

ISSN 1816-0301

# ИНФОРМАТИКА

**2(34)**

**АПРЕЛЬ-ИЮНЬ**

**2012**

## **Редакционная коллегия:**

*Главный редактор*

**А.В. Тузиков**

*Заместитель главного редактора*

**М.Я. Ковалев**

*Члены редколлегии*

С.В. Абламейко, В.В. Анищенко, П.Н. Бибило, М.Н. Бобов,  
А.Н. Дудин, А.Д. Закревский, С.Я. Килин, В.В. Краснопрошин,  
С.П. Кундас, Н.А. Лиходед, П.П. Матус, С.В. Медведев, А.А. Петровский,  
Ю.Н. Сотсков, Ю.С. Харин, А.Ф. Чернявский, В.Н. Ярмолик  
Н.А. Рудая (*заведующая редакцией*)

---

*Адрес редакции:*

220012, Минск,  
ул. Сурганова, 6, к. 305  
тел. (017) 284-26-22  
e-mail: [rio@newman.bas-net.by](mailto:rio@newman.bas-net.by)  
<http://uiip.bas-net.by>

ОБЪЕДИНЕННЫЙ ИНСТИТУТ ПРОБЛЕМ ИНФОРМАТИКИ  
НАЦИОНАЛЬНОЙ АКАДЕМИИ НАУК БЕЛАРУСИ

---

---

# ИНФОРМАТИКА

---

---

ЕЖЕКВАРТАЛЬНЫЙ НАУЧНЫЙ ЖУРНАЛ

*Издается с января 2004 г.*

---

---

№ 2(34) • апрель-июнь 2012

## СОДЕРЖАНИЕ

### ОБРАБОТКА ИЗОБРАЖЕНИЙ, СИГНАЛОВ И РЕЧИ

<b>Ковалев В.А., Дмитрук А.А., Сафонов И.В.</b> Метод поиска связей между морфологическими структурами гистологических изображений и показателями состояния онкологических больных .....	5
<b>Прадун Д.В., Мамчич А.А.</b> Алгоритм автоматического определения теней с использованием энтропийного метода бинаризации изображений .....	12
<b>Кравчонок А.И.</b> Алгоритм быстрого вычисления оптического потока при помощи SSE2-инструкций процессоров семейства x86 .....	19
<b>Прокопович Г.А.</b> Применение гетероассоциативных нейронных сетей для записи и восстановления информации .....	38
<b>Вашкевич М.И., Азаров И.С., Петровский А.А.</b> Подавление эффекта акустической обратной связи в слуховых аппаратах с использованием неравнополосного банка фильтров .....	50
<b>Цирульник Л.И., Покладок Д.А.</b> Грамматический словарь и правила определения словесного ударения для синтеза речи по тексту на мобильных устройствах .....	61

### ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

<b>Бибило П.Н., Соловьев А.Л.</b> Функциональные VHDL-модели элементов FPGA семейства Spartan 3 для конвертации проектов цифровых систем в заказные СБИС .....	69
<b>Новиков Д.Я., Черемисинова Л.Д.</b> Импликативный метод анализа частичных булевых функций .....	79
<b>Золоторевич Л.А.</b> Построение тестов и верификация потоковых моделей цифровых устройств на языке VHDL .....	87

## МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

<b>Еремина А.Р.</b> Инвариантность стационарного распределения состояний неоднородных сетей массового обслуживания с многорежимными стратегиями и дисциплиной DPS .....	98
<b>Емеличев В.А., Коротков В.В.</b> Анализ устойчивости парето-оптимального портфеля проектов бикритериальной инвестиционной задачи с критериями Вальда и Сэвиджа .....	107

---

---

Редактор Г.Б. Гончаренко  
Корректор А.А. Михайлова  
Компьютерная верстка О.Я. Леоник

---

Сдано в набор 03.05.2012. Подписано в печать 05.06.2012.  
Формат 60×84 1/8. Бумага офсетная. Гарнитура Таймс.  
Усл. печ. л. 13,7. Уч.-изд. л. 13,4. Тираж 100 экз. Заказ 9.

---

Государственное научное учреждение «Объединенный институт проблем информатики Национальной академии наук Беларуси».  
ЛИ № 02330/0549421 от 08.04.2009.  
Ул. Сурганова, 6, 220012, Минск.

