными операциями, используемыми в процедурах анализа и синтеза систем управления.

В работе [12] предложено используемое далее формализованное представление знаний теории управления (модель предметной области):

$$M = \langle D, \Re, O \rangle, \tag{1}$$

где $D = \{d_i \mid d_i = \langle \Im(d_i), \wp(d_i), \aleph(d_i), \Xi(d_i) \rangle\}$ — множество формализованных обобщений математических моделей компонентов систем управления, называемых "*предметами*", обладающих:

- *свойствами* $\rho_j \in \wp(d_i) = \{\rho \mid \rho \in \{\text{true} \mid \text{false}\}\}$,
- *характеристиками* $\chi_j \in \aleph(d_i) = \{\chi_{k_i} \mid \chi_{k_i} \in C^{N_{k_i}} \times = \times C^{N_{k_i}} \times ... \times C^{N_{k_i}}\}$, где C — множество комплексных чисел; N_{k_i} — размерность характеристики χ_i;
- *формами математических моделей* $\mu_j \in \Im(d_i) = \{\mu_1, ..., \mu_{\tau_i}\}$, где $\mu_j = \langle r_j, m_j \rangle$, $r_j = \{r_{j1}, r_{j2}, ..., r_{jk} \mid r_{jl} \in N\}$ — множество размерностей компонентов (матриц) математической модели μ_j, $m_j = \{m_{jk} \mid m_{jk} \in C^{r_{jx1}} \times C^{r_{jk2}} \times ... \times C^{r_{jkl}}\}$ — множество коэффициентов компонентов (матриц m_{jk}) математической модели μ_j;
- классификационными *признаками*, представляющими собой выделенные свойства предметов $\xi_j \in \Xi(d_i) \subseteq \wp(d_i)$;

$\Re = \{r \mid r = \langle c, s, g \rangle : \wp \cup \aleph \cup \Im \cup \Re \cup O \to \{\text{true} \mid \text{false}\}\}$ — множество *отношений* (предикатов) над предметами, их компонентами и атрибутами, действиями (операциями);

$O = \{o \mid o = \langle c, s, g, q \rangle : \wp \cup \aleph \cup \Im \cup \Re \to \wp \cup \aleph \cup \Im \cup \Re\}$ — множество *действий* (*операций*) над предметами и их атрибутами.

Действия $o_i = \langle c_i, s_i, g_i, q_i \rangle \in O$ и отношения $r_i = \langle c_i, s_i, g_i \rangle \in \Re$ характеризуются своими атрибутами: $c_i \in \wp \cup \Re$ — условия применимости; $s_i \in \wp \cup \aleph \cup \Im$ — исходные данные; $g_i \in \wp \cup \aleph \cup \Im \cup \Re$ — результат выполнения действия (операции); $q_i \in \Re$ — требования к результату выполнения действия (операции).

2. Исследование возможности представления математических моделей компонентов систем управления в выбранном формализме

Исследуем возможности принятого способа формализации знаний в виде модели (1) для представления различных математических моделей компонентов систем управления. Рассмотрим наиболее распространенные математические модели компонентов систем управления. В связи с простотой таких атрибутов предметов, как "свойство", опустим их из рассмотрения. Несколько больший интерес представляют характеристики предметов. К ним относятся как скалярные величины, так и массивы значений вещественного или комплексного типа. Для скалярных характеристик, например, для установившейся ошибки системы управления с одним входом и одним выходом

$$y_{st} = \lim_{t \to \infty} y(t) \tag{2}$$

общее представление характеристики $\chi_j \in \aleph(d_i) = \left\{\chi_{k_i} \mid \chi_{k_i} \in C^{N_{k_i}} \times C^{N_{k_i}} \times ... \times C^{N_{k_i}}\right\}$ получит конкретизацию $y_{st} \doteq \chi_j \in \aleph(d_i) = \left\{\chi_{k_i} \mid \chi_{k_i} \in C\right\}$, где j — номер характеристики предмета d_i, а символ $\doteq$ есть "обозначение". К характеристикам, значениями которых являются массивы чисел, например, собственные значения (корни) характеристического полинома замкнутой системы управления $\lambda = \left\{\lambda_1, ..., \lambda_n \mid \lambda_i \in C\right\}$, применимо описание вида $\lambda \doteq \chi_j \in \aleph(d_i) = \left\{\chi_{k_i} \mid \chi_{k_i} \in C^n\right\}$.

Распространяя представление $\chi_j \in \aleph(d_i) = \left\{\chi_{k_i} \mid \chi_{k_i} \in C^{N_{k_i}} \times C^{N_{k_i}} \times ... \times C^{N_{k_i}}\right\}$ на многомерный случай (несколько входов и выходов системы управления), выберем к рассмотрению значения частот среза по каждому каналу управления (возмущения)

$$\omega_{cp} = \left\{\omega_{cp_{11}}, \omega_{cp_{12}}, ...\omega_{cp_{1r}}, \omega_{cp_{21}}, \omega_{cp_{22}}, ..., \omega_{cp_{2r}}, ..., \omega_{cp_{m1}}, \omega_{cp_{m2}}, ..., \omega_{cp_{mr}} \mid \omega_{cp_{ij}} \in R \subset C\right\}, \quad (3)$$

где $\omega_{cp_{ij}}$ — частота среза частотной характеристики [21], построенной по передаточной функции от i-го входа к j-му выходу.

Тогда в рамках используемой модели (1) данная характеристика получит представление

$$\omega_{\text{ср}} = \chi_j \in \aleph(d_i) = \left\{\chi_{k_i} \mid \chi_{k_i} \in C^m \times C^r\right\}.$$

Перейдем к рассмотрению представления форм математических моделей.

Начнем с частотной передаточной функции:

$$W(j\omega) = \frac{\Sigma_{i=0}^{m} a_i (j\omega)^i}{\Sigma_{i=0}^{n} d_i (j\omega)^i}. \quad (4)$$

С точки зрения представления данных для задания передаточной функции (4) необходимо указать коэффициенты полиномов числителя и знаменателя, а также их размерности: $\mu_1 = \langle r_1, m_1 \rangle$, $r_1 = \left\{r_{11}, r_{12} \mid r_{1l} \in N\right\}$, $r_{11} = m + 1$, $r_{12} = n + 1$, $m_1 = \left\{\langle a, d \rangle \mid a \in R^{r_{11}}, d \in R^{r_{12}}\right\}$.

Далее рассмотрим популярную форму представления математической модели динамической системы в пространстве состояний [16]:

$$\dot{x} = Ax + Bu,\ x \in R^{r_x},\ u \in R^{r_u}, \quad (5)$$

$$y = Cx + Du, y \in R^{r_y}. \quad (6)$$

Аналогично и в этом случае, необходимо задать размерности векторов x, u, y и коэффициенты соответствующих матриц A, B, C, D. Тогда математическая модель (5), (6) в модели представления знаний (1) примет следующий вид:

$$\mu_2 = \langle r_2, m_2 \rangle,\ r_2 = \left\{r_{21}, r_{22}, r_{23} \mid r_{2l} \in N\right\},$$
$$r_{21} = r_x,\ r_{22} = r_u,\ r_{23} = r_y,$$
$$m_1 = \left\{\langle a, b, c, d \rangle \mid a \in R^{r_{21}} \times R^{r_{21}},\right.$$
$$\left. b \in R^{r_{21}} \times R^{r_{22}}, c \in R^{r_{23}} \times R^{r_{21}}, d \in R^{r_{23}} \times R^{r_{22}}\right\}.$$

И так далее для всех видов математических моделей, необходимых для решения выбранного класса задач теории управления.

Приведенные примеры показывают, что математические модели компонентов систем (объектов) управления в векторно-матричной форме представления включают в качестве базовых элементов целочисленные значения, массивы вещественных и комплексных чисел, которые сопровождаются поясняющим текстом и математическими формулами. Таким образом, базовые конструкции построения пользовательского интерфейса должны включать средства визуализации (редактирования) многомерных массивов, размерности которых задаются целочисленными значениями. Как следствие, средства построения человеко-машинного интерфейса представления (визуализации, редактирования) математических моделей систем управления сводятся к построению интерпретатора (процедуры) формируемых пользователем декларативных описаний математических моделей. Такой интерпретатор призван обеспечить построение адекватных визуальных форм.

3. Подход к созданию средств визуализации математических моделей компонентов систем управления

В настоящее время наиболее распространенной (даже доминирующей) формой организации интерактивного взаимодействия с пользователем является оконный интерфейс. При этом поля экранной формы могут содержать информацию различного вида: поясняющая информация, исходные данные (поля ввода данных), результаты (поля вывода данных), управляющие компоненты (кнопки, переключатели и т. д.). Собственно поля ввода и вывода данных также различаются в зависимости от представляемой информации (текстовая, числовая, скалярная, массивы, графики, рисунки и т. д.). Основные требования к организации визуального представления информации могут быть сформулированы следующим образом:

- представление информации на экранной форме должно быть компактным, но обеспечивающим необходимый уровень детальности;
- структура отображения на экранной форме должна соответствовать структуре представляемой информации;
- выводимая информация должна иметь пояснение подходящего по контексту вида (комментарий, всплывающая подсказка, формула, рисунок и т. д.).

Для реализации перечисленных требований необходимо разработать внешнюю (пользовательскую) и внутреннюю (машинную) формы декларативного представления информации и разработать алгоритм генерации экранной формы пользовательского интерфейса.

В работе [22] предложен язык ИНСТРУМЕНТ+, представляющий собой язык описания моделей вида (1) и задач на них. Его развитие [23, 24], реализованное в системе ГАММА-3 [20], получило наименование ИНСТРУМЕНТ-П. Модель вида (1) на языке ИНСТРУМЕНТ-П является фреймовой моделью предметной области [25], в которой:

- фреймы представляют собой *абстрактные* модели понятий ТАУ, используемые для целей автоматизации решения задач проектирования САУ;
- фреймы сгруппированы в подмодели (*модули*) в соответствии с семантикой предметной области ТАУ;
- фреймы представляют собой *иерархически* взаимосвязанную совокупность с наследованием потомками компонентов предков.

Описание модели вида (1) на языке ИНСТРУМЕНТ-П представляется в виде конструкции, которая именуется МОДЕЛЬ.

Конструкция МОДЕЛЬ снабжается уникальным именем и включает описания *предметов*, *действий* (операций), выполняемых над предметами и их компонентами, *отношений*, устанавливаемых между предметами, их компонентами, операциями.

ПРЕДМЕТЫ идентифицируются именем и типом и могут иметь:

- КОМПОНЕНТЫ — как правило, другие предметы МОДЕЛИ, входящие в состав данного предмета;
- СВОЙСТВА — атрибуты предмета, принимающие логические значения и отражающие существенные особенности данного предмета;
- ХАРАКТЕРИСТИКИ — атрибуты, характеризующие предмет, значения которых могут использоваться для оценки поведения предмета;
- ПРОЦЕССЫ — отражение изменения элементов предмета в зависимости от некоторого параметра (например, времени);
- ФОРМЫ — описание форм математической модели предмета.

ДЕЙСТВИЯ (операции) идентифицируются уникальным именем и имеют следующие атрибуты:

- ТИП_ДЕЙСТВИЯ — отражает классификацию действий в предметной области ТАУ. Выделяют четыре типа действия: анализ свойств, вычисление характеристик, преобразование форм предметов, синтез предметов;
- АРГУМЕНТЫ — список имен предметов, их свойств и характеристик, выступающих в качестве исходных данных действия (операции);
- РЕЗУЛЬТАТ — список имен предметов, их свойств и характеристик, формируемых в результате выполнения действия;
- ПЛАН — последовательность операторов (процедура) языка ИНСТРУМЕНТ-П;
- УСЛОВИЯ_ПРИМЕНИМОСТИ — логическое выражение, в котором используются имена свойств предметов, отношения; истинное значение выражения позволяет выполнить данную операцию.

ОТНОШЕНИЯ — специальный вид действий, возвращаемое значение которых всегда является логическим. Тип отношения задается его именем. Например, БОЛЬШЕ(x, y) — принимает значение ИСТИНА, если $x > y$. Число аргументов отношения определяется его типом.

Структура записи модели на языке ИНСТРУМЕНТ-П:

```
МОДЕЛЬ имя _ модели
   (ПРЕДМЕТ имя [:тип];наименование
    ([КОМПОНЕНТЫ
      (имя [:тип];наименование(описание) имя _ действия)...]
     [СВОЙСТВА
      (имя [:тип];наименование(описание) имя _ действия)...]
     [ХАРАКТЕРИСТИКИ
      (имя [:тип];наименование(описание) имя _ действия)...]
     [ПРОЦЕССЫ
      (имя [:тип];наименование(описание) имя _ действия)...]
     [ФОРМЫ
      (имя [:тип];наименование(описание) имя _ действия)...] ...
    (ДЕЙСТВИЯ
      (имя [:тип];наименование
        (ИСХОДНЫЕ ДАННЫЕ (список _ имен _ данных)
         РЕЗУЛЬТАТ (список _ имен _ данных)
         ПЛАН (операторы _ языка _ ИНСТРУМЕНТ+)) ...)
    (ОТНОШЕНИЯ
      (имя [:тип];наименование
    (ИСХОДНЫЕ ДАННЫЕ (список _ имен _ данных)
     ПЛАН (операторы _ языка _ ИНСТРУМЕНТ+)) ...)
```

Рис. 1. Блок-схема алгоритма генерации экранной формы пользовательского интерфейса представления данных на основе декларативного описания на языке ИНСТРУМЕНТ-П

Здесь в квадратных скобках, используемых в качестве метасимволов, указаны необязательные конструкции, которые могут отсутствовать в описаниях конкретных элементов модели. Подчеркиванием для наглядности выделены ключевые слова языка. Каждый компонент модели может сопровождаться поясняющей текстовой информацией, а также и формульным представлением математических моделей (используется нотация LaTeX).

Укрупненная блок-схема алгоритма работы интерпретатора (процедуры генерации пользовательского интерфейса) описаний моделей (1) на языке ИНСТРУМЕНТ-П представлена на рис. 1.

4. Реализация средств представления данных в системе ГАММА-3

Система ГАММА-3 является развитием систем ГАММА-1РС, ГАММА-2РС [8—10] и Инструмент-3м-И [11—14]. Такое объединение позволило создать более гибкое и функционально более полное средство автоматизации решения задач синтеза и анализа законов (алгоритмов) управления.

Система ГАММА-3 позволяет решать задачи как в традиционной процедурной, так и в непроцедурной (декларативной) постановках.

Для процедурно поставленной задачи последовательность ее решения задается в виде программы на проблемно-ориентированном языке ГАММА [26]. В связи с тем, что в современной теории управления наиболее распространено векторно-матричное представление математических моделей систем управления, именно матрица в языке ГАММА выбрана в качестве базового типа данных по умолчанию. Как следствие, синтаксис языка ГАММА близок к синтаксису входного языка системы MATLAB, учитывая популярность последней в среде специалистов в теории управления. Следовательно, для освоения языка ГАММА не потребуется дополнительных усилий. Реализованные в системе ГАММА-3 возможности решения процедурно поставленных задач ориентированы на ученых-исследователей, разрабатывающих новые методы синтеза и анализа САУ. Пример решения процедурно поставленной задачи синтеза и анализа САУ представлен на рис. 2.

Для декларативно поставленных задач используются возможности, унаследованные от системы ИНСТРУМЕНТ-3м-И [11—14]. Постановка задачи осуществляется на модели представления знаний вида (1) посредством указания атрибутов задачи: "исходные данные", "искомый результат", "требования, предъявляемые к искомому результату". Состав атрибутов задается в интерактивном режиме посредством выбора необходимых наименований (имен) предметов, их свойств, характеристик, форм математических моделей, входящих в модель знаний (1). Каждый пользователь-исследователь может создавать собственные модели знаний, разрабатывая новые методы решения задач управления, либо использовать базовые, созданные разработчиками системы ГАММА-3, для решения "типовых" задач в декларативной постановке. В качестве планирующей подсистемы решения декларативно поставленных задач используются планирующие искусственные нейронные сети (ПИНС). В связи с этим обстоятельством для представления планов решения задач разработан язык ИНСТРУМЕНТ-ОП [12], поддерживающий парадигму "Ориентированный на правила". На рис. 3—5 представлены фрагменты экранных форм, иллюстрирующих работу с моделями представления знаний, декларативной формой постановки задач, протоколом решения задач.

Система ГАММА-3 непрерывно развивается авторским коллективом разработчиков. Развитие осуществляется как в техническом, так и в содержательном направлениях. В архитектурном плане система ГАММА-3 открыта к расширениям. Это обстоятель-

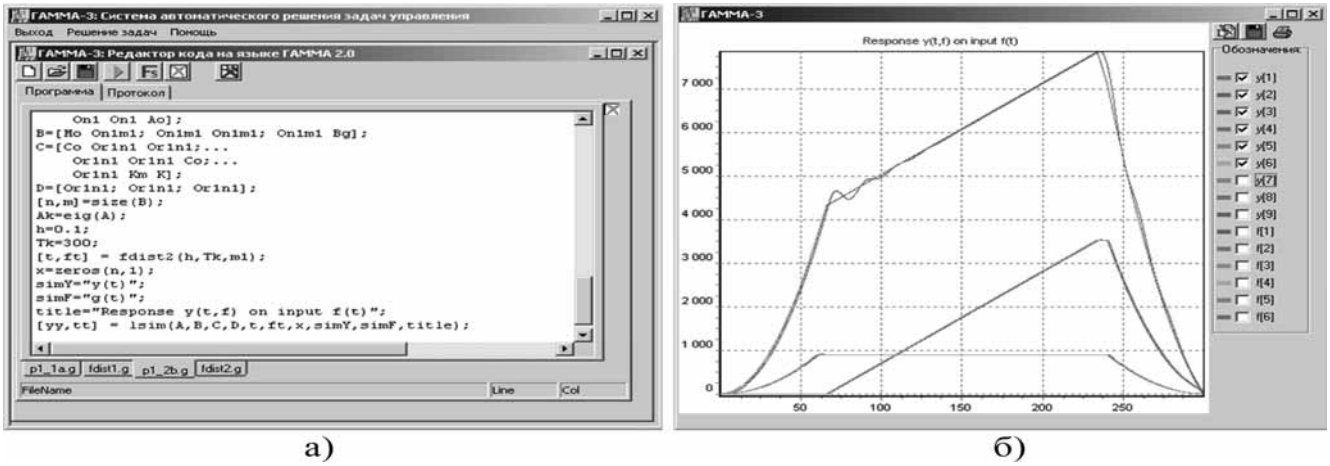


Рис. 2. Пример решения процедурно поставленной задачи синтеза и анализа САУ:
а — программа на языке ГАММА; *б* — графики переходных процессов

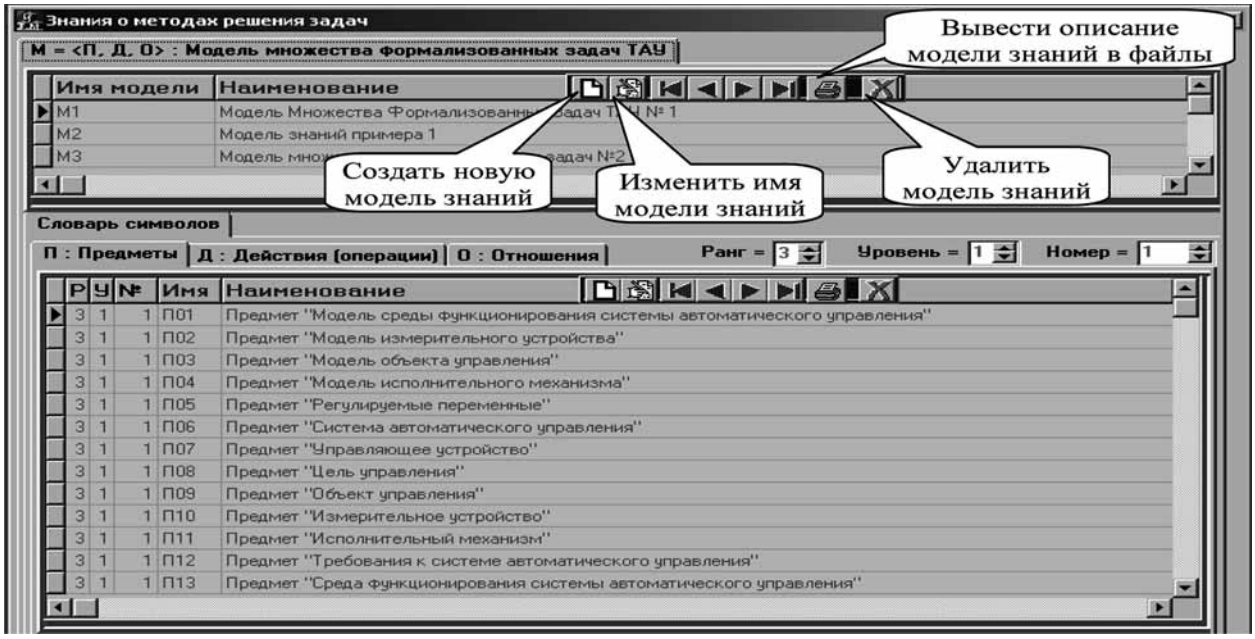


Рис. 3. Вид экранной формы визуализации компонентов модели представления знаний (Предметы)

ство позволило создать как локальный, так и распределенный (функционирующий в виде совокупности подсистем, реализованных в виде отдельных серверов на различных узлах вычислительной сети) варианты организации системы.

С технической точки зрения система ГАММА-3 в настоящее время существует в трех модификациях:

1) *локальная* версия, рассчитанная на ограниченную группу пользователей и устанавливаемая локально на персональный компьютер;

2) *клиент-серверная* версия, ориентированная на использование в локальной вычислительной сети предприятия;

3) *облачный* вариант реализации [27], предоставляющий возможности решения задач с использованием доступа к системе через Интернет.

Каждый из перечисленных вариантов реализации обладает равноценными возможностями по решению как процедурно, так и декларативно поставленных задач. С содержательной точки зрения осуществляется развитие возможностей в направлении расширения классов решаемых задач, включенных в базовую комплектацию.

Пользовательский интерфейс каждого варианта реализации системы ГАММА-3 с проектировщиком систем управления предоставляет возможности

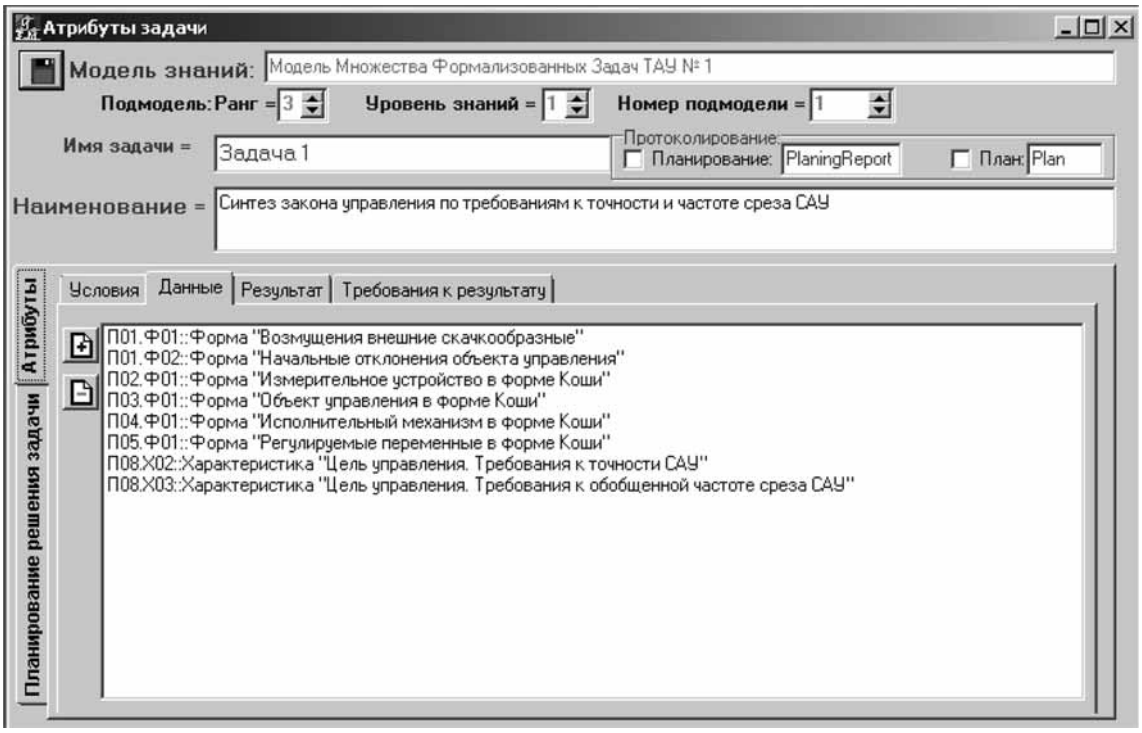


Рис. 4. Вид экранной формы визуализации атрибутов задач на модели представления знаний (Задача 1 на модели М1)

подготовки исходных данных, постановки задачи с использованием подготовленных данных и визуализированных результатов решения задач. Использование процедурного подхода представления знаний о методах решения задач [12] (элементарные операции в виде программных модулей и процедуры решения задач в виде программ на языке ГАММА [26]) обусловило и процедурную интерпретацию математических моделей компонентов систем управления проектными операциями, используемыми в ходе решения задач. Как следствие, декларативное описание математических моделей и их процедурную интерпретацию стало возможным разделить и создать средства автоматической генерации экранных форм визуализации математических моделей компонентов проектируемых систем управления.

В системе ГАММА-3 принят иерархический подход к хранению информации (рис. 6). Главной единицей хранения информации в системе ГАММА-3 является структура, именуемая "Проект". В состав проекта входят:

- *предметы*, являющиеся формализованной абстракцией систем автоматического управления и их компонентов (объекты управления, управляющие устройства, измерительные устройства, исполнительные механизмы, цели управления, внешняя среда и т. д.), общедоступные для использования в разрабатываемых в рамках проекта системах управления;
- *системы управления* — совокупность описаний разрабатываемых систем управления и связанных с ними компонентов и подсистем, представляющих собой конкретизацию формальных описаний, доставляемых предметами;
- *процедуры* — специализированные программы решения задач проектирования и исследования разрабатываемых систем управления конкретного проекта;
- *модели знаний* — формализованные описания знаний о методах решения задач проектирования и исследования систем автоматического управления, построенные при разработке конкретного проекта, обеспечивающие возможность решения задач в непроцедурной постановке и не включенные в общедоступную для всех пользователей базу знаний системы.

В целях обеспечения гибкости представления данных в системе ГАММА-3 разработан язык описания структур данных и знаний ИНСТРУМЕНТ-П. Описание данных (фактов в терминологии системы ГАММА-3) на языке ИНСТРУМЕНТ-П позволяет автоматически построить экранную форму, обеспечивающую адекватную визуализацию. Для обеспечения большей гибкости используется LaTeX-подобный язык описания математических моделей (формул) компонентов предметов.

Проиллюстрируем средства визуализации данных системы ГАММА-3 на примере предмета "Объект управления", фрагмент описания формы математической модели которого имеет следующий вид:

```
  Ф02 "форма вход-выход"
     рисунок "T1(s)*y=T2(s)*u+T3(s)*f,y\in R^r,u\in R^m,f\in
R^q \newrow T1(s)=\sum^{g _ y} _ {i=0}T1 _ {i}s^i,
T2(s)=\sum^{g _ u} _ {i=0}T2 _ {i}s^i, T3(s)=\sum^{g _ f} _ {i=0}T3 _ {i}s^i"
     целый r "размерность вектора выходных переменных" =1,
     целый m "размерность вектора управлений" =1,
     целый q "размерность вектора возмущений" =1,
     целый gy "степень полинома при выходных переменных" =0:3,
     целый gu "степень полинома при управлениях" =0:0,
     целый gf "степень полинома при возмущениях" =0:0,
     вещественный T1[1:r;1:r;0:gy] "полином при выходных
переменных" {1 0 0 1 1.1 0 0 3.1 2.7 0 0 12.5 1.3 0 0 3.1},
     вещественный T2[1:r;1:m;0:gu] "полином при управлениях"
{20 12 2 120 2.1 1.1 2.1 21},
     вещественный T3[1:r;1:q;0:gf] "полином при возмущениях"
{3 1 0.1 30}
```

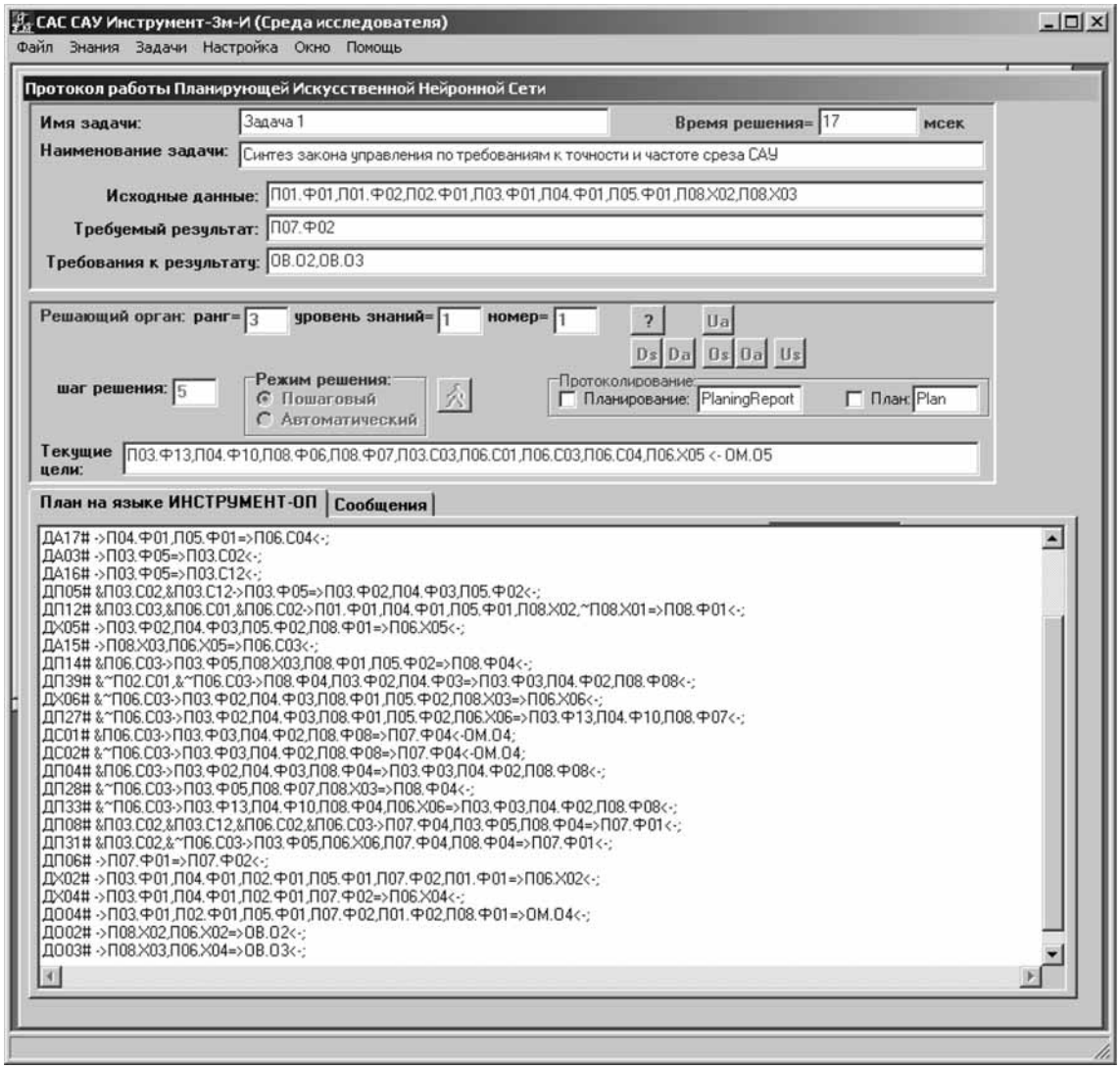


Рис. 5. Результат планирования решения задачи — план на языке ИНСТРУМЕНТ-ОП

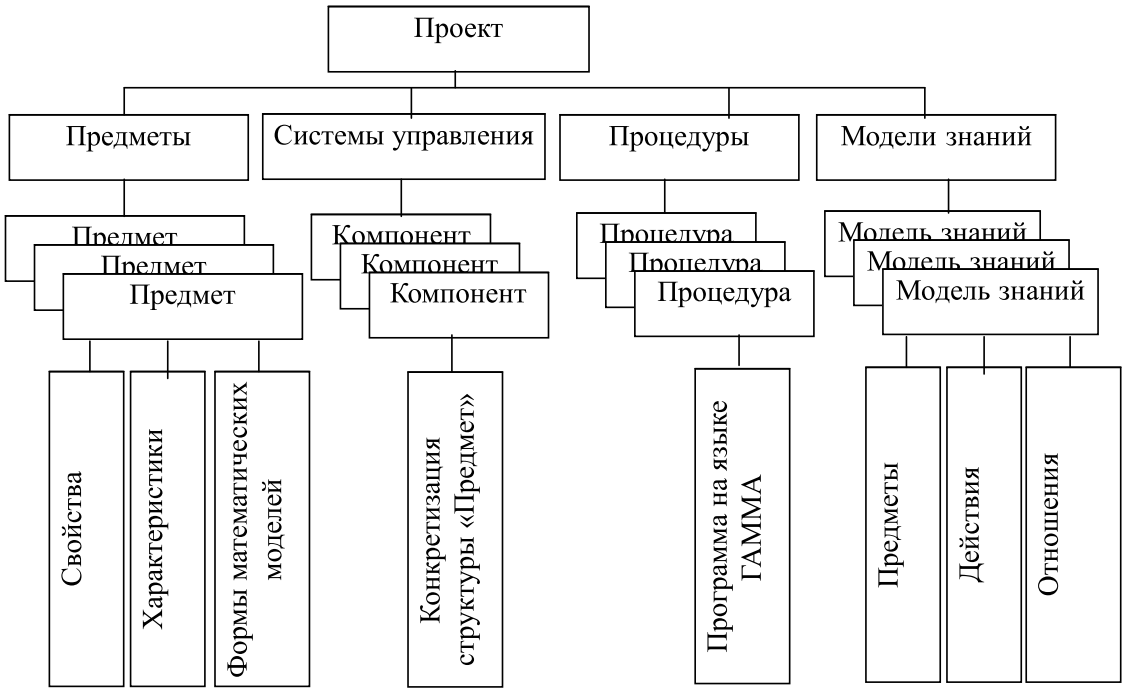


Рис. 6. Состав структуры "Проект"

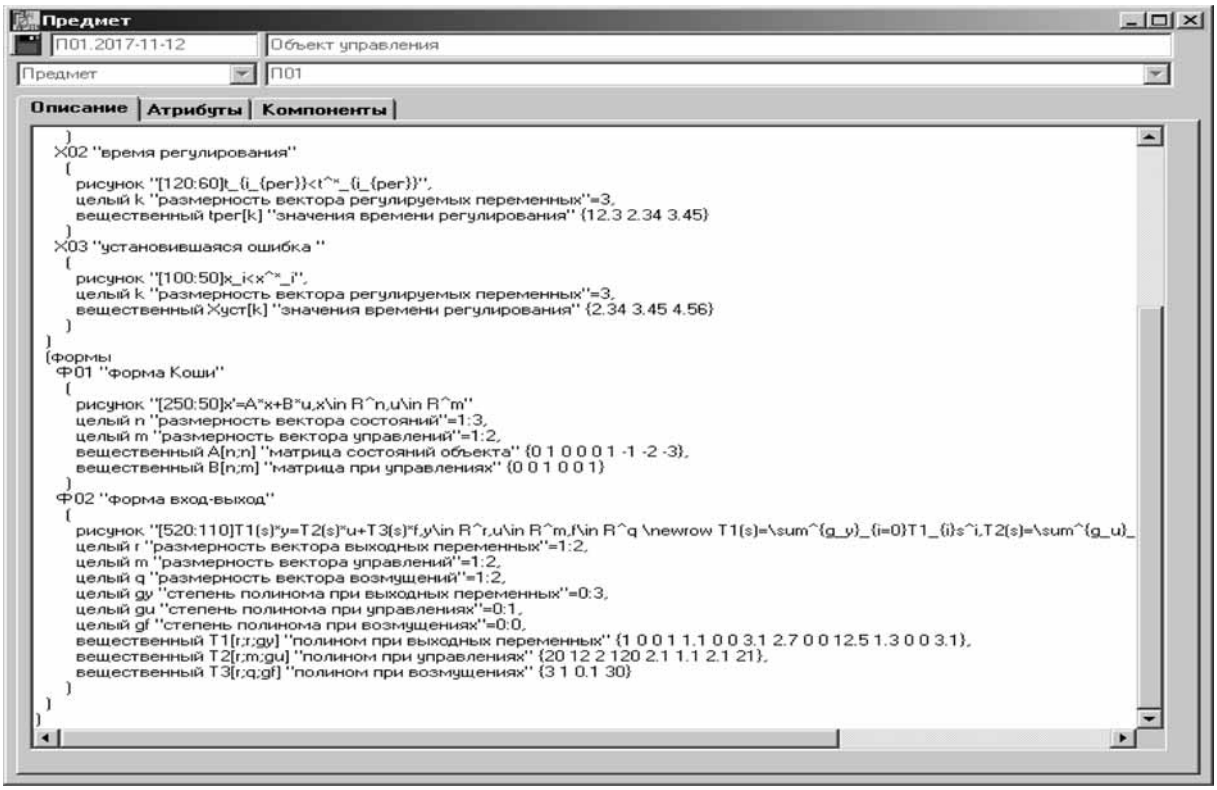


Рис. 7. Текстовое описание предмета

Рис. 8. Визуализация формы "Вход — выход" предмета "Объект управления" с именем П01.2017-11-12

На рис. 7 представлена экранная форма визуализации текстового (внутреннего) описания предмета "Объект управления" на языке ИНСТРУМЕНТ-П.

Адаптивный алгоритм построения пользовательского интерфейса визуализации данных реализуется [28] транслятором [26] текстового описания на языке ИНСТРУМЕНТ-П в визуальные формы компонентов пользовательского интерфейса (функции PredmetTextToForm(), doLexAnalis(), doSyntaxAnalisPredmet(), PredmetToForm()).

На рис. 8 представлена автоматически сгенерированная экранная форма визуализации предмета "Объект управления" по текстовому (см. приведенный выше фрагмент) описанию на языке ИНСТРУМЕНТ-П.

Заключение

На основе принципа разделения декларативного описания математических моделей и их процедурной интерпретации в системе ГАММА-3 реализованы средства представления понятий теории управления, позволяющие использовать выбранную модель формализации знаний, объединить внешнюю, удобную для пользователя и внутреннюю, удобную для представления в ЭВМ формы описания математических моделей компонентов систем управления, автоматически генерировать экранные формы пользовательского интерфейса в привычном для специалистов в теории управления виде.

В системе ГАММА-3 для обеспечения гибкости, автоматической подстройки (адаптивности) пользовательского интерфейса разработаны универсальные динамические структуры внутреннего представления данных в виде системы взаимосвязанных классов языка C++. Их формирование осуществляется автоматически встроенным транслятором по описаниям, задаваемым пользователем на языке ИНСТРУМЕНТ-П. Благодаря такому подходу пользователь помогает системе подстроить (адаптировать) под собственные предпочтения человеко-машинный интерфейс. Таким образом, предлагаемый подход позволяет реализовать адаптивность пользовательского интерфейса, а также открытый к расширениям системы характер ГАММА-3.

Это осуществляется посредством автоматизированного подключения к системе библиотек динамической загрузки, включающих программные реализации новых проектных операций (действий) модели предметной области теории управления для обеспечения возможности решения новых задач. Вопросы реализации данных возможностей рассматриваются в отдельной статье.

Дальнейшая работа направлена на упрощение создания новых проектных операций. Пользователю не потребуется обращаться к средствам разработки

программного обеспечения. В дальнейшем разработка программного обеспечения проектных операций будет осуществляться в рамках системы ГАММА-3.

Работа выполнена при поддержке РФФИ (проекты 13-07-00647, 15-07-99684).

Идея создания ориентированных на инженеров-проектировщиков систем автоматизации решения задач разработки и исследования алгоритмов управления принадлежит профессору Альберту Георгиевичу Александрову, Учителю и Человеку с большой буквы. Работы в данном направлении продолжаются благодарными учениками. Памяти Альберта Георгиевича посвящается.