Отпечатано с оригинала-макета на ризографе Объединенного института проблем информатики Национальной академии наук Беларуси.  
Ул. Сурганова, 6, 220012, Минск.

# INFORMATICS

---

---

PUBLISHED QUATERLY

*Issued since 2004*

---

---

№ 2(34) • April-June 2012

## CONTENTS

### IMAGE, SIGNAL AND SPEECH PROCESSING

<b>Kovalev V.A., Dmitruk A.A., Safonau I.U.</b> A method for identification and visualization of histological image structures relevant to the cancer patient conditions .....	5
<b>Pradun D.V., Mamchich A.A.</b> The algorithm of automatic shadow detection using entropic thresholding method .....	12
<b>Kravchonok A.I.</b> Fast optical flow computation algorithm based on SSE2-instructions of x86 processor family .....	19
<b>Prakapovich R.A.</b> Heteroassociative neural networks application for data recording and recovery .....	38
<b>Vashkevich M.I., Azarov E.S., Petrovsky A.A.</b> Acoustic feedback cancellation in hearing aid using nonuniform filter bank .....	50
<b>Tsirulnik L.I., Pokladok D.A.</b> Grammatical dictionary and stress placement rules for mobile device TTS-synthesis .....	61

### LOGICAL DESIGN

<b>Bibilo P.N., Soloviev A.L.</b> Functional VHDL-models of FPGA-elements of Spartan 3 family for conversion of projects of digital systems into ASIC .....	69
<b>Novikov D.Ya., Cheremisinova L.D.</b> Implicative method of partially defined Boolean functions analysis .....	79
<b>Zolotorevich L.A.</b> Construction of tests and verification of digital devices flow models on VHDL language .....	87

## MATHEMATICAL MODELING

<b>Eryomina A.R.</b> Invariance of stationary state distribution of heterogeneous queuing networks with multimode strategies and DPS dispatching rule .....	98
<b>Emelichev V.A., Korotkov V.V.</b> Stability analysis of pareto-optimal project portfolio of bi-criteria investment problem with Wald's and Savage's criteria .....	107

## ОБРАБОТКА ИЗОБРАЖЕНИЙ, СИГНАЛОВ И РЕЧИ

УДК 004.93; 519.68

В.А. Ковалев, А.А. Дмитрук, И.В. Сафонов

МЕТОД ПОИСКА СВЯЗЕЙ МЕЖДУ МОРФОЛОГИЧЕСКИМИ СТРУКТУРАМИ  
ГИСТОЛОГИЧЕСКИХ ИЗОБРАЖЕНИЙ И ПОКАЗАТЕЛЯМИ СОСТОЯНИЯ  
ОНКОЛОГИЧЕСКИХ БОЛЬНЫХ

*Предлагается метод поиска и визуализации структур, связанных с показателями состояния онкологических больных. В основе метода лежит многоступенчатая процедура, включающая подсчет признаков изображения, извлечение главных компонент, корреляцию главных компонент с известными свойствами объекта и проекцию найденных зависимостей на исходные гистологические снимки с целью выделения информативных структурных образований. Находятся зависимости между клиническими показателями и морфологическими структурами на соответствующих изображениях.*

**Введение**

Визуальный осмотр гистологических изображений является «золотым стандартом» при постановке окончательного диагноза [1, 2], определении стадии и лечения в случае многих видов рака. При этом в общей дисциплине анализа биомедицинских изображений проблеме анализа гистологических изображений до сих пор уделяется недостаточно внимания [3–5]. Основная причина этого заключается в заметной морфологической сложности, которая присуща микроскопным снимкам тканей живых организмов [3, 4]. В этой связи авторами рассмотрена задача нахождения соответствий между диагностическими показателями онкологических больных и морфологическими структурами на гистологических изображениях ткани яичника.

Выбор типа анализируемых изображений обусловлен тем, что рак яичника – разрушительное заболевание, признанное одной из главных причин женских гинекологических смертей в мире [6]. Начальные признаки злокачественной опухоли в виде небольшого и безболезненного увеличения живота остаются без внимания, и первое обращение женщин в больницы происходит обычно на поздних стадиях заболевания. Поэтому пятилетняя выживаемость в этом случае не превышает 20 % [6, 7].