Список литературы

1. **Денинг В., Эссиг Г., Маас С.** Диалоговые системы "человек — ЭВМ". Адаптация к требованиям пользователя. Пер. с англ. М.: Мир, 1984. 112 с.
2. **Попов Э. В.** Экспертные системы: Решение неформализованных задач в диалоге с ЭВМ. М.: Наука, 1987. 288 с.
3. **Батоврин В. К., Васютович В. В., Гуляев Ю. В.** и др. Технология открытых систем / Под ред. А. Я. Олейникова. М.: Янус-К, 2004. 288 с.
4. **TECHNICAL REPORT** ISO/IEC TR 14252 ANSI/IEEE Std 1003.0 First edition 1996-12-15 Information technology — Guide to POSIX ® Open System Environment (OSE).
5. **Гуляев Ю. В., Олейников А. Я.** Открытые системы: от принципов к технологии // Информационные технологии и вычислительные системы. 2003. № 3. С. 6—11.
6. **Дорри М. Х., Кочемасов А. В., Рощин А. А.** Автоматизация проектирования систем и средств управления / Пакет "Экспресс-Радиус 2.1". Уч. пос. М.: Изд-во МИРЭА, 2000. 106 с.
7. **Козлов О. С., Кондаков Д. Е., Скворцов Л. М.** и др. Программный комплекс для исследования динамики и проектирования технических систем // Информационные технологии. 2005. № 9. С. 18—25.
8. **Alexandrov A. G., Panin S. Yu., Stepanov M. F.** CADSD GAMMA-1PC as processor for controllers // IFAC YAC'95: International Federation of Automatic Control Youth Automatic Conference. Beijing P. R. China: Chinese Association of Automation (CAA) Youth Committee of CAA Beijing Institut of Technology (BIT), 1995. Vol. II. P. 845—849.
9. **Александров А. Г., Панин С. Ю.** Система ГАММА-1РС для синтеза регуляторов многомерных систем // Автоматизация в промышленности. 2003. № 3. С. 18—22.
10. **Александров А. Г., Михайлова Л. С., Исаков Р. В.** Структура программного обеспечения для автоматизации разработки алгоритмов автоматического управления // Автоматика и телемеханика. 2005. № 4. С. 176—184.
11. **Степанов М. Ф.** Анализ и синтез систем автоматического управления в программной среде "ИНСТРУМЕНТ-3м-И" // Известия высших учебных заведений. Приборостроение. 2004. Т. 47. № 6. С. 27—30.
12. **Степанов М. Ф.** Автоматическое решение формализованных задач теории автоматического управления. Монография. Саратов: Сарат. гос. техн. ун-т, 2000. 376 с.
13. **Степанов М. Ф.** Планирующие искусственные нейронные сети в самоорганизующихся интеллектуальных системах управления // Докл. Росс. акад. естеств. наук. Поволж. межрегион. отделение. 1999. № 1. С. 73—99.
14. **Степанов М. Ф.** Нейронные сети для планирования решения задач теории автоматического управления // Проблемы управления. 2004. № 2. С. 66—71.
15. **MATLAB** User's Guie. Math Works. 2001.
16. **Справочник** по теории автоматического управления / Под ред. А. А. Красовского. М.: Наука, 1987. 712 с.
17. **Frederick D. K., Kraft R. P., Sadeghi T.** Computer-aided control system analysis and design using interactive graphics // IEEE Control Systems Magazine. 1982. December. P. 19—25.
18. **Автоматизированное** проектирование систем управления / Под ред. М. Джамшиди, Ч. Дж. Чергета; Пер. с англ. В. Г. Дунаева и А. Н. Косилова. М.: Машиностроение, 1989. 344 с.
19. **Александров А. Г.** О принципах построения системы анализа динамики и синтеза устройств управления (САПР САУ) // Аналитические методы синтеза регуляторов: Межвуз. науч. сб. Саратов: Сарат. политехн. ин-т, 1982. С. 123—136.
20. **Александров А. Г., Михайлова Л. С., Степанов М. Ф.** Система ГАММА-3 и ее применение // Автоматика и телемеханика. 2011. № 10. С. 19—27.
21. **Бесекерский В. А., Попов Е. П.** Теория систем автоматического регулирования. М.: Наука, 1975. 767 с.
22. **Степанов М. Ф.** Объектно-ориентированная модификация языка интерпретации структурных схем процедур методов синтеза систем автоматического управления (ИНСТРУМЕНТ+) // Методы и средства управления технологическими процессами МСУТП 99. Сб. трудов Третьей междунар. науч. конференции. Саранск: Изд-во Мордовского ун-та, 1999. С. 300—303.
23. **Брагин Т. М., Степанов М. Ф., Степанов А. М.** Комплексный подход к автоматизации проектирования и аппаратной реализации интеллектуальных систем управления средствами многофункциональной системы ГАММА-3 // Шестая Всероссийская мультиконференция по проблемам управления. Материалы мультиконференции: в 4 т. Ростов-на-Дону: Изд-во Южного федер. ун-та, 2013. Т. 1. С. 116—120.
24. **Степанов М. Ф., Степанов А. М.** Принципы построения, архитектура средств проектирования, моделирования и исследования интеллектуальных систем управления // Восьмая Всероссийская мультиконференция по проблемам управления. Материалы 8-й Всероссийской мультиконференции: в 3 т. Ростов-на-Дону: Изд-во Южного федер. ун-та. 2015. Т. 1. С. 109—112.
25. **Минский М.** Фреймы для представления знаний / Пер. с англ. М.: Энергия, 1979. 152 с.
26. **Степанов М. Ф., Михайлова Л. С.** Интегрированная среда проектирования систем управления Гамма: реализация входного языка // Интеллектуальные системы: Труды Девятого междунар. симпозиума / Под ред. К. А. Пупкова. М.: РУСАКИ, 2010. С. 191—193.
27. **Александров А. Г., Михайлова Л. С., Брагин Т. М.** и др. Аспекты применения облачных технологий для автоматизации решения задач проектирования систем управления в системе "ГАММА-3" // Системы проектирования, технологической подготовки производства и управления этапами жизненного цикла промышленного продукта (CAD/CAM/PDM — 2012). Труды 12-й Международной конференции / Под ред. Е. И. Артамонова. М.: ООО "Аналитик". 2012. С. 111—115.
28. **Александров А. Г., Михайлова Л. С., Степанов М. Ф.** и др. О развитии концепции автоматического решения задач теории управления в системе ГАММА-3 // Мехатроника, автоматизация, управление. 2011. № 9. С. 14—19.
29. **Степанов М. Ф., Степанов А. М.** Лингвистические аспекты реализации процедур синтеза и анализа систем управления в системе ГАММА-3 // Системы проектирования, технологической подготовки производства и управления этапами жизненного цикла промышленного продукта (CAD/CAM/PDM — 2016). Труды 16-й Международной конференции / Под ред. А. В. Толока. М.: ООО "Аналитик", 2016. С. 321—325.

Adaptive User Interface for Computer-Aided Control System Design

M. F. Stepanov, mfstepanov@mail.ru, Institute of Electrical and Mechanical Engineering of Yury Gagarin State Technical University of Saratov, Saratov, 410054, Russian Federation,
A. M. Stepanov, amstepanov@mail.ru, Institute of Precision Mechanics and Control of Russian Academy of Sciences, Saratov, 410028, Russian Federation

Corresponding author:

Stepanov Mikhail F., Professor, Institute of Electrical and Mechanical Engineering of Yury Gagarin State Technical University of Saratov, Saratov, 410054, Russian Federation,
E-mail: mfstepanov@mail.ru

Received on November 21, 2017
Accepted on January 09, 2018

The problem of the variety of various types and forms of mathematical models of control systems components is considered. This problem creates additional difficulties in the development of open systems for automation of control tasks. The aim of the work is to develop a method and means for implementing the openness of automation systems for solving problems, both in the types of mathematical models used and in the composition of the set of design operations used in solving problems. We will achieve this goal through the implementation of an adaptive interface that automatically adjusts to tasks, types of mathematical models using user-preferred designations.

For achievement of a goal, we need to solve the following tasks: 1) developing a declarative representation of mathematical models of control theory concepts; 2) developing universal dynamic data structures of the internal representation of mathematical models; 3) developing a procedural means for interpreting the internal representation of mathematical models, including the automatic creation of the user interface.

An approach to solving the problem is proposed based on the principle of separating the declarative description of mathematical models of the components of control systems and their procedural interpretation. The basis of the approach is a model of a formalized representation of knowledge of the subject domain of automatic control in the form of a triad of "predmets — actions — relations". The predmets represent models of concepts of control theory, the actions are operations on them, and the relations determine the connections between them, set requirements for the results of solving problems. The declarative component of the model uses the frame representation of knowledge. To describe the components of the knowledge representation model used, the formal INSTRUMENT-P language was developed. The procedural component complements the description of actions and relations by software implementation in the form of functions of the traditional programming language.

In the GAMMA-3 system, to provide flexibility, automatic adjustment (adaptability) of the user interface, universal dynamic structures of the internal data representation are developed in the form of a system of interconnected classes of the C++ language. Their formation is performed automatically by the built-in translator according to the descriptions given by the user in the INSTRUMENT-P language. Thanks to this, the user helps the system to adapt the human-machine interface to its own preferences. Essentially, based on a user-defined declarative description, the system determines the user's preferences by the classes of tasks to be solved, the types of mathematical models of the control system components used, taking into account the preferred designations.

Particularly it should be noted that it is possible to specify a mathematical representation of mathematical models for subsequent display on the screen forms of the user interface. For this goal we use a record of mathematical expressions in the language of LaTeX, for which a special translator is designed.

Next, the user interface is created automatically. Thus, the proposed approach makes it possible to realize the adaptability of the user interface, as well as the openness of the GAMMA-3 system to extensions. This is done through the automated connection to the system of dynamic load libraries, including software implementations of new design operations (actions) of the control theory domain model to enable the solution of new tasks.

Further work is aimed at simplifying the creation of new design operations. The user does not need to access the software development tools. In the further based on algorithm set by the user, creation of the software of design operations will be carried out by means of system GAMMA-3.

The idea of creation of the computer-aided control system design software for engineers-designers belongs to Professor Albert Georgievich Aleksandrov, the Great Teacher and the Person. Grateful disciples conduct researches in the given direction now. To memory of the professor Aleksandrov's it is devoted.

Keywords: *computer-aided control system design, presentation of data and knowledge, linguistic data providing, automatic creation of user interface*

Acknowledgements: *Work is executed at support of the Russian Foundation for Basic Research (RFBR) (projects 13-07-00647, 15-07-99684).*

For citation:

Stepanov M. F., Stepanov A. M. Adaptive User Interface For Computer-Aided Control System Design, *Programmnaya Ingeneria*, 2018, vol. 9, no. 3, pp. 109—122.

DOI: 10.17587/prin.9.109-122

References

1. **Dening W., Essig H., Maas S.** *The adaptation of virtual man-computer interfaces to user requirements in dialog*, Springer-Verlag, 1981.

2. **Popov E. V.** *Ekspertnye sistemy: Reshenie neformalizovannyh zadach v dialoge s EVM* (Knowledge system: Not formalized tasks solution in dialogue with the computer), Moscow, Nauka, 1987, 288 p. (in Russian).

3. **Batovrin V. K., Vasyutovitch V. V., Gulyaev Yu. V.** et al. *Tehnologija otkrytyh sistem* (Open System technology) / Edited by A. Ya. Oleynikov, Moscow, Janus-K, 2004. 288 p. (in Russian).

4. **TECHNICAL REPORT** ISO/IEC TR 14252 ANSI/IEEE Std 1003.0 First edition 1996-12-15 Information technology — Guide to POSIX ® Open System Environment (OSE).

5. **Gulyaev Yu. V., Oleynikov A. Ya.** *O*tkrytye sistemy: ot prinzipov k tehnologii (Open System: From principles to technology*), Information technology and computer systems,* 2003, no. 3, pp. 6—11 (in Russian).

6. **Dorri M. Kh., Kochemasov A. V., Roshchin A. A.** *Avtomatizacija proektirovanija system i sredstv upravlenija / Paket "Ekspress-Radius 2.1"* (Automation of System Design and Control Facilities, in Toolbox "Ekspress-Radius 2.1" / Toolbox "Ekspress-Radius 2.1"), Moscow, Mosk. Inst. Radiotekhn. Elektron. Avtom., 2000, 106 p. (in Russian).

7. **Kozlov O. S., Kondakov D. E., Skvortsov L. M., Timofeev K. A., Hodakovskij V. V.** Programmnyi kompleks dlja issledovanija dinamiki i proektirovanija tehnicheskih sistem (Program Complex for Studying Dynamics and Design of the Technical Systems), *Informacionnye Tehnologii*, 2005, no. 9, pp. 18—25 (in Russian).

8. **Alexandrov A. G., Panin S. Yu., Stepanov M. F.** CADSD GAMMA-1PC as processor for controllers, *IFAC YAC'95: International Federation of Automatic Control Youth Automatic Conference. Beijing P. R. China: Chinese Association of Automation (CAA) Youth Committee of CAA Beijing Institut of Technology (BIT),* 1995, vol. II, pp. 845—849.

9. **Alexandrov A. G., Panin S. Yu.** Sistema GAMMA-1PC dlja sinteza regulyatorov mnogomernyh sistem (System GAMMA-1PC for synthesis of multivariable control system), *Avtomatizacija v promyshlennosti*, 2003, no. 3, pp. 18—22 (in Russian).

10. **Aleksandrov A. G., Mikhailova L. S., Isakov R. V.** Struktura programmnogo obespechenija dlja avtomatizacii razrabotki algoritmov avtomaticheskogo upravlenija (Structure of software for computer-aided control system design), *Avtomatika i Telemehanika*, 2005, no. 4, pp. 176—184 (in Russian).

11. **Stepanov M. F.** Analiz i sintez system avtomaticheskogo upravlenija v programmnoi srede "INSTRUMENT-3m-I" (Analysis and synthesis of control system by software "INSTRUMENT-3m-I", *Izvestija vysshih uchebnyh zavedenii. Priborostroenie*, 2004, vol. 47. no. 6, pp. 27—30 (in Russian).

12. **Stepanov M. F.** *Avtomaticheskoe reshenie formalizovannykh zadach teorii avtomaticheskogo upravleniya* (Automatic Solution of Formalized Problems of the Automatic Control Systems), Saratov, Saratovskii Gosudarstvennyi Tehnicheskii Universitet, 2000, 376 p. (in Russian).

13. **Stepanov M. F.** Planiruyszie iskusstvennye neyronnye seti v camoorganizuyszihsja intellektualnyh sistemah upravlenija (Artificial Neural Planning Networks in Self-organizing Intelligent Systems), *Doklady Rossiiskoi Akademii Estestvennyh Nauk, Povolzskoe Mezregionalnoe Otdelenie*, 1999, no. 1, pp. 73—99. (in Russian).

14. **Stepanov M. F.** Neyronnye seti dlja planirovanija reshenija zadach teorii avtomaticheskogo upravlenija (Neural Networks for Solutions Planning of the Problems of Automatic Control Theory), *Problemy Upravlenija*, 2004, no. 2, pp. 66—71 (in Russian).

15. **MATLAB** User's Guide. Math Works. 2001.

16. **Spravochnik** *po teorii avtomaticheskogo upravlenija* (Reference book of the theory of automatic control) / Eds. by A. A. Krasovskii, Moscow, Nauka, 1987, 712 p. (in Russian).

17. **Frederick D. K., Kraft R. P., Sadeghi T.** Computer-aided control system analysis and design using interactive graphics, *IEEE Control Systems Magazine,* 1982, December, pp. 19—25.

18. **Avtomatizirovannoe** *proektirovanie system upravlenija* (Computer-aided control system analysis and design)/ Eds. by M. Jamshidi, Ch. J. Cheget, Moscow, Mashnostroenie, 1989, 344 p. (in Russian).

19. **Alexandrov A. G.** O prinzipah postroenija sistemy analiza dinamiki i sinteza ustroystv upravlenija (SAPR SAU) (About principles of construction of system of the analysis of dynamics and synthesis of control systems), *Analiticheskie Metody Sinteza Regulatorov*, Saratovskii Politehnicheskii Institut, 1982, pp. 123—136 (in Russian).

20. **Aleksandrov A. G., Mikhailova L. S., Stepanov M. F.** GAMMA-3 system and its application, *Automation and Remote Control,* 2011, vol. 72, no. 10, pp. 2023—2030.

21. **Besekersky V. A., Popov E. P.** *Teorija system avtomaticheskogo regulirovanija* (The Theory of Systems of Automatic Control), Moscow, Nauka, 1975, 767 p. (in Russian).

22. **Stepanov M. F.** Ob'ektno-orientirovannaja modifikacija jazyka interpretacii strukturnyh shem procedur metodov sinteza system avtomaticheskogo upravlenija (INSTRUMENT+) (Object-oriented updating of language of interpretation of block diagrams of procedures of methods of synthesis of systems of automatic control (INSTRUMENT+)), *Metody i sredstva upravlenija tehnologicheskimi processami (MSUTP 99) (Methods and Control Facilities of Technological Processes* (MCFTP 99), *Proceedings of the Third International Scientific Conference*, Saransk, Publishing house of the Mordovian university, 1999, pp. 300—303 (in Russian).

23. **Stepanov M. F., Bragin T. M., Stepanov A. M.** Kompleksnyi podhod k avtomatizacii proektirovanija i apparatnoi realizacii intellektual'nyh system upravlenija sredstvami mnogofunkcionalnoi sistemy GAMMA-3 (Complex approach to computer-aided design and hardware realization of intellectual control systems by means of the multipurpose GAMMA-3 system), *Sixth All-Russian Multiconference of Control Problems. Proceedings on 4 vol. Rostov-On-Don*, Publishing house of Southern federal university, 2013, vol. 1, pp. 116—120 (in Russian).

24. **Stepanov M. F., Stepanov A. M.** Principy postroenija, arhitektura sredstv proektirovanija, modelirovanija i issledovanija intellektualnyh system upravlenija (Principles of creation, architecture of means of designing, modelling and research of intellectual control systems*), Eighth All-Russia multiconference on control problems. Proceedings on 3 vol.,* Rostov-On-Don. Publishing house of Southern federal university, 2015, vol. 1, pp. 109—112. (in Russian).

25. **Minsky M.** *A framework for representation*, AI Memo N. 306, MIT Cambridge, 1974, 82 p.

26. **Stepanov M. F., Mikhailova L. S.** Integrirovannaja sreda proektirovanija system upravlenija GAMMA: realizacija vhodnogo jazyka (Computer-aided control systems designing by GAMMA system: realization of the source language), *Intellectual Systems: Proceedings of the Ninth international symposium* / Eds. by K. A. Pupkov, Moscow, RUSAKI, 2010, pp. 191—193. (in Russian).

27. **Aleksandrov A. G., Mikhailova L. S., Bragin T. M., Stepanov A. M., Stepanov M. F.** Aspekty primenenija oblachnyh tehnologii dlja avtomatizacii reshenija zadach proektirovanija system upravlenija v sisteme "GAMMA-3" (Aspects of cloudy technologies application for automation of the tasks decision of control systems designing by GAMMA-3 system), *Systems of Designing, Technological Preparation of Manufacture and Management of Stages of Life Cycle of an Industrial Product (CAD/CAM/PDM —* 2012), *Proceedings of the 12-th international conference* / Eds by E. I. Artamonov, Moscow, LLC Analytic. 2012, pp. 111—115 (in Russian).

28. **Alexandrov A. G., Michailova L. S., Stepanov M. F., Bragin T. M., Stepanov A. M.** O razvitii koncepcii avtomaticheskogo reshenija zadach teorii upravlenija v sisteme GAMMA-3 (About Development of the Concepts of the Automatic Decision of the Tasks of the Theory of Control in System), *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2011, no. 9, pp. 14 — 19 (in Russian).

29. **Stepanov M. F., Stepanov A. M.** Lingvisticheskie aspekty realizacii procedur sinteza i analiza system upravlenija v sisteme GAMMA-3 (Linguistic aspects of realization of the synthesis and analysis procedures of control systems by GAMMA-3 system), *Proceedings of the 16-th international conference Systems of Designing, Technological Preparation of Manufacture and Management of Stages of Life Cycle of an Industrial Product (CAD/CAM/PDM — 2016)* / Eds by A. V. Tolok, Moscow, LLC Analytic, 2016, pp. 321—325. (in Russian).

УДК 004.75 DOI: 10.17587/prin.9.123-131

В. П. Потапов, д-р техн. наук, проф., зам. дир., e-mail: potapov@ict.sbras.ru,
С. Е. Попов, ст. науч. сотр., e-mail: popov@ict.sbras.ru, **А. Ю. Ощепков,** аспирант,
e-mail: aosivt@gmail.com, Федеральное государственное бюджетное учреждение науки
Институт вычислительных технологий Сибирского отделения Российской академии наук,
г. Новосибирск

Хранение и обработка данных спутниковых мульти- и гиперспектральных снимков на основе формата Apache Parquet

Рассмотрены способы хранения и алгоритмы последующей обработки данных мульти- и гиперспектральных спутниковых снимков, которые реализуются механизмами распределенных вычислительных систем, входящих в экосистему Apache Hadoop. Отличительной особенностью представленных в работе решений является способ хранения данных дистанционного зондирования. Такой способ позволяет снизить объем хранимой информации за счет архивации посредством технологии Apache Parquet, а также дает возможность работы с данными с использованием запросов Spark SQL. Приведены решения конкретных задач на примере вычисления нормализованных вегетационных индексов спутниковых снимков космических аппаратов Ресурс-П и Sentinel-2A на базе фрэйворков Apache Spark и Apache Flink.

Ключевые слова: *Apache Parquet, Apache Avro, Apache Spark, Apache Flink, Java, GDAL, распределенные информационные системы, сжатие данных, мульти- и гиперспектральные спутниковые снимки*

Введение

Распределенные технологии для обработки информации используются в различных областях научных исследований, в том числе для дистанционного зондирования земли (ДЗЗ) за счет спутниковых данных, получаемых с космических аппаратов (КА) различных форматов и диапазонов съемки.

Вследствие нарастания потоков и накопления большого массива информации, связанной с обработкой данных ДЗЗ, возникает необходимость определения оптимального способа их хранения. Поскольку спутниковые снимки достаточно велики, один канал для извлечения данной информации со спутникового снимка может занимать более 1 Гбайта. При этом размер результирующего файла может увеличиться в 2—3 раза в зависимости от глубины данных в итоговом снимке. Причина в том, что в спутниковом изображении данные могут быть представлены различными типами: byte, short, integer, long, float, double и numeric.

Данные современных КА имеют сложную структуру и занимают большой объем дискового пространства, даже после преобразования из RAW-формата к нативному виду. Так, например, радарные спутниковые снимки КА Sentinel-1A [1] с трехполосной проходкой сенсора могут достигать до 7,5 Гбайт в сжатом виде (zip-сжатие), мульти- и гиперспектральные изображения КА Sentinel-2A [1] и Ресурс-П [2] содержат массив файлов общим размером до 7 и 15 Гбайт соответственно.

Принимая во внимание отмеченные выше соображения, для определения оптимального способа хранения больших объемов спутниковых данных необходимо проводить сравнительный анализ их форматов и выявлять положительные и отрицательные стороны таких форматов с позиции времени упаковки и доступа к сжатым данным в распределенной среде. Для эффективной обработки такого рода массивов необходима информационная система, функционально способная считывать, извлекать, преобразовывать и алгоритмически рассчитывать данные предметной области, эффективно распределяя потоки по вычислительным ресурсам.

В современном технологическом стеке существует достаточно большое количество программных платформ (фрэймворков) для комплексной обработки потоковой информации. К их числу относятся: Apache Storm — многофункциональный фрэймворк для использования потоковой обработки в реальном времени [3]; Heron [4]; Amazon Kinesis — виртуальный сервис для потоковой обработки медиа контента [5];

Disco Map Reduce — фрэймворк для интеллектуального анализа потоковых данных, вероятностного моделирования и полнотекстового индексирования [6]; Apache Kylin — фрэймворк построения распределенных OLAP-представлений в системе Hadoop с поддержкой ANSI SQL [7]; фрэймворки Apache Apex, Flex, Beam, Tez, Spark, Flink, построенные на основе экосистемы Apache Hadoop, объединяющие потоковую и пакетную обработки [7—12]. В подавляющем большинстве случаев представленные платформы и используемые в их составе технологии рассматриваются в контексте работы с большими массивами социальных данных, интернет-ресурсов, для анализа потоков текстовых сообщений и т. п. Структура таких платформ, т. е. программные реализации классов, интерфейсов и методов, имплементируются в средства хранения и обработки в соответствии с форматом именно такого рода информации.

Существует также ряд исследований, в которых описывается сравнение распределенных технологий, их использование и преимущества применения для обработки стека (до 420 Гбайт) данных дистанционного зондирования во взаимодействии с распределенной файловой системой HDFS [13]. Например, мультиитерационный алгоритм сингулярного разложения матриц над данным дистанционного зондирования на основе массово-распределенных технологий [14], кластеризация изображения с использованием метода k-средних [15], создание систем принятия решения на основе построение SOLAP-кубов для мониторинга окружающей среды [16]. Следует отметить, что в подобных решениях увеличение производительности достигается за счет наращивания аппаратной составляющей кластера (больше вычислительных ядер — больше снимков), без акцентирования внимания на оптимизацию расчетного алгоритма в контексте подхода к хранению данных в распределенных файловых системах.

Таким образом, по мнению авторов, задача поиска оптимального способа хранения и обработки спутниковых мульти- и гиперспектральных данных (далее — спектральные данные) за счет применения распределенных процедур для отдельных компонентов снимков (спектральных полос) является актуальной, а ее решение востребовано в современных программных системах процессинга ДЗЗ.

Постановка задачи

Для повышения производительности программных платформ постобработки спектральных данных и уменьшения времени их исполнения в распределенной среде были сформулированы следующие задачи.

1. Сравнение и выбор формата хранения данных, применимых во фрэймворках, с массово-параллельным исполнением заданий на примере экосистемы Apache Hadoop.

2. Сравнительный анализ программных платформ экосистемы Apache Hadoop и выбор для программной реализации алгоритмов постобработки применительно к выбранному формату хранения.

3. Программная реализация и тестирование производительности программных платформ на данных, полученных с современных КА, на примере Sentinel-2A и Ресурс-П с применением выбранных формата хранения данных и среды исполнения заданий.

Постановка задачи рассматривалась в контексте только программного API выбранных фрэймворков, без привязки и сравнения в плане различных вариантов аппаратного обеспечения тестового кластера.

Сравнение форматов хранения

В программной реализации фрэймворков экосистемы Apache Hadoop используется модель устойчивых распределенных наборов данных (*Resilent Distributing Dataset*, RDD-набор) с входными/выходными потоками информации. Модель RDD поддерживает такие форматы представления информации, как JSON/Text, XML, SequenceFile, Apache Avro, Apache Parquet.

Формат JSON (JavaScript Object Notation) — текстовый формат со вложенной структурой пар ключ — значение, предназначенный для обмена данными. В RDD-наборе поддерживается хранение, поиск и передача объектов, включая строковые переменные, числа, массивы, логические значения [17]. Аналогичным образом реализована поддержка формата XML (*Extensible Markup Language*) [18].

Формат SequenceFile — представляет собой наборы пар ключ — значение в двоичном формате, записанные в бинарной последовательности. Такой формат имеет перечисленные далее преимущества над текстовыми форматами.

- Наборы побайтно разделимы (*shared*), поддерживают модель map/reduce [19].
- Наборы поддерживают сжатие над ключом и значением. Форма SequenceFile предоставляет три различных режима сжатия данных (табл. 1). Сжатие ускоряет чтение и запись, а также уменьшает использование дискового пространства.
- Компактные наборы являются альтернативой большому числу мелких файлов. Имя файла может использоваться как ключ, а контент файла — как значение.

Таблица 1

Режимы сжатия файлов последовательности

Режим	Описание
BLOCK	Сжимает последовательности записей в блоках. Интерфейс org.apache.hadoop.io.SequenceFile.BlockCompressWriter
NONE	Не сжимает запись. Интерфейс org.apache.hadoop.io.SequenceFile.Writer
RECORD	Сжатие по одной записи. Интерфейс org.apache.hadoop.io.SequenceFile.RecordCompressWriter

• Поддерживается прямая сериализация и десериализация нескольких произвольных типов данных.

Отрицательная сторона формата SequenceFile — отсутствие встроенной поддержки поиска по комплексному ключу внутри файла последовательности, что необходимо при конкретизации значения внутри массива данных.

Формат Apache Avro — компактный формат сериализации/десериализации двоичных данных, который описывается посредством структурных JSON-схем. Данные хранятся в файле контейнера (файл. avro), в схеме (файл. avsc) хранится метаинформация, описывающая его структуру, поддерживается построчное ведение записи [20].

Формат Apache Parquet — компактный формат сериализации/десериализации двоичных данных с колончатым размещением, обеспечивающий эффективный доступ к данным, структурированность и сжатие [21].

Структурированные форматы файлов, такие как Avro, SequenceFile и Parquet, обеспечивают более высокую производительность за счет уменьшения размера данных на диске и, как следствие, сокращение операций ввода/вывода.

Таблица 2

Сравнение форматов хранения для спектральных данных

№ п/п	Формат	Время создания, с	Объем результирующего файла, Гбайт
1	XML	90	1,2
2	JSON	60	1,9
3	Sequence без архивирования	10	0,737
4	Sequence с архивированием по записи	34	0,236
5	Sequence с блочным архивированием	36	0,234
6	Apache Avro без архивирования	15	1,5
7	Apache Avro с Deflate-архивированием	94	0,273
8	Apache Avro со Snapy-архивированием	21	0,46
9	Apache Avro с GZip-архивированием	144	0,169
10	Apache Parquet без архивирования	26	0,262
11	Apache Parquet с GZip-архивированием	37	0,243
12	Apache Parquet со Snapy-архивированием	56	0,262

Для сравнения и выбора предпочтительного из представленных форматов данных проведены тесты для спектральных данных КА Sentinel-2A и Ресурс-П (табл. 2).

Тестовые испытания проводили на ЭВМ со следующими параметрами: 2xXeon E5-2650 8 @ 2.00GHz, 64 Гбайта RAM, система хранения HDFS, ОС Xubuntu x64 14.04. Использовался один спектральный канал объемом 750 Мбайт.

Формат Apache Parquet применяет сжатие на базе алгоритма измельчения и сборки (*record shredding and assembly algorithm*) [22]. Данный алгоритм эффективен для текстовых файлов и файлов с последовательным размещением байтов значения объекта, например, светимости точки в спектральной полосе спутникового снимка. Исходя из результатов тестирования, выбор был сделан в пользу Apache Parquet по следующим причинам: более интенсивное сжатие данных без использования дополнительных архиваторов, в отличие от строкового размещения; индексированные объекты внутри Parquet-файла позволяют извлекать определенный его элемент; колончатое хранение данных позволяет обращаться не ко всему набору данных, а только к указанным столбцам. В табл. 2 показано, что внутренний формат Parquet эффективен для хранения файлов спектральных полос снимка, также дополнительное применение архиватора практически не дает преимущества (размер 0,262 Гбайт (без архивирования) против 0,243 Гбайт (Gzip)).

Сравнительный анализ программных платформ экосистемы Apache Hadoop

Анализ различных технологий распределенных вычислений [7—12, 20—21, 23—24] показал, что на настоящее время де-факто стандартом для прикладной разработки, в том числе и в области геоинформатики являются программные платформы (фрэймворки) экосистемы Apache Hadoop. Такие платформы относятся к классу SN-систем, которые предполагают режим разделения ресурсов, когда у каждого вычислительного узла имеется своя оперативная память, дисковые массивы и процессорные единицы. Для сравнения были взяты основные программные платформы и рассмотрены их функциональные свойства (возможности) применительно к задаче обработки спектральных данных (табл. 3).

В плане выбранного подхода, где каждое задание является изолированным контейнером вычислений над распределенным набором данных, определяемым значениями точек одного или нескольких спектральных снимков, преимущество отдают компонентам с поддержкой пакетного режима обработки данных и с поддержкой механизмов сжатия (*batch-proccessing framework*). Исходя из проведенного анализа можно сделать вывод, что такому условию удовлетворяют фрэймворки Spark и Flink. С позиции поставленной

Таблица 3

Сравнение программных платформ Apache Distributed Programming (Apache Hadoop)

Свойство компонента	Фрэймворк Apache Ecosystem						
	Spark	**Ignite**	**Tez**	**Flink**	**Apex**	**Storm**	**Beam**
Возможность кеширования данных и хранения промежуточных результатов вычислений в оперативной памяти	+	+	–	+	+	+	+
Поддержка потоков	+	+	–	+	+	+	+
Пакетная система обработки данных	+	–	+	+	–	–	–
Поддержка распределенной файловой системы	+	+	+	+	+	+	–
Изолированные JVM-контейнеры	+	–	+	+	–	–	–
Поддержка сжатия данных	+	–	–	+	–	–	–

задачи — они поддерживают два основных API (инструмента): API DataSet для обработки конечных наборов данных (пакетная обработка); API DataStream для обработки потенциально неограниченных потоков данных (потоковая обработка), позволяя обеспечить потоковую передачу данных в реальном времени с высокой пропускной способностью. Эти интерфейсы предоставляют широкий набор программных механизмов, таких как Spark/Flink SQL для работы со структурированными данными, поддерживают создания SQL-подобных запросов (NoSQL) к различным источникам данных [23, 24].

Программная реализация

На первом этапе предлагаемого в статье подхода к оптимизации вычислений в средах с массово-параллельным исполнением расчетных заданий необходимо описать процесс создания файлов в выбранном способе хранения (Apache Parquet). Рассмотрим диаграмму последовательности программной реализации этого процесса (рис. 1).

На **шаге 1** происходит конфигурирование программных объектов взаимодействия с HDFS, где осуществляется загрузка файла изображения в объект типа **FileStatus**. На **шаге 2** запускается процесс итерирования объектов **FileStatus** посредством создания потока Stream, выделение на каждом шаге программного объекта Dataset (gdalDataset) и формирование коллекции List<Dataset>. На **шаге 3** создается объект типа Parquet Writer для сохранения Parquet-файла. На **шаге 4** происходит создание и инициализация объекта **SatelliteImage** parquetObject. На **шаге 5** осуществляется итерация коллекции gdalDataset,

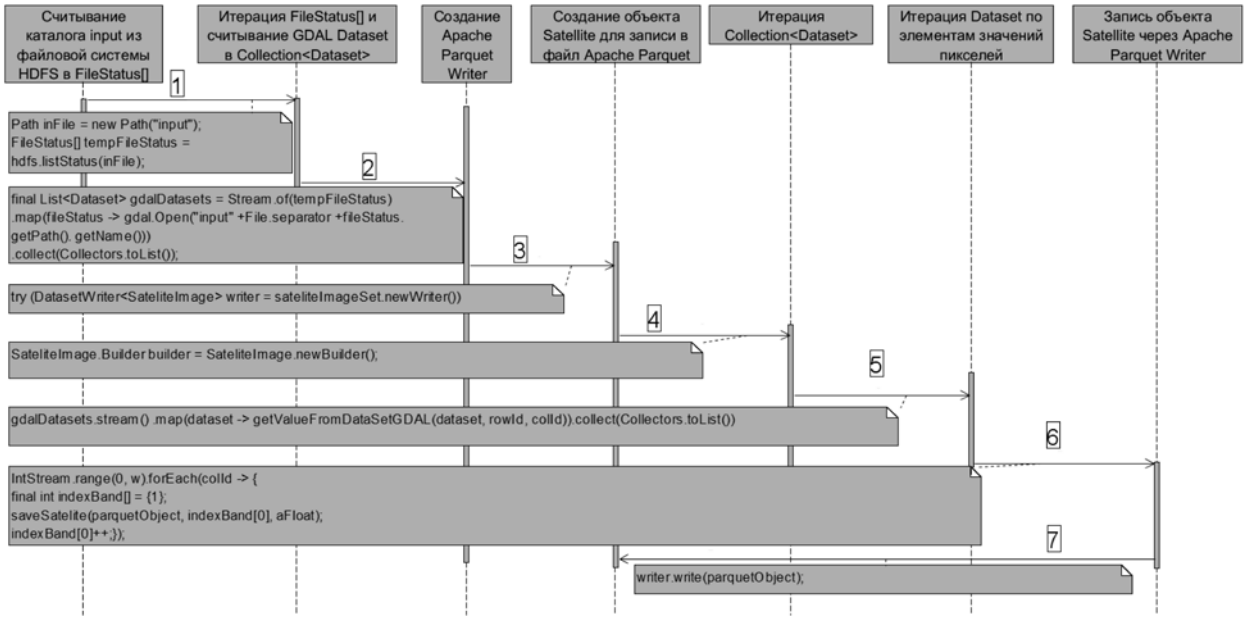


Рис. 1. Процесс создания Parquet-файла

выделение конкретного Dataset и передача его в метод getValueFromDataSetGDAL. На **шаге 6** происходит итерирование значений точек изображений, хранящихся в коллекции Dataset, и передача значений в метод saveSatellite, который принимает на вход инициализированный объект Apache Parquet, номер канала и передаваемое значение точки спектральных каналов изображения. За счет рефлексии определяется вызываемый метод в объекте Parquet-файла. На **шаге 7** происходит запись полученного объекта в распределенную файловую систему HDFS. По завершении всех проходов итераций на выходе получаем Parquet-файл, который имеет структуру, представленную на рис. 2.

```
{
  "type": "record",
  "name": "SateliteImage",
  "namespace": "aosivt.model",
  "fields": [
    {"name": "rowId", "type": "int"   },
    {"name": "colId", "type": "int"   },
    {"name": "b1"   , "type": "float", "default": "0" },
    {"name": "b2"   , "type": "float", "default": "0" },
    {"name": "b3"   , "type": "float", "default": "0" },
    {"name": "b4"   , "type": "float", "default": "0" },
    {"name": "b5"   , "type": "float", "default": "0" },
    {"name": "b6"   , "type": "float", "default": "0" },
    {"name": "b7"   , "type": "float", "default": "0" },
    {"name": "b8"   , "type": "float", "default": "0" },
    {"name": "b9"   , "type": "float", "default": "0" },
    {"name": "b10"  , "type": "float", "default": "0" },
    {"name": "b11"  , "type": "float", "default": "0" },
    {"name": "b12"  , "type": "float", "default": "0" }
  ]
}
```

Рис. 2. Метаописание схемы хранения и размещения данных в Parquet-файле: b1-b12 — наименование спектральных полос на примере снимка Ресурс-П

Далее опишем процесс работы программной реализации алгоритма обработки Parquet-файлов на базе программных платформ Apache Spark и Apache Flink (рис. 3 и 4 соответственно)

На **шаге 1** (номер 1 на диаграмме, см. рис. 1) происходит инициализации запуска расчетов, определяется конфигурация кластера, создается сессия Apache Spark.