Наряду с изображениями образцов ткани яичника, окрашенных гематоксилин-эозином (рутинная окраска), были рассмотрены изображения, окрашенные маркером D2-40. Этот маркер выделяет лимфатические сосуды в тканях яичника (если точнее, то он является селективным маркером лимфатического эндотелия в нормальных тканях и патологических сосудах).

Излагаемый метод разработан в рамках большого проекта, направленного на изучение ангиогенеза злокачественных опухолей в ткани яичника [8]. Ангиогенез как развитие новой системы кровеносных сосудов – это важный фактор существенного увеличения опухоли и метастазирования [7]. Без ангиогенеза размер опухоли естественным образом ограничивается 1–2 мм, поскольку для дальнейшего ее роста необходим кислород и питание [9]. Надежды на лечение рака связаны с поиском способов торможения процесса ангиогенеза. Отсюда следует, что обнаружение связей между структурой опухоли, характеристиками ее роста и состоянием пациента является первостепенной задачей для онкологии [6, 7].

В клинической практике предусмотрен сбор информации в базу данных пациентов, содержащую изображения различных модальностей и показатели онкологических больных. К таким показателям относятся, например, социальный статус пациента, результаты обследований и лабораторных тестов до и после операции, история болезни и т. п. Техническая проблема состоит в поиске статистически значимых зависимостей между морфологическими структурами на изображениях, представленными в форме количественных характеристик, и записями в таблице показателей пациентов. Такие корреляции могут быть найдены непосред-

венно с помощью известных методов многомерного статистического анализа. Однако эта техника применима только для априорного анализа, когда существуют предположения о возможных взаимосвязях между структурами на изображении и состоянием пациента и статистический анализ применяется с целью подтвердить или опровергнуть эти гипотезы. Данный подход использовался авторами на этапе предварительного исследования. Осуществлялся поиск соответствий между состоянием пациента и структурами, отмеченными маркером D2-40. Для этого лимфатическая сеть была сегментирована и охарактеризована пятью количественными признаками. Далее был проведен поиск корреляций признаков с данными пациентов. Полученные частные и достаточно скромные выводы не оправдали затраченное время и ресурсы на решение данной задачи [8]. Поэтому рассмотрен альтернативный исследовательский подход, который позволяет, во-первых, обнаружить множество объективно существующих связей и, во вторых, интерпретировать найденные результаты с медицинской точки зрения. Таким образом, работа выполнена в области интеллектуального анализа изображений (image mining). Подобно интеллектуальному анализу данных (data mining) анализ изображений может быть интерпретирован в этом смысле как процесс обнаружения скрытых и неочевидных структур [10, 11] и извлечения из изображений связей и данных, не хранящихся в базе в явном виде [12–14]. В силу морфологической и структурной сложности гистологических изображений процесс их анализа сложно автоматизировать. Поэтому всесторонний интеллектуальный анализ данных является многообещающим для решения поставленных задач.

Для получения ожидаемых результатов разрабатываемый метод поиска связей между морфологическими структурами гистологических изображений и показателями состояния онкологических больных должен удовлетворять следующим требованиям:

1) дескрипторы изображений должны быть в достаточной мере информативными для адекватного представления широкого диапазона морфологических признаков изображений, как цветных, так и представленных оттенками серого;

2) количественные признаки, вычисленные с помощью дескрипторов и коррелированные с данными пациентов, должны допускать несложное отображение отобранных корреляций на оригинальные изображения для визуализации ключевых морфологических структур;

3) число признаков для описания одного изображения не должно быть намного больше общего количества пациентов. Это требование позволит избежать появления случайных корреляций, которые неизбежны для очень больших наборов данных.

Целью работы является разработка метода поиска связей, удовлетворяющего указанным выше условиям. Использование метода демонстрируется на анализе базы данных пациенток, страдающих раком яичника.