Параметры узла определены локальными ресурсами машины. **На шаге 2** создается объект Spark **Dataset** с предопределенным форматом записи **Row**. Посредством ранее созданной сессии происходит обращение к Parquet-файлу, расположенному в распределенной файловой системе HDFS. Фрэймворк Apache Spark обладает встроенным механизмом считывания данных из Apache Parquet, приведенных к типу Dataset, минуя промежуточные этапы использования схемы определения. При обращении к хранилищу данных, реализованному посредством Apache Parquet, осуществляется выборка SQL-запросом, который в процессе считывания информации в набор данных Apache Spark проводит расчет (например, NDVI-индекса растительности [25]). На **шаге 3**

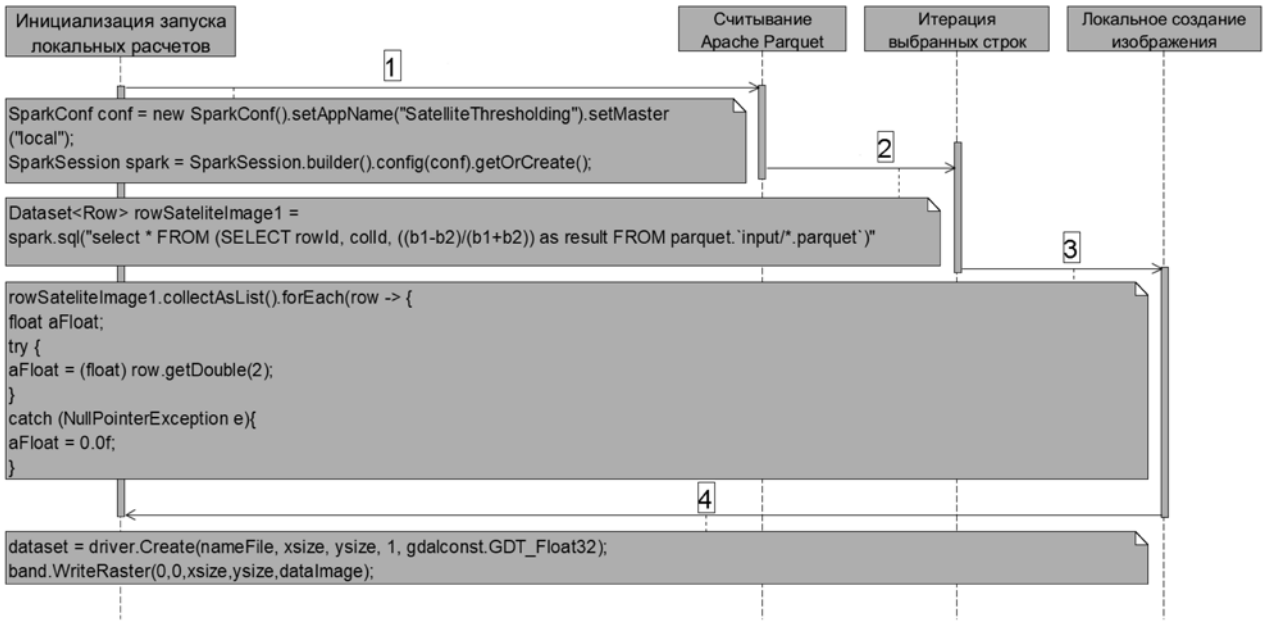


Рис. 3. Диаграмма последовательности. Пост-процессинг Parquet-файла на базе Apache Spark API

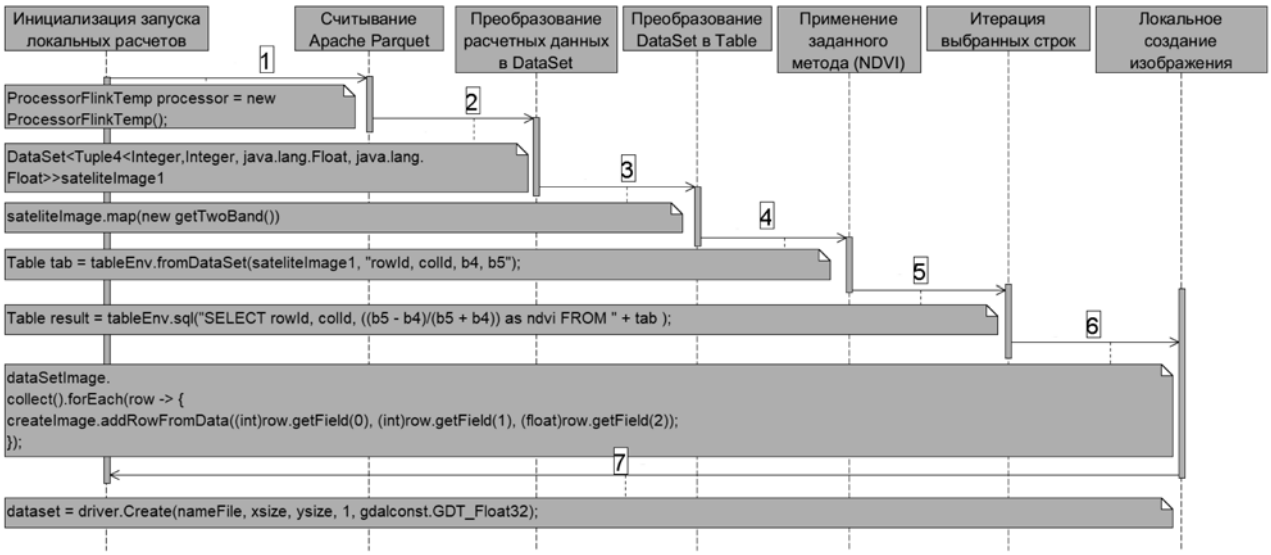


Рис. 4. Диаграмма последовательности. Пост-процессинг Parquet-файла на базе Apache Flink API

происходит итерация Row-элементов Dataset и передача полученных значений точек в метод объекта **CreateImag**e addRowFromData. В созданном объекте **CreateImage** определены значения высоты и ширины результирующего изображения, а также его имя и путь для сохранения в файловую систему. Метод addRowFromData за одну итерацию принимает три значения: номер строки, номер столбца и значение индекса (тип Float). Полученный объект можно использовать в сторонних сервисах брокеров сообщений, построенных на основе программных платформ Kafka, RabbitMQ, для передачи обработанных данных, способных принять и сформировать аналитические отчеты. На **шаге 6** происходит сохранение данных и формирование изображения стандартного формата (PNG, JPEG и т. п.) посредством библиотеки Java GDAL [26].

Аналогичный процесс рассмотрим для фрэйморка Apache Flink (см. рис. 4).

На **шаге 1** происходит инициализация запуска расчетов. Создается объект **ExecutionEnvironment**, отвечающий за исполняемое окружение, объект **FlinkConfiguration**, настраивающий среду глобальных переменных Flink. Данные объекты предопределены как статические переменные объекта **ProcessorFlinkTemp**. Далее инициализируется объект пакетного исполняемого окружения **BatchTableEnvironment** на основе ранее созданного **ExecutionEnvironment**. **BatchTableEnvironment** необходим для обращения к локальным конечным наборам данных. На **шаге 2** осуществляется заполнение коллекции **Dataset** переопределенным типом Tuple4<Integer,Integer,Float,Float> посредством метода getSateliteImageDataSource, который возвращает объект DataSource<Tuple2<Void,SateliteImage>>, содержащий непосредственно данные изображения. На **шаге 3** методом getTwoBand получаем значения необходимых спектральных полос. К сожалению, нативная поддержка Apache Parquet отсутствует в платформе Apache Flink, однако при этом присутствует поддержка абстрактных объектов FileInputFormat. Для работы с форматом Parquet-файлов использовался дополнительный API Apache Avro, предоставляющий классы Schema и AvroParquetInputFormat, которые позволяют реализовать доступ к файлам Parquet-формата на базе схем хранилища. На **шаге 4** происходит преобразование объекта DataSet<Tuple4<Integer,Integer,Float,Float>> с набором данных (номер строки, колонки в изображении, значение точки красного канала и инфракрасного каналов). Преобразование Dataset в Table позволяет обращаться к данным SQL-подобными запросами. На **шаге 5** создается результирующая таблица с расчетными данными (NDVI-индекса растительности [25]).

Шаги 6 и 7, представленные на диаграмме, аналогичны **шагу 4** в алгоритме, который реализуется в Apache Spark, за исключением необходимости отслеживания null-значений.

Тесты производительности

Для сравнения выбранных программных платформ применительно к формату хранения данных Apache Parquet и к скорости работы в режиме пакетной обработки были проведены тесты производительности с характеристиками, представленными в табл. 4.

Тесты проводили на мини-кластере Cloudera CDH v5.9.0 со следующими параметрами: cloudera-master: 2xXeon E5-2650 8@2.00GHz, 64 Гбайт RAM; cloudera-node1: Dell — 2xXeon E5-2620v2 6@ 2.10GHz, 96 Гбайт RAM; cloudera-node2: Intel Core i5-2400K@3.5GHz, 16Гбайт RAM. Total cores: 32, RAM 54 Гбайт. На всех узлах использована распределенная файловая система HDFS, поверх системы ввода/вывода NAS Dell PowerVault MD3800f (SAS 12x4Tb и 12x600Gb). Доступ

Таблица 4

Тесты производительности Apache Spark/Flink в сравнении с ПО Exelis ENVI и Sentinel Toolbox в распределенной файловой системе HDFS (Расчет индекса NDVI [25])

Тип снимка	Объем данных, Гбайт	Время работы, с			
		Apache Spark	Apache Flink	ENVI	Sentinel Toolbox
Apache Parquet (CompressionCodecName.UNCOMPRESSED)					
Ресурс-П	20,1	61	413	723	—
Sentinel-2A	5,75	27	143	—	184
Apache Avro (CodecFactory.bzip2Codec)					
Ресурс-П	20,1	106	705	723	—
Sentinel-2A	5,75	59	307	—	184
Примечание. Общее число снимков, использованных в расчетах, равно 10.					

узлов к NAS обеспечивают 16 Гбит/с сетевые платы Emulex lpe 16000 через коммутатор Brocade 300 с коммутационной матрицей 192 Гбит/с, ОС Ubuntu x64 14.04.

Запуск ПО Exelis ENVI и Sentinel Toolbox осуществляли на аппаратном обеспечении одного узла (cloudera-master), без использования функциональности массово-параллельного исполнения платформы Apache Spark.

При сравнении полученных результатов тестирования относительно проведенных расчетов в программных комплексах Exelis ENVI/Sentinel Toolbox были получены идентичные значения точек изображений (в поточечном сравнении). Из данных табл. 4 следует, что время обработки данных формата Apache Parquet на базе программной платформы Apache Spark в 7—9 раз меньше, чем аналогичное время на платформе Apache Flink при низкой нагрузке на аппаратную часть. Отсутствие нативной поддержки Apache Parquet в программной модели Flink, и как следствие, дополнительные шаги преобразования к типу Dataset для доступа через SQL-запросы, привели к падению производительности. Также несмотря на то, что Apache Avro с GZip-архивированием получает меньший размер файла спектральной полосы снимка, время, затрачиваемое на его подготовку, в 7 раз выше, чем лучший результат (время подготовки/размер) для формата Apache Parquet (см. табл. 2, п.п. 9, 10), что дает негативный вклад в общее время работы.

Заключение

Рассмотрены несколько способов конвертации данных спектральных спутниковых снимков, а также реализации их обработки. Использование в качестве хранилища данных Apache Parquet предоставляет возможности кластерных вычислений, поддерживает множественный доступ и оптимальное сжатие данных. В операциях попиксельной обработки присутствует возможность обращения к данным снимка как к реляционной базе данных с возможностью выделения конкретных зон интересов (областей снимка, заданных начальным и конечным пикселем, соответственно по горизонтали и вертикали), исключая обращение ко всему набору данных.

Из представленных результатов тестирования можно сделать вывод, что использование Apache Spark предпочтительней для обработки конечных наборов данных, которыми располагает Apache Parquet. Фреймворк Apache Spark поддерживает нативные методы ввода/вывода сжатых данных. В отличие от Apache Flink они обеспечивают меньшую нагрузку на аппаратную часть компьютера.

Результаты анализа, представленные в работе, демонстрируют эффективность использования Apache Spark в сочетании с хранилищем Apache Parquet. Такое сочетание может быть эффективно применено к хранению и обработке данных в других областях науки, где необходимо проводить сравнение, анализ, хранение больших объемов информации, сохраняя высокий уровень доступности к данным, предоставляя при этом возможности распределенного доступа и обработки данных.

Список литературы

1. **ESA** Sentinel Online. URL: https://sentinel.esa.int/web/sentinel/missions/sentinel-5p (дата обращения 07.12.2017).
2. **Ресурс-П** — Российские космические системы. URL: http://russianspacesystems.ru/bussines/dzz/orbitalnaya-gruppirovka-ka-dzz/resurs-p/ (дата обращения 07.12.2017).
3. **Apache** Storm Documentation. URL: http://storm.apache.org/releases/1.1.1/index.html (дата обращения 30.11.2017).
4. **Heron** Documentation. URL: https://twitter.github.io/heron/docs/getting-started/ (дата обращения 30.11.2017).
5. **Документация** Amazon Kinesis. URL: https://aws.amazon.com/ru/kinesis/ (дата обращения 30.11.2017).
6. **Disco** Map Reduce Documentation. URL: http://disco.readthedocs.io/en/develop/ (дата обращения 30.11.2017).
7. **Apache** Kylin Overview. URL: http://kylin.apache.org/docs21/ (дата обращения 30.11.2017).
8. **Apache** Apex Documentation. URL: http://apex.apache.org/docs.html (дата обращения 30.11.2017).

9. **Apache** Spark TM — Lightning-Fast Cluster Computing. URL: http://spark.apache.org/ (дата обращения 30.11.2017).
10. **Apache** Flink: Scalable Batch and Stream Data Processing. URL: https://flink.apache.org/ (дата обращения 30.11.2017).
11. **Carbone P., Ewen S., Haridi S., Katsifodimos A., Markl V., Tzoumas K.** Apache Flink: Stream and Batch Processing in a Single Engine // Bulletin of the IEEE Computer Society Technical Committee on Data Engineering. 2015. Vol. 38. P. 28—38.
12. **Perera S., Perera A., Hakimzadeh K.** Reproducible Experiments for Comparing Apache Flink and Apache Spark on Public Clouds // Computing Research Repository. Интернет-журнал. 14.10.16. URL: https://arxiv.org/abs/1610.04493.
13. **Wei Huang, Lingkui Meng, Dongying Zhang.** In-Memory Parallel Processing of Massive Remotely Sensed Data Using an Apache Spark on Hadoop YARN Model // IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing. 2016. Vol. 10. P. 3—19.
14. **Sun Z., Chen F., Chi M., Zhu Y.** A Spark-Based Big Data Platform for Massive Remote Sensing Data Processing // Data Science. Lecture Notes in Computer Science. 2015. Vol. 9208. P. 120—126.
15. **Tapan Sharma, Vinod Shokeen, Sunil Mathur.** Multiple K Means++ Clustering of Satellite Image Using Hadoop MapReduce and Spark // International journal of advanced studies in computer science and engineering. 2016. Vol. 5. P. 23—31.
16. **Li J., Meng L., Wang F. Z., Zhang W., Cai Y.** A map-reduce-enabled SOLAP cube for large-scale remotely sensed data aggregation // Computers & Geosciences. 2014. Vol. 70. P. 110—119.
17. **Введение** в JSON. URL: http://www.json.org/json-ru.html (дата обращения 30.11.2017).
18. **Основы** XML для начинающих пользователей. URL: https://www. ibm.com/developerworks/ru/library/x-newxml/ (дата обращения 30.11.2017).
19. **Обзор** файла последовательности. URL: https://wiki.apache.org/hadoop/SequenceFile (дата обращения 30.11.2017).
20. **Документация** формата Apache Avro. URL: http://avro.apache.org/docs/current/ (дата обращения 30.11.2017).
21. **Документация** формата Apache Parquet. URL: http://parquet.apache.org/documentation/latest/ (дата обращения 30.11.2017).
22. **Efficient** DataFrame Storage with Apache Parquet URL: https://tech.blue-yonder.com/efficient-dataframe-storage-with-apache-parquet/ (дата обращения 09.01.2018).
23. **Rishi Yadav**. Spark Cookbook. Birmingham, UK: Packet Publishing Ltd., 2015. 98 p.
24. **Friedman E., Tzoumas K.** Introduction to Apache Flink, CA: O'Reilly Media, Inc. 2016. 46 p.
25. **Вегетационные** индексы. URL: http://gis-lab.info/qa/vi.html (дата обращения 11.12.2017).
26. **Overview** GDAL/OGR Java Bindings API. URL: http://gdal.org/java/ (дата обращения 11.12.2017)/

Storage and Data Handling Multi and Hyperspectral Satellite Images based on Apache Parquet

V. P. Potapov, potapov@ict.sbras.ru**, S. E. Popov,** popov@ict.sbras.ru**,**
A. Ju. Oshchepkov, aosivt@gmail.com, Institute of Computational Technologies SB RAS, Novosibirsk, 630090, Russian Federation

Corresponding author:

Popov Semion E., Chief Scientist, Institute of Computational Technologies SB RAS, Novosibirsk, 630090, Russian Federation,
E-mail: popov@ict.sbras.ru

Received on December 15, 2017
Accepted on January 12, 2018

The article describes ways storing and processing the satellite spectral imaging data by means of distributed computing systems included in the Apache Hadoop. The review of different works devoted to the distributed processing of such data shows that improvement of the performance is achieving by the build-up or extending hardware parts of the computing cluster.

The distinctive feature of the proposed approach is the way of storing the spectral images data in the Parquet-file format. It shows that a columnar disposition of the data provides an access to different pieces of the image pixel values like to the record in the database, avoiding the whole image loading into CPU memory. Besides, it retains the way of the parallel image processing by the per-pixel manner.

The authors have made a comparative analysis of the storage formats for the spectral images, such as JSON, XML, sequence-file, Apache Avro, Apache Parquet. Which is consists from the following steps: the data extraction from Parquet-file, the data conversion to the Spark or Flink Dataset, the computing of the normalized vegetation index, and includes the process of the result data iterating and saving them to the HDFS.

The stress tests have been accomplished on the hybrid frameworks of the Apache Hadoop ecosystem. The Apache Spark API has been chosen as the preferable spectral images processor by the reason of the native input/output methods for the Parquet-file and lesser load to the cluster hardware.

In conclusion, authors demonstrate the calculating of the normalized vegetation index (NDVI) on the example of two images of the spacecraft missions (Resource-P and Sentinel-1A) based on the Apache Spark and the Apache Flink frameworks for the auditing and confirmation that the choice of the technology described in the work was correct.

Keywords: *Apache Parquet, Apache Spark, Apache Flink, Java, GDAL, distributed information systems, remote sensing data, spectral satellite images*

For citation:

Potapov V. P., Popov S. E., Oshchepkov A. Ju. Storage and Data Handling Multi and Hyperspectral Satellite Images based on Apache Parquet, *Programmnaya Ingeneria,* 2018, vol. 9., no 3, pp. 123—131.

DOI: 10.17587/prin.9.123-131

References

1. **ESA** Sentinel Online, available at: https://sentinel.esa.int/web/sentinel/missions/sentinel-5p.
2. **Resurs-P** — Rossijskie kosmicheskie sistemy (Russian space systems), available at: http://russianspacesystems.ru/bussines/dzz/orbitalnaya-gruppirovka-ka-dzz/resurs-p/ (in Russian).
3. **Apache** Storm Documentation, available at: http://storm.apache.org/releases/1.1.1/index.html
4. **Heron** Documentation, available at: https://twitter.github.io/heron/docs/getting-started/
5. **Amazon** Kinesis Documentation available at: https://aws.amazon.com/ru/kinesis/
6. **Disco** Map Reduce Documentation, available at: http://disco.readthedocs.io/en/develop/
7. **Apache** Kylin Documentation, available at: http://kylin.apache.org/docs21/
8. **Apache** Apex Documentation, available at: http://apex.apache.org/docs.html
9. **Apache** Spark TM — Lightning-Fast Cluster Computing, available at: http://spark.apache.org/
10. **Apache** Flink: Scalable Batch and Stream Data Processing, available at: https://flink.apache.org/
11. **Carbone P., Ewen S., Haridi S., Katsifodimos A., Markl V., Tzoumas K.** Apache Flink: Stream and Batch Processing in a Single Engine, *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 2015, vol. 38, pp. 28—38.
12. **Perera S., Perera A., Hakimzadeh K.** Reproducible Experiments for Comparing Apache Flink and Apache Spark on Public Clouds, *Computing Research Repository, Internet Journal*, 14.10.16, available at: https://arxiv.org/abs/1610.04493.
13. **Wei Huang, Lingkui Meng, Dongying Zhang** In-Memory Parallel Processing of Massive Remotely Sensed Data Using an Apache Spark on Hadoop YARN Model, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing,* 2016, vol. 10, pp. 3—19.
14. **Sun Z., Chen F., Chi M., Zhu Y.** A Spark-Based Big Data Platform for Massive Remote Sensing Data Processing, *Data Science. Lecture Notes in Computer Science*, 2015, vol. 9208, pp. 120—126.
15. **Tapan Sharma, Vinod Shokeen, Sunil Mathur** Multiple K Means++ Clustering of Satellite Image Using Hadoop MapReduce and Spark, *International journal of advanced studies in computer science and engineering,* 2016, vol. 5, pp. 23—31.
16. **Li J., Meng L., Wang F. Z., Zhang W., Cai Y.** A map-reduce-enabled SOLAP cube for large-scale remotely sensed data aggregation, *Computers & Geosciences*, 2014, vol. 70, pp. 110—119.
17. **Documentation** JSON, available at: http://www.json.org/json-ru.html.
18. **Osnovy** XML dlja nachinajushhih pol'zovatelej (XML basics for beginners), available at: https://www.ibm.com/developerworks/ru /library/x-newxml/ (in Russian).
19. **Overview** Sequence File, available at: https://wiki.apache.org/hadoop/SequenceFile
20. **Documentation** Apache Avro, available at: http://avro.apache.org/docs/current/
21. **Documentation** Apache Parquet, available at: http://parquet.apache.org/documentation/latest/
22. **Efficient** DataFrame Storage with Apache Parquet, available at: https://tech.blue-yonder.com/efficient-dataframe-storage-with-apache-parquet/
23. **Rishi Yadav**. *Spark Cookbook*, UK, Packet Publishing Ltd. 2015, 98 p.
24. **Friedman E., Tzoumas K.** *Introduction to Apache Flink*, CA, O'Reilly Media, Inc. 2016. 46 p.
25. **Vegetation** indices, available at: http://gis-lab.info/qa/vi.html
26. **Overview** GDAL/OGR Java Bindings API, available at: http://gdal.org/java/

УДК 004.422 DOI: 10.17587/prin.9.132-139

С. Д. Махортов, д-р физ.-мат. наук, зав. кафедрой, e-mail: msd_exp@outlook.com,
И. В. Клейменов, магистрант, e-mail: alenor96@gmail.com, Воронежский государственный университет

Мобильное приложение для определения спектра частот и выделения нот в акустическом сигнале

Вычисление спектра частот во входящем аудиосигнале представляет довольно распространенную и хорошо изученную задачу. Однако ее решение на мобильном устройстве накладывает определенные ограничения, такие как сравнительно небольшой объем имеющейся оперативной памяти и небольшая производительность процессора. В настоящей работе рассмотрен один из подходов к решению указанной проблемы, направленный на минимизацию затрат ресурсов, что дает дополнительные возможности обработки входного сигнала в реальном времени без существенных задержек.

Ключевые слова: *спектр частот, YIN, акустический сигнал, распознавание нот, мобильное приложение*

Введение

С течением времени компьютерные технологии все ближе соприкасаются с различными сторонами человеческой жизни. Не стала исключением и музыкальная сфера — на данный момент разработано множество приложений, которые тем или иным образом помогают процессу создания и записи музыки. Спектр их назначения широк, он включает, например, виртуальные гитарные процессоры, а также программы, предоставляющие возможность детального редактирования музыкальных партитур с последующим их воспроизведением. Одним из наиболее популярных видов подобных приложений стали тюнеры, дающие возможность настроить инструмент в любой момент в любом месте с использованием, например, всего лишь мобильного телефона.

Подобные тюнеры работают благодаря возможности разложения входного акустического сигнала на частотные спектры, что в свою очередь позволяет однозначно определить звучащую ноту.

Основываясь на данном принципе, можно пойти еще дальше — распознавать не только отдельные ноты, но и генерировать партитуры в реальном времени, проводя анализ всего музыкального произведения. Одна из трудностей, которая встает на пути создания приложения с подобной функциональностью, это проблема быстродействия. Для достаточно точного разбора музыкального произведения требуется выполнение огромного количества математических расчетов за короткий промежуток времени.

О реализации мобильного приложения для распознавания последовательностей нот было заявлено представителями Томского государственного университета систем управления и радиоэлектроники в 2014 г. [1]. Утверждалось, что разрабатываемое приложение верно определяет не менее 90 % нот, а другие известные решения, применимые для распознавания нот на персональных компьютерах, такие как Melodyne и Praat, верно распознают лишь 62 и 71 % соответственно. Однако на текущий момент (конец 2017 г.) авторам настоящей статьи не удалось обнаружить возможности скачать указанный продукт для тестирования.

Следует отметить, что упомянутые выше два известных приложения также используют алгоритмы распознавания частот, но с несколько другими целями. Приложение Melodyne, автором идеи которого является П. Нойбэкер (Германия), представляет собой продукт для обработки аудиозаписей, ориентированный на инструментальные и вокальные партии. Приложение предоставляет пользователю визуальную частотную модель звука с возможностью изменения той или иной частоты, наложения различных звуковых эффектов, таких как эхо, задержка и т. д. Реализована также функция для превращения частотных моделей в нотную модель с дальнейшей возможностью транспонирования аудиозаписи [2].

Приложение Praat, предназначенное для анализа человеческой речи, разработано в Амстердаме П. Боэрсма и Д. Викнинком [3]. Продукт реализует возможности записи голоса с дальнейшей его обработкой на основе частотной модели. Он, в частности, может использоваться для изучения иностранных языков, а также для реабилитации людей с нарушениями речевого аппарата. Основная идея заключается в сверке звуков, произносимых пользователем, с какими-либо другими (например, с произношением носителей языка в случае применения Praat для обучения).

Задача создания мобильного приложения для распознавания нот была успешно решена шведской компанией DoReMIR Music Research AB, которая разработала решение сначала для мобильных устройств на платформе операционной системы iOS, а затем и для персональных компьютеров [4]. Продукт данной компании также предлагает пользователю возможности генерации музыкальных партитур на основе распознанных нот.

Таким образом, существующие широко известные проекты обработки цифровой музыки не охватывают столь популярную платформу, как Android.

Целью настоящей работы является создание мобильного приложения для операционной системы Android, обладающего возможностью обработки входного акустического сигнала, в частности, определения звучащей в конкретный момент времени музыкальной ноты с последующим внесением ее в партитуру. Эти функциональные возможности должны быть реализованы с условием работы в реальном времени, т. е. все перечисленные действия должны быть применимы к постоянно меняющемуся сигналу. Однако в начальной версии данного приложения требуется способность выделения лишь одной единовременно звучащей ноты. Другими словами, способность к качественному распознаванию двузвучий и аккордов пока не декларируется.

Существующие базовые технологии

Согласно теореме Котельникова (в зарубежной литературе — Найквиста — Шеннона) [5], акустический сигнал может быть восстановлен по своим дискретным отсчетам с частотой дискретизации, в два раза превышающей максимальную частоту спектра. Следовательно, при распознавании акустического сигнала, воспроизводимого музыкальным инструментом, необходимо использовать частоту дискретизации не меньше, чем частота максимально высокой ноты. Как правило, таковой считается нота ми пятой октавы с частотой (5274 ± 80) Гц [6].

Наиболее распространенным инструментом для решения задачи распознавания частотного спектра сигнала является алгоритм быстрого преобразования Фурье [7]. Однако он требует выполнения большого числа математических вычислений, что может существенно влиять на скорость обработки сигнала, особенно при реализации на маломощных мобильных устройствах. Другой способ, рассматриваемый в настоящей работе, заключается в использовании алгоритма YIN. Обзор данного алгоритма приведен в журнале акустического общества Америки в 2002 г. [8].

В настоящей работе описана общая логика реализации приложения с использованием каждого из отмеченных выше алгоритмов и проведен сравнительный анализ их быстродействия. Для повышения точности определения частот используются дополнительные методы, такие как наложение оконных функций на входящий сигнал (при быстром преобразовании Фурье) и использование цифровых фильтров.

В результате процесс определения ноты, представленной исходным аудиосигналом, состоит из перечисленных далее шагов.

Шаг 1. Оцифровка сигнала.

Шаг 2. Пропускание сигнала через цифровой фильтр.

Шаг 3. Получение частотного спектра с помощью одного из соответствующих алгоритмов.

Шаг 4. Если был использован алгоритм быстрого преобразования Фурье, то применение оконной функции.

Шаг 5. Сопоставление полученной частоты с заранее известной таблицей частот нот.

В рассматриваемой задаче вычисление спектра аудиосигнала проводится за короткий отрезок времени. Однако в теории спектральный анализ предназначен для исследования непрерывных периодических сигналов. Поэтому при "обрезании" сигнала в его спектре появляются исходно не существовавшие высокочастотные составляющие. Этот эффект может приводить к ситуации, когда при наличии идущих друг за другом сигналов с малой и большой амплитудами малый сигнал теряется на фоне фиктивной составляющей сигнала с большой амплитудой.

Для борьбы с появлением таких ложных составляющих (артефактов) прибегают к использованию оконных функций, изменяющих оригинальный сигнал в каждом анализируемом отрезке (окне). Использование оконной функции заключается в предварительном умножении на нее отсчетов сигнала, после чего проводится расчет спектра. Некоторые типы оконных функций имеют нулевые коэффициенты на краях, что приводит к потере крайних значений выборки. Для таких функций при расчете используется небольшой сдвиг.

Приведем некоторые примеры оконных функций, а также кадры после их применения.

- Синус-окно (рис. 1):

$$w(n)=\sin\frac{\pi\times n}{N-1}.$$

- Окно Гаусса (рис. 2):

$$w(n)=\exp\left(-\frac{1}{2}\left(\frac{n-A}{\sigma\times A}\right)^2\right),\quad A=\frac{N-1}{2}.$$

- Окно Хемминга (рис. 3):

$$w(n)=0{,}54-0{,}46\cos\left(\frac{2\times\pi\times n}{N-1}\right).$$

Здесь $n = 0, 1, ..., N-1$, N — число отсчетов в рассматриваемом кадре, σ — коэффициент, определяющийся опытным путем.

На рис. 1—3 отображены результирующие кадры после применения соответствующих оконных функций. На рис. 2 наблюдается полное отсутствие артефактов спектра. Однако при этом имеется рас-

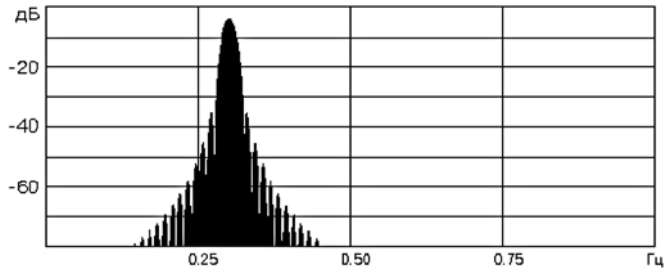


Рис. 1. Синусное окно

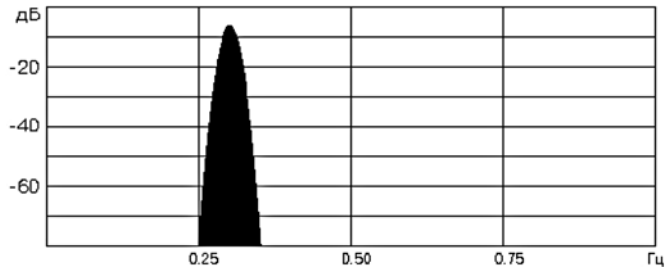


Рис. 2. Окно Гаусса

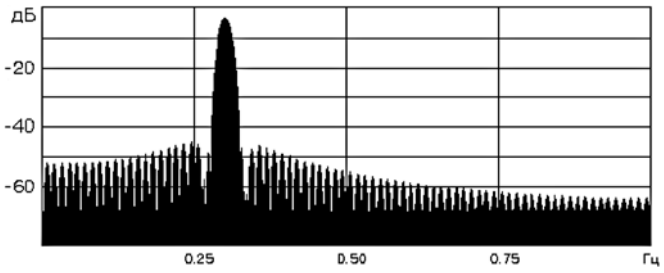


Рис. 3. Окно Хемминга

ширенный основной "лепесток", что в свою очередь приводит к потере точности вычисления частоты. В то же время на рис. 1 и 3 расширение "лепестка" заметно меньше, но артефакты все еще присутствуют.

Таким образом, нельзя сделать однозначный вывод о том, какая из оконных функций объективно лучше. Подбор оптимального решения рекомендуется проводить экспериментальным методом, основываясь на точности расчета частоты с использованием каждой из нескольких оконных функций или их суперпозиций.

Реализация с использованием алгоритма Фурье

Дискретное преобразование Фурье — это один из видов преобразований Фурье, используемый для анализа частот в дискретном сигнале. Дискретное преобразование Фурье принимает на вход дискретную функцию, описывающую входящий сигнал:

$$X_k = \sum_{n=0}^{N-1} x_n \mathbf{e}^{-\frac{2\pi i}{N} kn},$$

где N — число отсчетов; x_n — значение сигнала в момент n, X_k, $k = 0, ..., N-1$ — комплексные амплитуды синусоидальных сигналов, из которых состоит исходный сигнал, k — индекс частоты.

Результирующий массив X_k будет заполнен полезной информацией только наполовину, так как вторая его часть является зеркальным отражением первой и может быть исключена из рассмотрения. Данное обстоятельство представляет негативную сторону алгоритма в плане эффективности, так как половина времени, затраченная на вычислительный процесс, не приносит полезного результата. На основе оставшихся данных можно легко вычислить значения амплитудного и фазового спектров сигнала:

$$A_k = \sqrt{X_k.\text{real}^2 + X_k.\text{imaginary}^2};$$

$$\Phi_k = \text{atan2}\left(X_k.\text{imaginary}, X_k.\text{real}\right),$$

где X_k.real и X_k.imaginary — вещественная и мнимая части числа X_k соответственно.

Эти спектры также будут дискретными с шагом

$$S = \frac{\text{Частота дискретизации}}{\text{Число отсчетов}}.$$

Отсюда очевидно, что увеличение числа отсчетов исходного сигнала дает более точное разрешение по частоте. Однако при постоянной частоте дискретизации, с увеличением числа отсчетов получаем также увеличение рассматриваемого временного отрезка. В музыкальных произведениях ноты имеют различную длительность звучания и могут быстро сменять друг друга. Поэтому при анализе более длительного временного интервала ноты могут накладываться друг на друга, в результате амплитуда длинных нот "закроет" собой амплитуду коротких. Таким образом, в процессе разбора реальной музыкальной композиции имеется ограничение выбора числа отсчетов для рассматриваемого кадра.

Получив необходимую информацию о свойствах сигнала, можно приступить к поиску частот. Одним из наиболее точных методов, используемых для выполнения подобной задачи, является вычисление частоты на основе определения задержки фаз у спектров двух кадров, наложенных друг на друга, но немного сдвинутых во времени. Подробно с принципами работы данного алгоритма можно ознакомиться на соответствующем сайте [9].

В результате, имея точное значение частоты в каждом рассматриваемом кадре, легко сопоставить это значение с известной таблицей частот нот. Определив название звучащей ноты, можно занести ее в партитуру.

Реализация с использованием алгоритма YIN

Данный алгоритм разработан А. Шевейни и Х. Кавахарой в 2001 г. [8]. Он основывается на использовании метода автокорреляции с проведением

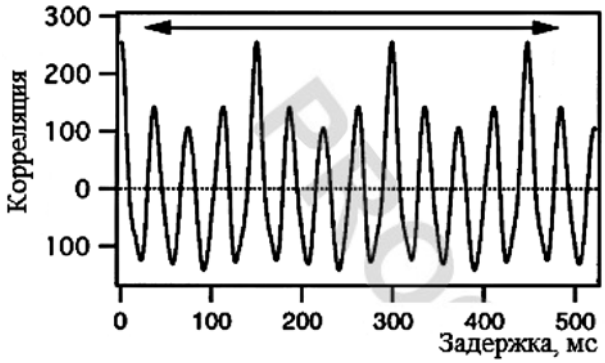


Рис. 4. Пример автокорреляционной функции (1)

некоторых специальных корректировок. По сравнению с другими алгоритмами, использующими метод автокорреляции, он позволяет снизить число ошибок почти в три раза. Данный алгоритм не устанавливает никаких ограничений на максимальную верхнюю частоту, что позволяет успешно применять его при анализе как музыкальных произведений, так и человеческой речи.

Автокорреляционная функция для дискретного сигнала x_t может быть определена следующим образом:

$$r_t(\tau)=\sum_{j=t+1}^{t+W} x_j x_{j+\tau}, \qquad (1)$$

где $r_t(\tau)$ — автокорреляционная функция задержки τ, вычисленная в момент времени t с размером окна W, также известная как модифицированная автокорреляционная функция [10]. Данная функция представлена на рис. 4, она построена на основе сигнала, отображенного на рис. 5.

Зачастую при обработке сигналов используется несколько иная функция [10]:

$$r'_t(\tau)=\sum_{j=t+1}^{t+W-\tau} x_j x_{j+\tau}. \qquad (2)$$

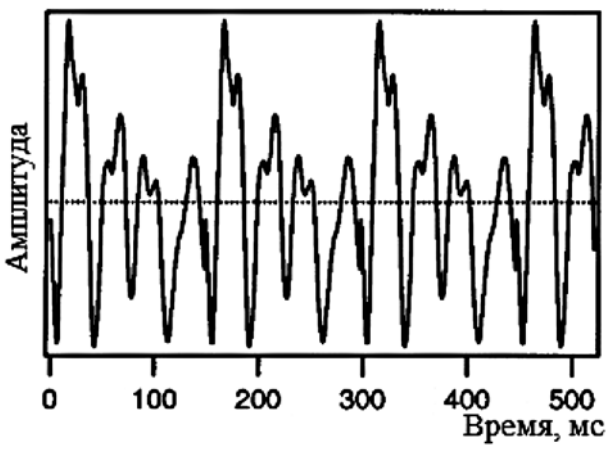


Рис. 5. Пример обрабатываемого сигнала

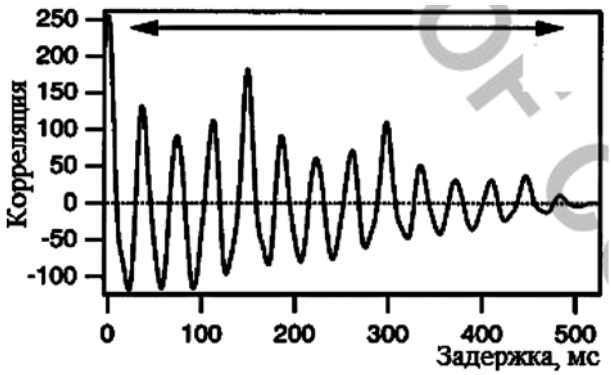


Рис. 6. Пример автокорреляционной функции (2)

В данном варианте размер окна уменьшается с увеличением значения τ. Такая зависимость приводит к тому, что график функции стремится к нулю как функция задержки, представленная на рис 6.

Обе приведенные функции дают одинаковый результат в случае, если сигнал за пределами отрезка $[t + 1, t + W]$ равен нулю, однако разные результаты в противоположном случае. Применяемая к периодическому сигналу автокорреляционная функция находит "пики" (максимальные значения) по всему рассматриваемому периоду. Автокорреляционный метод выбирает наибольшие ненулевые задержки, отыскивая их в диапазонах задержек (горизонтальные отметки на рис. 4 и 6). Очевидно, что если ограничение на минимальную задержку близко к нулю, то алгоритм может ошибочно выбрать нулевую задержку. Напротив, если верхнее ограничение минимальной задержки достаточно большое, то алгоритм может выбрать пик более высокого порядка. Функция (1) устойчива ко второй трудности, тогда как функция (2) устойчива к первой. В итоге алгоритм использует модифицированную функцию

$$r''_t(\tau)=r_t(\tau)(1-\tau/\tau_{\max}), \text{ при } \tau \leqslant \tau_{\max},$$

0 в противном случае.

Создателями алгоритма YIN было экспериментально выяснено, что изменением параметра $\tau_{\max}$ можно контролировать (уменьшать) вероятность одной ошибки в пользу увеличения другой, что позволяет уменьшать вероятность ошибок в целом при заранее известных параметрах сигнала.

Однако, несмотря на этот прием, автокорреляционный метод выдает большое число ошибок при решении практических задач. Поэтому алгоритм YIN использует еще несколько вспомогательных конструкций, существенно снижающих число ошибок по сравнению с базовым автокорреляционным методом. К ним относится применение накопительной нормализующей функции разности, которая использует результаты сравнения обрабатываемого кадра сигнала с кадром, взятым с некоторым сдвигом T, и пороговый механизм, пропускающий только те рассчитанные частоты, для которых значение функции разности меньше некоторого заданного значения погрешности.

Для выяснения вопроса, какой из двух описанных выше методов целесообразно использовать в приложениях, был проведен ряд тестов. Оба метода показывают точность, которая допустима для решения поставленной задачи.

Большинство современных решений концентрируется на распознавании точной частоты, даже с учетом того обстоятельства, что при обработке нот октав, начиная с третьей, большая точность не требуется [7]. С этой целью, видимо, эффективнее использовать преобразование Фурье. Однако при сужении рассматриваемой задачи до распознавания отдельных музыкальных нот данный подход не является обязательным. К тому же алгоритм YIN позволяет масштабировать погрешность, что является серьезным преимуществом, поскольку при распознавании высоких нот допустимые погрешности могут исчисляться сотнями герц.

Таким образом, основным параметром сравнения стало быстродействие двух алгоритмов. Следует отметить, что увеличение скорости обработки кадров позволяет соответственно увеличить точность верного распознавания быстро сменяющих друг друга частот. При обработке сигнала, где временной промежуток между сменой частот по продолжительности стремится к времени обработки отдельного кадра, можно выделить почти в два раза больше звучащих частот. Несмотря на то что столь малое время между сменой нот редко встречается в произведениях, отмеченный результат дает возможность распознавать некоторые технические приемы игры, такие как "слайд" на гитаре. Соответствующая информация может быть добавлена в партитуру.

Результаты проведенных тестов даны в таблице. На основе представленных данных видно явное преимущество алгоритма YIN в плане времени обработки кадров, что и послужило причиной выбора данного метода для реализуемого приложения. Размер кадра 16 бит обусловлен общепринятым стандартом записи для одноканального звука без потерь. Размер кадра 32 бита используется для двухканального звука соответственно [6].

Сравнение скорости работы алгоритмов Фурье и YIN

Характеристики сигнала	Среднее время обработки кадра, мс	
	Алгоритм Фурье	Алгоритм YIN
Частота дискретизации — 8000 Гц Размер кадра — 16 бит	0,1	0,05
Частота дискретизации — 16 000 Гц Размер кадра — 16 бит	0,15	0,08
Частота дискретизации — 48 000 Гц Размер кадра — 16 бит	0,23	0,13
Частота дискретизации — 8000 Гц Размер кадра — 32 бит	0,16	0,07
Частота дискретизации — 16 000 Гц Размер кадра — 32 бит	0,21	0,11
Частота дискретизации — 48 000 Гц Размер кадра — 32 бит	0,29	0,16

Распараллеливая вычисления при обработке кадра, вероятно, можно добиться и большего увеличения скорости обработки.

Архитектура мобильного приложения

В процессе проектирования приложения основные его функциональные возможности были разделены по следующим программным модулям.