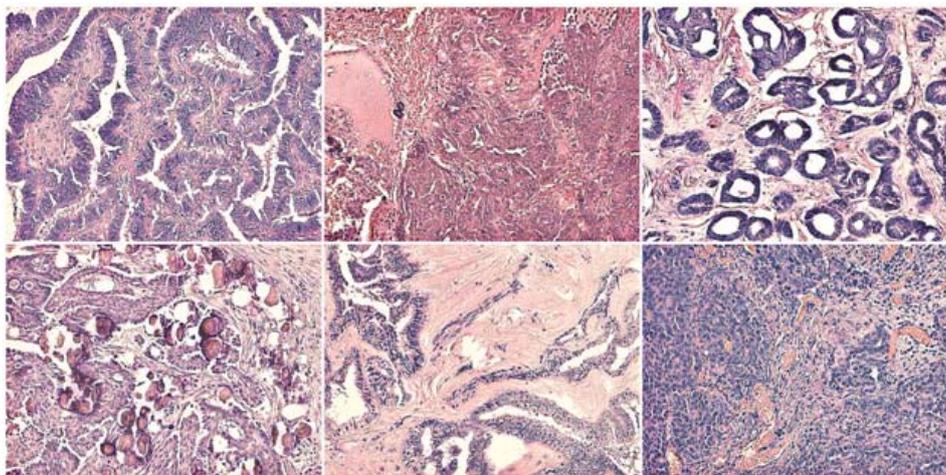
## 1. Материалы и методы

База данных состояла из таблицы с показателями состояния больных и гистологических изображений опухолевой ткани яичника по 68 пациентам онкологического диспансера (женщины, средний возраст 59,8 лет, среднее квадратическое отклонение возраста 11,2). Набор гистологических изображений представлял собой 952 полноцветных снимка размером  $2048 \times 1536$  пикселей, полученных с помощью микроскопа Leica DMD108 при 200-кратном увеличении. В состав набора включены 272 изображения с рутинной прокраской (гематоксилином и эозином), по четыре изображения на каждого пациента, и 680 изображений ткани яичника, обработанной маркером D2-40, по 10 изображений на каждого пациента. Примеры всех типов изображений показаны на рис. 1.

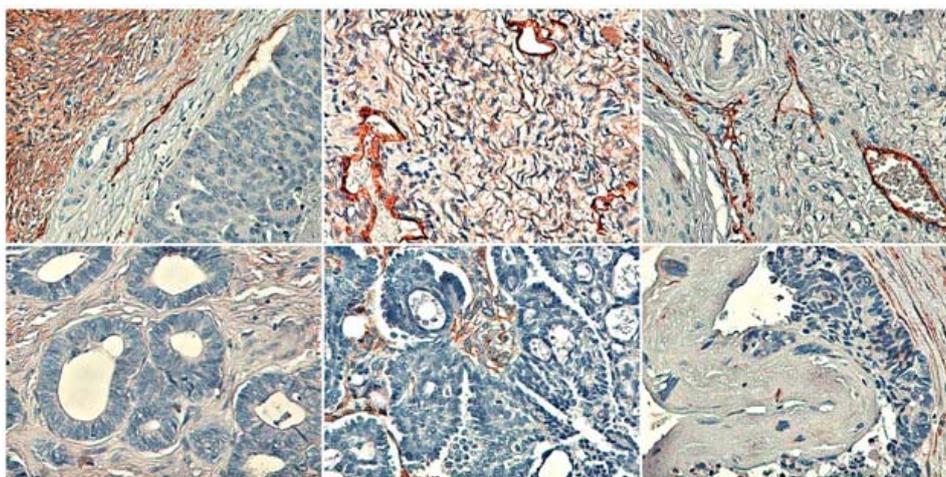
Таблица с показателями больных содержала около 80 различных признаков (TNM-классификацию опухоли, количество курсов химиотерапии, дифференциацию опухоли, детали хирургических операций и т. п.).