- Модуль, применяющий цифровой фильтр к входящему сигналу (Filtering module).
- Модуль, выполняющий YIN-алгоритм на кадре сигнала (YoRHa module — YIN-oriented Recognition Habitat module).
- Модуль, определяющий ноты в кадре сигнала (Note determination module).
- Модуль, отвечающий за конфигурацию параметров распознавания ноты (Note determination settings).
- Модуль, записывающий ноты в файл партитуры (Sheet music writer module).
- Модуль, отвечающий за конфигурацию параметров записи партитур (Sheet music writer settings).

Подобная структура позволяет легко изменять и замещать модули в цепочке обработки сигнала. Так, например, в случае решения перейти к использованию алгоритма Фурье вместо алгоритма YIN, достаточно легко заменить модуль YoRHa на новый, содержащий логику выполнения алгоритма Фурье, после чего добавить модуль применения оконной функции к сигналу. На рис. 7 представлена схема взаимодействия описанных модулей при обработке входящего сигнала.

Filtering module получает на вход кадр исходного сигнала заданного размера, после чего применяет к каждому отсчету кадра передаточную функцию фильтра. В реализации рассматриваемого приложения используется фильтр Баттерворта [11], выбранный опытным путем с целью свести искажения исходного сигнала к минимуму. Таким образом, кадр сначала проходит фильтр низких частот с частотой среза, равной 100 Гц, после чего пропускается через фильтр высоких частот с частотой среза, равной 5400 Гц (частота ноты ми пятой октавы с небольшим запасом).

YoRHa module отвечает за применение алгоритма поиска частоты к входному кадру. Основными характеристиками сигнала для этого модуля являются частота дискретизации, размер кадра и необходимый для алгоритма YIN параметр порога (*threshold*). Последний регулирует расчет вероятности того, что найденная частота является истиной. Практика показывает, что для решения поставленной задачи оптимальным значением данного параметра является 0,2, в то время как частота дискретизации и размер кадра могут устанавливаться в зависимости от ресур-

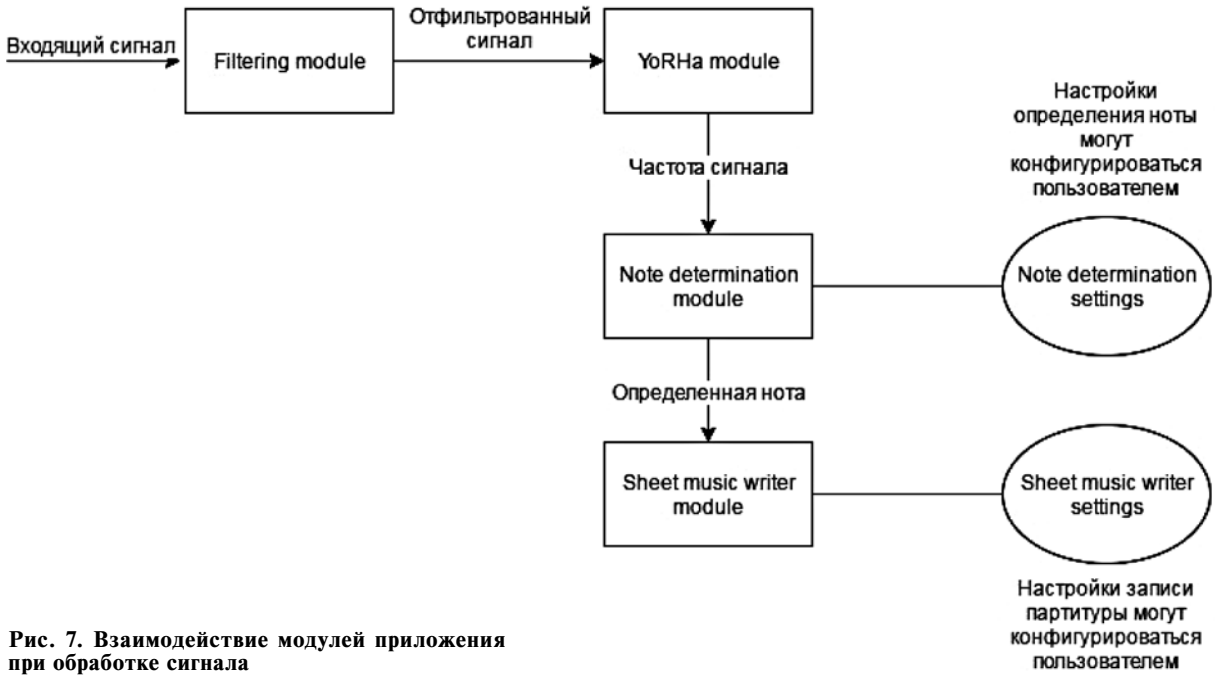


Рис. 7. Взаимодействие модулей приложения при обработке сигнала

сов устройства, на котором приложение запускается. При работе данной программы происходит автоматический опрос системы на предмет максимального размера буфера, в который будет последовательно считываться исходный сигнал. На основе этого размера приложение рассчитывает оптимальную частоту дискретизации, которая, однако, не может быть меньше частоты Найквиста для входящего сигнала.

Note determination module сопоставляет частоту входящего сигнала с частотой музыкальной ноты на основе таблицы частот нот, которая может изменяться пользователем. Данная возможность реализована по той причине, что на протяжении истории частотные параметры нот имели тенденцию к изменению [12]. Корректировки таблицы могут быть проведены как программно, так и через интерфейс приложения, с возможностью сброса настроек к "заводским".

Note determination settings module напрямую взаимодействует с пользовательским интерфейсом, предоставляя возможности изменения следующих параметров распознавания нот.

- Таблица частот нот.
- Параметры отображения нот на экране, если приложение работает в режиме тюнера.
- Способ обозначения нот — классический или буквенный (в свою очередь, разделены на английский и немецкий).

Sheet music writer module реализует создание приложением партитур на основе распознанных нот и их сохранение и/или публикацию. Партитура представляет собой файл, содержащий информацию о числе каждой ноты каждой октавы с временем, в которое они поступили в данный модуль, что позволяет восстановить их расположение в хронологической последовательности. Данная особенность необходима в процессе чтения сохраненных партитур на устройстве. Однако на данный момент этот модуль пока не обладает возможностью устанавливать качественный размер ноты (определять, является ли нота половиной, четвертью, триолью и т. д.).

После окончания процесса записи партитура может быть сохранена во внутренней памяти устройства и/или опубликована путем пересылки файла данных по одному из каналов, доступных на устройстве. Также пользователь имеет возможность оставлять в этом файле цифровую подпись, совпадающую с уникальной подписью приложения, генерирующейся на основе MAC-адреса устройства.

Модуль Sheet music writer settings, как и Note determination settings, взаимодействует с пользовательским интерфейсом, позволяя пользователю управлять следующими параметрами создания партитуры:

- включение или отключение транспозиции нот (сдвиг всех нот партитуры на конкретный музыкальный интервал);
- редактирование личной информации автора (название партитуры, имя автора и т. д.);
- выбор директории для хранения созданных партитур;
- максимальная длительность партитуры (время, после которого активная для записи партитура будет сохранена и запись продолжится в новую).

Заключение

В результате выполненной работы было создано мобильное приложение для платформы Android, удовлетворяющее требованиям производительности и точности при решении задачи определения частотного спектра входного акустического сигнала.

Предварительно были рассмотрены два наиболее популярных подхода к решению данной задачи с последующим сравнением их эффективности в рамках применения на мобильном устройстве. В результате были получены следующие выводы.

- Оба описанных алгоритма удовлетворяют требованиям точности решения поставленной задачи.
- При необходимости повышения точности возможно использование как сторонних аппаратных средств (цифровые и аналоговые фильтры), так и модификации алгоритмов определения частотных спектров применением математических функций (оконные функции).
- Для решения поставленной задачи в реальном времени на мобильном устройстве целесообразно использовать алгоритм YIN как менее требовательный к оперативной памяти и вычислительным ресурсам.

Результаты проведенного исследования обосновывают целесообразность перехода музыкальных приложений, анализирующих частотный спектр аудиосигнала, с использования алгоритма Фурье на применение алгоритма YIN. Таким образом, решается задача, связанная с высокими требованиями к вычислительным ресурсам и продолжительным временем работы. В частности, современные приложения для распознавания звучащей последовательности аккордов или определения тональности воспроизводимого музыкального произведения могут занимать значительную часть свободных ресурсов операционной системы, затрудняя или вообще делая невозможной работу других приложений на достаточно продолжительное время.

В дальнейшем планируется доработка приложения в целях реализации функции распознавания целостных аккордов. В качестве метода решения возможно дополнение программы моделью нейронной сети, обученной для распознавания различных музыкальных созвучий. Один из подходов описан, например, в работе Н. Боулангера-Левандоски, посвященной распознаванию аккордов с помощью рекуррентных нейросетей [13]. Предложенное им решение обусловлено невозможностью однозначно определить аккорд, распознав лишь звучащие ноты. Причина в том, что один и тот же аккорд может иметь несколько обращений, т. е. несколько наборов звучащих нот с разной частотой могут образовывать один и тот же аккорд [13]. С учетом этой информации обучение нейронной сети всевозможным вариантам обращений аккордов представляется достаточно простым и эффективным направлением для расширения возможностей системы.

Список литературы

1. **В ТУСУРе** создали программу, способную преобразовать напетую мелодию в ноты с минимальной погрешностью // Томский государственный университет систем управления и радиоэлектроники. URL: https://tusur.ru/ru/novosti-i-meropriyatiya/novosti/prosmotr/-/novost-v-tusure-sozdali-programmu-sposobnuyu-preobrazovat-napetuyu-melodiyu-v-noty-s-minimalnoy (дата обращения 15.12.2017).
2. **What is** Melodayne. URL: http://www.celemony.com/en/melodyne/what-is-melodyne (дата обращения 23.12.2017).
3. **PRAAT** Short Tutorial. URL: https://web.stanford.edu/dept/linguistics/corpora/material/PRAAT_workshop_manual_v421.pdf (дата обращения 23.12.2017).
4. **DoReMIR** Music Research AB. URL: http://scorecloud.com (дата обращения 15.12.2017).
5. **Баскаков С. И.** Радиотехнические цепи и сигналы, 2-е издание. М.: Высшая школа, 1998. 446 с.
6. **Кирн П.** Цифровой звук. Реальный мир. М.: Вильямс, 2008. 713 с.
7. **Сато Ю.** Обработка сигналов, первое знакомство. М.: Додэка-XXI, 2010. 178 с.
8. **De Cheveigné A., Kawahara H.** YIN, a fundamental frequency estimator for speech and music // The Journal of the Acoustical Society of America. 2002. Vol. 111, Issue 4 URL: http://asa.scitation.org/doi/abs/10.1121/1.1458024 (дата обращения: 14.12.2017).
9. **Guitar** Pitch Shifter. URL: http://www.guitarpitchshifter.com (дата обращения: 15.11.2017).
10. **Rabiner L. R., Schafer R. W.** Digital Processing of Speech Signals. Bell Laboratories, 1978. 256 p.
11. **Титце У., Шенк К.** Полупроводниковая схемотехника. Пер. с нем.: Справочное руководство. М.: Мир, 1982. 512 с.
12. **Вахромеев В. А.** Элементарная теория музыки. М.: Государственное музыкальное издательство, 1961. 244 с.
13. **Boulanger-Lewandowski N., Bengio Y., Pascal V.** Audiochord recognition with recurrent neural networks// In Proceedings of the International Society for Music Information Retrieval Conference (ISMIR). 2013.URL: http://www-etud.iro.umontreal.ca/~boulanni/ISMIR2013.pdf (дата обращения 10.12.2017).

A Mobile Application for Frequency Recognition and Fetching Notes from Acoustic Signal

S. D. Makhortov, msd_exp@outlook.com, **I. V. Kleymenov,** alenor96@gmail.com, Voronezh State University, Voronezh, 394018, Russian Federation

Corresponding author:

Makhortov Sergey D., Professor, Head of Department, Voronezh State University, Voronezh, 394018, Russian Federation,
E-mail: msd_exp@outlook.com

Received on December 15, 2017
Accepted on December 27, 2017

This article describes common approaches to frequency and notes recognition on mobile devices and demonstrates advantages of YIN algorithm for solving this problem. Considering the fact that mobile devices have less memory and less processor power, it is important to find a solution that allows performing computation with low resource requirement. In addition, for proper musical notes fetching the fast run-time environment is needed. Usually Fast Fourier Transformation is used for solving this problem but it has to perform many difficult mathematic operations that may take a lot of time. This article proves practicability of using a different method — YIN algorithm. The YIN was invented by Alain de Cheveigne and Hideki Kawahara in 2002 as an alternative to well-known methods such as Fast Fourier Transformation (FFT). This algorithm is based on autocorrelation method with several essential improvements which makes it possible to decrease error probability in acoustic signal processing. As at the moment of writing there are no applications for Android OS that could reliably perform frequency to notes transformation with following music sheet creation, an attempt was made to develop such an application. This paper describes application architecture and provides detail about the responsibility of each module. Article presents comparison of YIN and FFT algorithms based on speed that was measured on an Android device with multiple input acoustic signal configurations. These results prove the practicability of YIN usage as it provides faster calculation with less memory resource consumption.

Keywords: *frequency spectrum, YIN, acoustic signal, notes recognition, mobile application*

For citation:

Makhortov S. D., Kleymenov I. V. A Mobile Application for Frequency Recognition and Fetching Notes from Acoustic Signal, *Programmnaya Ingeneria,* 2018, vol. 9, no. 3, pp. 132—139.

DOI: 10.17587/prin.9.132-139

References

1. **V TUSURe** sozdali programmu, sposobnuju preobrazovat' napetuju melodiju v noty s minimal'noj pogreshnost'ju (In TUSUR have created a program that can convert hum the melody in notes with minimum error), *Tomskij gosudarstvennyj universitet sistem upravlenija i radiojelektroniki*, 2014, available at: https://tusur.ru/ru/novosti-i-meropriyatiya/novosti/prosmotr/-/novost-v-tusure-sozdali-programmu-sposobnuyu-preobrazovat-napetuyu-melodiyu-v-noty-s-minimalnoy.

2. **What is Melodayne**, 2017, available at: http://www.celemony.com/en/melodyne/what-is-melodyne.

3. **PRAAT** Short Tutorial. 2003, available at: https://web.stanford.edu/dept/linguistics/corpora/material/PRAAT_workshop_manual_v421.pdf.

4. **DoReMIR** Music Research AB. 2017, available at: http://scorecloud.com

5. **Baskakov S. I.** *Radiotehnicheskie cepi i signaly, 2-e izdanie* (Radio-technic circuits, 2nd edition), Moscow, Vysshaja shkola, 1998, 446 p. (in Russian).

6. **Kirn P.** *Cifrovoj zvuk. Real'nyj mir* (Digital sound. Real world), Moscow, Izdatel'skij dom Willams, 2008, 713 p. (in Russian).

7. **Sato Y.** *Obrabotka signalov, pervoe znakomstvo* (Signal Processing. Getting Started), Moscow, Dodjeka-XXI, 2010, 178 p. (in Russian).

8. **De Cheveigné A., Kawahara H.** YIN, a fundamental frequency estimator for speech and music, *The Journal of the Acoustical Society of America*, 2002, vol. 111, issue 4, available at: http://asa.scitation.org/doi/abs/10.1121/1.1458024

9. **Guitar** Pitch Shifter. 2017, available at: http://www.guitar-pitchshifter.com

10. **Rabiner L. R., Schafer R. W.** Digital Processing of Speech Signals, Bell Laboratories, USA, 1978, 256 p.

11. **Tietze U., Schenk Ch.** *Poluprovodnikovaja shemotehnika* (Electronics circuits design application), Moscow, Mir, 1982, 512 p. (in Russian).

12. **Vahromeev V. A.** *Jelementarnaja teorija muzyki* (Basic music theory), Moscow, Gosudarstvennoe muzykal'noe izdatel'stvo, 1961, 244 p. (in Russian).

13. **Boulanger-Lewandowski N., Bengio Y., Pascal V.** Audio-chord recognition with recurrent neural networks, *In Proceedings of the International Society for Music Information Retrieval Conference (ISMIR),* 2013, available at: http://www-etud.iro.umontreal.ca/~boulanni/ISMIR2013.pdf

УДК 004.932.4 DOI: 10.17587/prin.9.140-144

С. З. Свердлов, канд. техн. наук, доц., e-mail: c3c@uni-vologda.ac.ru,
Вологодский государственный университет

Автоматическая настройка резкости при уменьшении цифровых изображений

Предложен алгоритм и программная реализация автоматической настройки резкости при уменьшении растровых цифровых изображений. Алгоритм основан на ограничении уровня артефактов, возникающих при выходе цвета за охват цветового пространства. Для выбора параметра настройки резкости используется кубическая аппроксимация зависимости доли выходящих за охват цветовых значений от коэффициента усиления.

Ключевые слова: *обработка изображений, цифровая фотография, масштабирование, растровое изображение, программное обеспечение, резкость*

Введение

При уменьшении растровых цифровых изображений требуется усиление резкости [1]. В работе [2] представлен эффективный алгоритм масштабирования изображений и реализующая его программа [3][1], которые обеспечивают высококачественное уменьшение изображений с настройкой резкости. Настройка выполняется с помощью движка, имеющего несколько фиксированных положений (рис. 1 см. вторую сторону обложки).

При обработке серии фотографий, например, с использованием Actions (операций — средства записи и последующего воспроизведения последовательности действий) в программе Adobe Photoshop, установка резкости вручную для каждой фотографии снижает оперативность работы. Установка же фиксированного уровня усиления резкости для всех фотографий серии не позволяет учесть особенности каждого снимка. В связи с этим возникает потребность в алгоритме и его программной реализации, позволяющих автоматически выбирать уровень усиления резкости и учитывающих особенности конкретной фотографии.

Принцип автоматической настройки резкости

Ограничением при усилении резкости является появление артефактов, которые проявляют себя в виде темных и светлых ореолов, зрительного утолщения тонких линий, появлении очень светлых и очень темных точек.

На рис. 3 (см. вторую сторону обложки) представлен в увеличенном виде фрагмент изображения, показанного на рис. 2 (см. вторую сторону обложки). Фрагменты отличаются уровнем усиления резкости. Настройка резкости выполнена с помощью программы C3C Image Size версии 3.1. Увеличение выполнено с помощью алгоритма "ближайшего соседа".

В варианте с умеренным усилением резкости (рис. 3, *б*) артефакты отсутствуют. На рис. 3, *в* артефакты усиления резкости уже заметны, например, на ресницах нижнего века. Вариант, представленный на рис. 3, *г*, очевидно, неприемлем.

При усилении резкости пиксели, которые темнее своего окружения, становятся еще темней, а те, что светлей окружающих пикселей, становятся светлее. Это может приводить к тому, что цвет пикселей после преобразования, связанного с усилением резкости, выходит за цветовой охват. В случае использования цветовой координатной системы RGB границы охвата определяются диапазоном [0...1] для значений красной, зеленой и синей компонент (цветовых координат R, G и B соответственно), если используется их вещественное представление. Выход за охват означает ситуацию, когда значения R, G или B оказываются отрицательными или превышают единицу. В случае выхода за охват в действие вступают алгоритмы отсечения, что может искажать цветовой тон пикселей, приводить к слиянию отличающихся цветов. Такие искажения проявляют себя как неприемлемые артефакты усиления резкости.

Алгоритм усиления резкости может применяться к значению (каналу, в терминах Adobe Photoshop) светлоты, которая вычисляется [4, 5] по формуле

$$L = 0{,}2125R + 0{,}7152G + 0{,}0722B.$$

Использование светлоты, а не значений R, G и B, сохраняет идею алгоритма, но позволяет сохранить цветовой тон и насыщенность пикселей изображения.

В обсуждаемом далее алгоритме автоматической настройки резкости предлагается использовать такое усиление, которое приводит к выходу за цветовой

[1] Программа доступна по адресу http://www.univologda.ac.ru/~c3c/plug-ins/c3cimagesize.htm

охват заданной ограниченной доли пикселей изображения. В рассмотренной ниже реализации алгоритма указанная доля составляет 0,1 % пикселей изображения. Как показывает опыт использования программы C3C Image Size, это соответствует деликатному усилению резкости, исключающему появление нежелательных артефактов.

Реализация автоматического усиления резкости

Обсуждаемый способ автоматической настройки резкости реализован в программе C3C Image Size, начиная с версии 3.0.

Алгоритм усиления резкости

Преобразования, связанные с усилением резкости, выполняются в линейном цветовом пространстве [4, 6]. Усиление резкости применяется к каналу светлоты. Расчет для строк изображения выполняется по формуле

$$y_i = \alpha x_i + (1-\alpha)\frac{x_{i-1} + x_i + x_{i+1}}{3};\ i \in [1,\ n-2], \qquad (1)$$

где α — сила эффекта; $\alpha > 1$ соответствует усилению резкости; $\alpha < 1$ — ослаблению. x_{i-1}, x_i, x_{i+1} — значения светлоты трех очередных пикселей исходной (до усиления резкости) строки; y_i — значение светлоты i-го пикселя после преобразования; n — число пикселей в строке. Обработка по столбцам выполняется аналогично. Коррекция цветовых координат R, G, B выполняется по формуле

$$C_{\text{нов.}} = C_{\text{стар.}}\frac{L_{\text{нов.}}}{L_{\text{стар.}}},$$

где $C_{\text{нов.}}$, $C_{\text{стар.}}$ — новое (после усиления резкости) и старое значения цветовых координат R, G и B; $L_{\text{нов.}}$, $L_{\text{стар.}}$ — новое (после усиления резкости) и старое значения светлоты. Такой расчет, когда цветовые координаты R, G и B умножаются на один и тот же коэффициент, не меняет цветовой тон и насыщенность цвета пикселя при коррекции резкости [4].

В программе C3C Image Size (см. рис. 1) параметр *Sharpness* (резкость) задается по логарифмической шкале. Значение *Sharpness*, равное 0, соответствует отсутствию усиления резкости ($\alpha = 1$), значение 1 соответствует $\alpha = \sqrt[4]{2}$ в формуле (1). Поскольку обработка в соответствии с формулой (1) выполняется дважды — вначале по строкам, а затем по столбцам, один шаг в настройке резкости соответствует изменению параметра α в $\sqrt{2}$ раз.

Зависимость доли пикселей, выходящих за цветовой охват, от значения Sharpness

Для исследования зависимости числа пикселей, цвет которых выходит за цветовой охват, и их доли по отношению ко всем пикселям изображения использованы три цифровые фотографии: портрет (рис. 2, см. вторую сторону обложки); пейзаж (рис. 4, см. третью сторону обложки); фотография памятника архитектуры (рис. 5, см. третью сторону обложки).

Фотографии масштабировались до размера, соответствующего экранному разрешению Full HD, — 1920 × 1080 пикселей — картинка вписывается в указанный прямоугольник. Значение *Sharpness* в программе C3C Image Size изменялось от 0 (отсутствие усиления резкости) до 8 (максимальное усиление). В программу встроен счетчик, с помощью которого вычисляется число пикселей уменьшенного изображения, светлота (а следовательно, хотя бы одна из цветовых координат) которых выходит за охват. Затем это число делится на площадь масштабированного изображения в пикселях и определяется доля (процент) пикселей, цвет которых выходит за охват. Указанный процент в дальнейшем будем обозначать P.

На рис. 6 показана зависимость, полученная в ходе обработки трех упомянутых фотографий.

Как можно заметить, и значение P, и характер ее зависимости от *Sharpness* существенно различаются для разных изображений. Так, для портрета, использованного в эксперименте, для значений *Sharpness* $\geqslant 3$ зависимость практически линейна. Для фотографии архитектурного памятника зависимость напоминает параболическую, а в случае пейзажа для значений *Sharpness* $\geqslant 4$ эта зависимость имеет выпуклость вверх.

В ходе предварительных исследований выяснилось, что можно удовлетворительно аппроксимировать каждую такую зависимость полиномом пятой степени, используя девять экспериментально полученных точек.

Алгоритм автоматического определения настройки резкости

Задача состоит в том, чтобы выбрать такое значение *Sharpness*, которое дает заданное значение $P = P_0$ (например, $P_0 = 0{,}1$ %). Это приводит к необходимости решать уравнение

$$P(Sharpness) = P_0 \qquad (2)$$

относительно неизвестной *Sharpness*.

Если использовать аппроксимацию левой части уравнения (2) полиномом пятого порядка, то потребуется решать уравнение пятой степени, что можно сделать численными методами без существенных затрат времени. Однако для получения коэффициентов аппроксимирующего многочлена необходимо 8 раз выполнить настройку резкости изображения при восьми различных значениях *Sharpness*, что сопряжено с неприемлемыми, как представляется, временными затратами.

Другой подход мог бы состоять в том, что уравнение (2) решается численно без использования аппроксимации. Однако и в этом случае для достижения приемлемой точности также потребовалось бы многократное определение значения левой части уравнения (2), что каждый раз сопряжено с коррекцией резкости изображения при заданном значении *Sharpness*. Так, для достижения точности 0,1 при

исходной длине интервала значений *Sharpness*, равной 8, потребовалось бы 7 раз выполнять коррекцию резкости при использовании метода половинного деления для решения уравнения.

В связи с изложенными обстоятельствами принят следующий подход при решении задачи. Рассматривается лишь диапазон значений *Sharpness* от 0 до 4. Статистика использования программы C3C Image Size (рис. 7) показывает, что в подавляющем большинстве случаев (97,5 %) пользователи, если не используют автоматическую настройку резкости, применяют значение параметра *Sharpness* из этого диапазона.

На интервале от 0 до 4 зависимость P(*Sharpness*) аппроксимируется полиномом по трем точкам (*Sharpness* = 0; 2; 4). Поскольку по очевидным причинам $P(0) = 0$, для этого необходимо выполнить предварительную коррекцию резкости только 2 раза. Аппроксимация (интерполяция) выполняется полиномом третьей степени с дополнительным условием равенства нулю его производной в точке *Sharpness* = = 0. Это условие уменьшает вероятность получения отрицательных значений аппроксимирующего полинома.

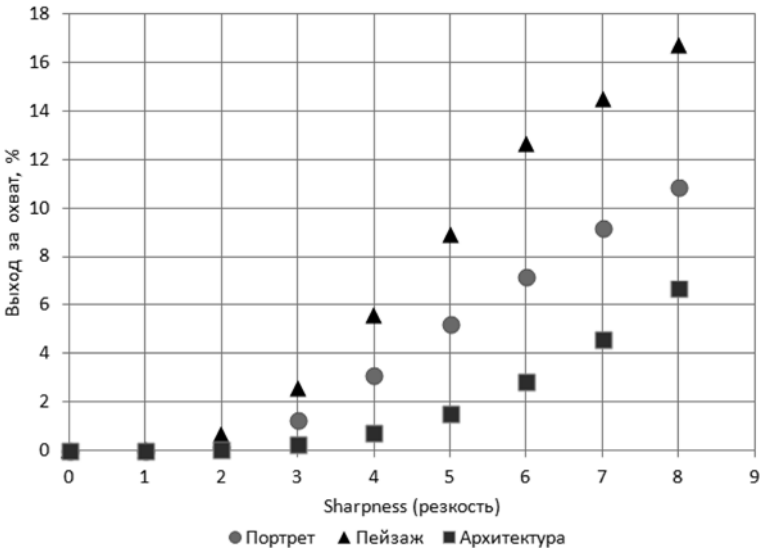


Рис. 6. Зависимость процента пикселей *P*, цвет которых выходит за цветовой охват, от параметра резкости *Sharpness*

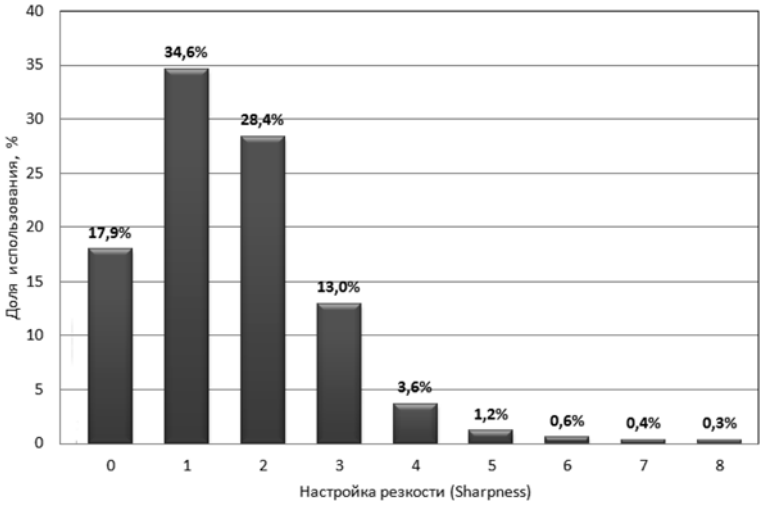


Рис. 7. Статистика использования настройки резкости пользователями программы C3C Image Size

Обозначим x_1 и x_2 отличные от нуля значения аргумента аппроксимирующего многочлена $f(x)$ в точках, в которых определены значения функции, а y_1 и y_2 — значения функции в этих точках. В нашем случае переменная x соответствует значениям *Sharpness*, а y — значениям P. Для определения коэффициентов аппроксимирующего многочлена можно записать следующие уравнения:

$$\begin{cases} f(x_1) = ax_1^3 + bx_1^2 + c = y_1; \\ f(x_2) = ax_2^3 + bx_2^2 + c = y_2; \\ f'(0) = c = 0. \end{cases} \quad (3)$$

Решая систему уравнений (3), получаем выражения для коэффициентов аппроксимирующего многочлена:

$$a = \frac{y_1 x_2^2 - y_2 x_1^2}{x_1^3 x_2^2 - x_2^3 x_1^2};$$
$$b = \frac{y_2 x_1^3 - y_1 x_2^3}{x_1^3 x_2^2 - x_2^3 x_1^2}; \quad (4)$$
$$c = 0.$$

С учетом (4) аппроксимирующий многочлен имеет вид

$$f(x) = ax^3 + bx^2. \quad (5)$$

На рис. 8 показаны результаты кубической аппроксимации зависимости P(*Sharpness*) для трех рассмотренных выше изображений.

Как видно на приведенных графиках, построенный по предложенной методике полином аппроксимирует зависимость P(*Sharpness*) с точностью, удовлетворительной для практических надобностей.

Последовательность действий, которые выполняет программа (рис. 9) при автоматической настройке резкости, включает перечисленные далее шаги.

1. Выполняется коррекция резкости при значениях *Sharpness* 2 и 4 ($x_1 = 2$; $x_2 = 4$). Тем самым измеряются значения $y_1 = P(2)$ и $y_2 = P(4)$.

2. По формулам (4) вычисляются коэффициенты a и b аппроксимирующего многочлена (5).

3. Уравнение (3) с левой частью вида (5) решается численно, определяется значение *Sharpness* для автоматической настройки резкости.

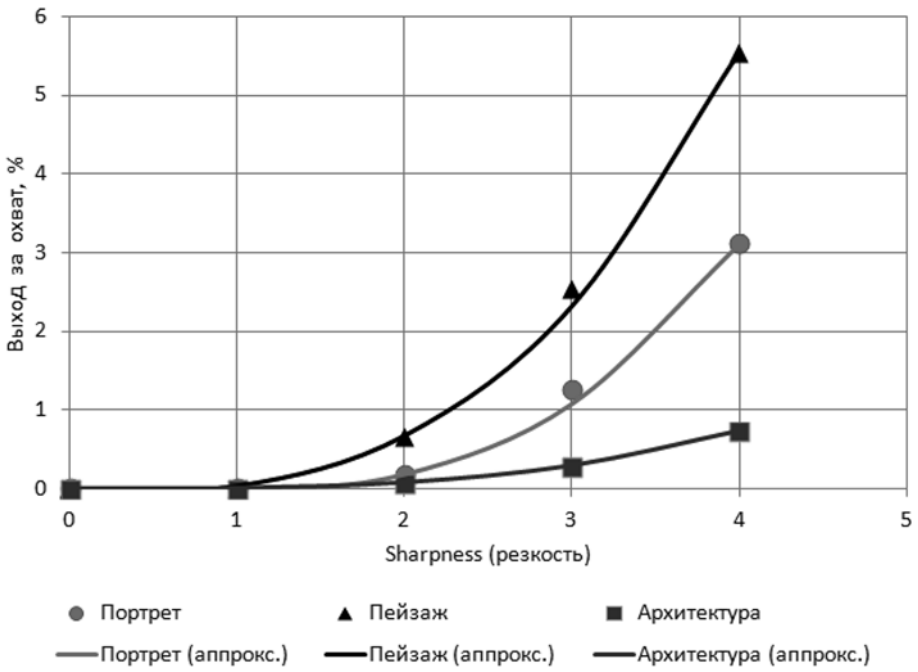


Рис. 8. Аппроксимация зависимости *P*(*Sharpness*) полиномом третьей степени

Рис. 9. Автоматическая настройка резкости в программе C3C Image Size

Изображение	Значение *Sharpness*	Фактическое значение *P*, %
Портрет	1,81	0,097
Пейзаж	1,09	0,057
Архитектура	2,17	0,096

4. Выполняется коррекция резкости при найденном значении *Sharpness*.

В таблице представлены результаты, полученные при автоматической настройке резкости для рассмотренных изображений при $P_0 = 0{,}1$ %.

Из данных таблицы видно, что в двух случаях точность аппроксимации оказывается очень высокой (0,096 % против целевых 0,1 %), в случае пейзажа погрешность выше, но она не представляется критичной.

Заключение

Опыт использования программы C3C Image Size, в которой реализован описанный алгоритм, показал, что автоматическая настройка является самым используемым вариантом усиления резкости. По статистике запусков программы (70 776 запусков в течение полугода) в 34,1 % случаев, когда пользователи применяли усиление резкости, они выбирали автоматическую коррекцию. Это, по мнению автора, свидетельствует о хорошей работе предложенного метода.

Интерес, проявленный к разработке, уже сегодня свидетельствует о ее востребованности на практике и позволяет надеяться на хорошие перспективы в будущем.

Список литературы

1. **Adobe** Systems Incorporated. Photoshop User Guide. URL: https://helpx.adobe.com/photoshop/user-guide.html.

2. **Свердлов С. З.** Алгоритм и программа для уменьшения цифровых изображений // Программная инженерия. 2016. Т. 7, № 5. С. 231—237.

3. **Свердлов С.** C3C Image Size. Высококачественное уменьшение изображений. Свидетельство о государственной регистрации программы для ЭВМ № 2016617493 от 06.06.2017.

4. **Свердлов С. З.** Ортогональная цветовая координатная система // Вестник Вологодского государственного педагогического университета. 2013. Т. 5. С. 25—29.

5. **Wyszecki G., Stiles W. S.** Color Science. Concepts and Methods, Quantitative Data and Formulae. Second Edition, Wiley-Interscience Publication, 2000. 968 p.

6. **INTERNATIONAL** STANDARD IEC 61966-2-1 ed1.0. Multimedia systems and equipment — Colour measurement and management — Part 2-1: Colour management — Default RGB colour space — sRGB.

Automatic Sharpness Adjustion when Reducing Digital Images

S. Z. Sverdlov, c3c@uni-vologda.ac.ru, Vologda State University, Vologda, 160600, Russian Federation

Corresponding author:

Sverdlov Sergey Z., Associate Professor, Vologda State University, Vologda, 160600, Russian Federation. E-mail: c3c@uni-vologda.ac.ru

Received on January 8, 2018
Accepted January 15, 2018

When you reduce raster digital images, you need a sharpening effect. When processing a series of photos (for example, using Actions in Adobe Photoshop), setting the sharpness manually for each photo reduces the responsiveness of the work. Setting the fixed gain of sharpness for all photos in the series does not allow to take the features of each picture into account. In this connection, there is a need for an algorithm that automatically selects the level of sharpening and takes into account the specific features of a particular photograph.

The limitation of sharpening is the appearance of artifacts that manifest themselves in the form of dark and light haloes, visual thickening of fine lines, the appearance of very light and very dark dots. With sharpening, pixels that are darker than their surroundings become even darker, those that are lighter than surrounding pixels become lighter. This can lead to the fact that the color of the pixels after the conversion, associated with the sharpening, goes beyond the color gamut. When exiting for gamut, clipping algorithms come into play, which can distort the color tone of the pixels, lead to the fusion of different colors. Such distortions manifest themselves as unacceptable artifacts of sharpening.

The presented algorithm for automatic adjustment of the sharpness of raster digital images is based on limiting the level of artifacts that occur when the color leaves behind the gamut of the color space. To select the sharpness adjustment parameter, a cubic approximation is used for the dependence of the proportion of the color values beyond the coverage on the gamut.

Keywords: *Image processing, digital photo, resize, raster graphics, software, plug-ins, sharpness*

For citation:

Sverdlov S. Z. Automatic Sharpness Adjustion when Reducing Digital Images, *Programmnaya Ingeneria,* 2018, vol. 9, no. 3, pp. 140—144.

DOI: 10.17587/prin.9.140-144

References

1. **Adobe** Systems Incorporated. Photoshop User Guide, available at: https://helpx.adobe.com/photoshop/user-guide.html.

2. **Sverdlov S. Z.** Algoritm i programma dlja umen'shenija cifrovyh izobrazhenij (Algorithm and Program to Downsizing the Digital Images), *Programmnaya Ingeneria*, 2016, vol. 7, no. 5, pp. 231—237 (in Russian).

3. **Sverdlov S.** C3C Image Size. Vysokokachestvennoe umen'shenie izobrazhenij. Svidetel'stvo o gosudarstvennoj registracii programmy dlja JeVM № 2016617493 ot 06.06.2017 (in Russian).

4. **Sverdlov S. Z.** Ortogonal'naja cvetovaja koordinatnaja Sistema (Orthogonal color coordinate system), *Vestnik Vologodskogo gosudarstvennogo pedagogicheskogo universiteta*, 2013, vol. 5, pp. 25—29 (in Russian).

5. **Wyszecki G., Stiles W. S.** *Color Science. Concepts and Methods, Quantitative Data and Formulae*, Second Edition, Wiley-Interscience Publication, 2000, 968 p.

6. **INTERNATIONAL** STANDARD IEC 61966-2-1 ed1.0. Multimedia systems and equipment — Colour measurement and management — Part 2-1: Colour management — Default RGB colour space — sRGB.

ООО "Издательство "Новые технологии". 107076, Москва, Стромынский пер., 4
Технический редактор *Е. М. Патрушева.* Корректор *Е. В. Комиссарова*

Сдано в набор 15.01.2018 г. Подписано в печать 22.02.2018 г. Формат 60×88 1/8. Заказ PI318
Цена свободная.

Оригинал-макет ООО "Адвансед солюшнз". Отпечатано в ООО "Адвансед солюшнз".
119071, г. Москва, Ленинский пр-т, д. 19, стр. 1. Сайт: **www.aov.ru**

Рисунки к статье С. З. Свердлова
«АВТОМАТИЧЕСКАЯ НАСТРОЙКА РЕЗКОСТИ
ПРИ УМЕНЬШЕНИИ ЦИФРОВЫХ ИЗОБРАЖЕНИЙ»

**Рис. 4. К северу от Вологды. Пейзаж
(исходное разрешение 4912×3027 пикселей)**

**Рис. 5. Псков. Архитектура
(исходное разрешение 5513×3101 пикселей)**

XX Всероссийская конференция

НАУЧНЫЙ СЕРВИС В СЕТИ ИНТЕРНЕТ

с 17 по 22 сентября 2018 г.

Конференцию проводит
Институт прикладной математики
им. М.В. Келдыша РАН

Конференция посвящена основным направлениям и тенденциям использования интернет-технологий в современных научных исследованиях. Основная цель конференции — предоставить возможность для обсуждения, апробации и обмена мнениями о наиболее значимых результатах, полученных ведущими российскими учеными за последнее время в данной области деятельности.

Конференция серии «Научный сервис в сети Интернет» проводится в двадцатый раз. В 2017 г. в ее работе приняли участие свыше 150 человек.

Тематика конференции

- Научные исследования и интернет, интернет-представительство научных организаций и проектов.
- Решение задач и обработка данных на суперкомпьютерах центров коллективного пользования.
- Интернет-проекты в области параллельных вычислений, математическое моделирование, вычислительные сервисы.
- Интернет-проекты для биомедицины.
- Модели и методы построения поисковых систем и систем навигации в интернете, технологии и системы распределенного хранения и обработки данных.
- Технологии и опыт построения информационных систем и баз данных, документации и результатов эксперимента на основе интернет-технологий.
- Цифровые библиотеки и библиографические базы, семантический веб, наукометрия в интернете.
- Онлайновая научная публикация, открытая наука, живая публикация, онлайновое рецензирование, мультимедийные иллюстрации.
- Популярный научный интернет, онлайновые энциклопедии, история науки в интернете.
- Интернет-активность ученого, персональная страница, профили ученого в библиографических базах, аттестация в интернете.
- Системное и инструментальное программное обеспечение, языки и модели программирования, формальные методы для интернет-технологий.

Конференция проводится с 17 по 22 сентября в пансионате «Моряк», расположенном в 20 километрах от Новороссийска в живописном месте на берегу Черного моря недалеко от поселка Дюрсо.

Сайт конференции: http://agora.guru.ru/abrau2018