Из-за упоминаемых выше особенностей гистологических изображений для их анализа наиболее часто используются дескрипторы не отдельных областей или сегментов, а изображения в целом. Хорошо зарекомендовали себя цветные матрицы совместной встречаемости, предложенные независимо в работах [15, 16]. Однако принимая во внимание первое требование из введения настоящей статьи, подход с совместной встречаемостью пикселей был модернизи-

рован до четырехмерного случая, а именно: для определения соседства использовались тройки пикселей, лежащих в вершинах равносторонних треугольников с заданной длиной стороны. Такое расширение базового подхода технически не сводится только к добавлению дополнительного измерения к матрице совместной встречаемости. Отличия гораздо глубже и связаны с проблемой описания разнообразных типов текстур с помощью статистик первого порядка (яркостей пикселей или значений цветов), второго (градиентов) и более высоких порядков. Эта проблема была тщательно исследована Б. Джулисом [17]. Ученым было экспериментально доказано [18], что паттерны активности мозга значительно различаются при наблюдении текстур с низкими и высокими пространственными корреляциями. При этом важно различать статистики высокого порядка в пространственном и в яркостном [19] смыслах.



а)



б)

Рис. 1. Примеры изображений: а) рутинная прокраска; б) прокраска D2-40

Формализуем используемые дескрипторы следующим образом. Пусть  $F_G = \{I(x, y)\} = \{I(i)\} = \{I(j)\} = \{I(k)\}$  – изображение в оттенках серого размером  $M \times N$  пикселей. Положим также, что все пиксели изображения пронумерованы с помощью индексов  $i, j$  и  $k$ , где  $i = \overline{1, MN}$ ,  $j = \overline{1, MN}$ ,  $k = \overline{1, MN}$  и их уровни интенсивности равны  $I(i)$ ,  $I(j)$ ,  $I(k)$  соответственно. Индексы естественным образом определяются координатами соответствующих пикселей:  $i = (x_i, y_i)$ ,  $j = (x_j, y_j)$ ,  $k = (x_k, y_k)$ . Тогда для серого изображения четырехмерная матрица совместной встречаемости типа  $III D$ , которая задается тройками пикселей  $(i, j, k)$ , распо-

ложенных в вершинах равносторонних треугольников с длинами сторон из множества  $D$ , может быть определена следующим образом:

$$W_{\text{ИИД}} = \|I(i), I(j), I(k), d\|; \quad (1)$$

$$d(i, j) = d(i, k) = d(j, k), \quad d \in D; \quad (2)$$

$$i < j, \quad i < k, \quad \forall i: y_j \geq y_i, y_k < y_i. \quad (3)$$

Неравенства (3) заключают в себе требование перебора всех возможных треугольников без повторений. Система уравнений (1)–(3) описывает алгоритм покрытия всего изображения равносторонними треугольниками. Как следует из этой системы, процедура покрытия состоит из последовательных смещений в каждую точку  $i$  изображения и построения вокруг этой точки равносторонних треугольников с заданной длиной стороны. При этом две другие вершины треугольника не должны оказаться выше базовой вершины  $i$ .

Когда рассматриваются не яркости пикселей, а их цвета, исходное цветовое пространство редуцируется и соответствующая цветная матрица совместной встречаемости типа *СССД* определяется по той же схеме с использованием индексов цветов вместо уровней яркости. После вычисления матрицы встречаемости обычно считаются признаки Харалика, и далее они используются на этапах описания, кластеризации и т. д. Однако данная процедура в рассматриваемом случае неприменима, поскольку признаки Харалика не могут быть отображены обратно на исходные изображения, что является необходимым условием и декларировано во втором требовании к используемым признакам в этой статье. Сами элементы матрицы совместной встречаемости могут быть отображены на исходное изображение [20], но их слишком много (более 7000), чтобы удовлетворить третьему требованию, предъявляемому к дескрипторам. Решение этого противоречия приводит к использованию метода главных компонент (Principal component analysis – PCA). С помощью PCA из элементов матрицы совместной встречаемости извлекается небольшое число некоррелированных между собой признаков.

Разработанный метод заключается в подсчете четырехмерных матриц совместной встречаемости, извлечении главных компонент, корреляции полученных компонент с данными пациента, выборе среди этих данных высококоррелированных значений и обратном их проецировании на матрицу совместной встречаемости. Последним шагом метода является использование выделенных таким образом элементов матрицы совместной встречаемости для визуализации соответствующих структур на гистологических изображениях.

Поскольку главные компоненты некоррелированы по определению, поиск значимых связей выполнялся с помощью простых однофакторных статистических методов, таких как корреляционный анализ главных компонент и показателей пациентов, а также  $t$ -тест Стьюдента. Последний применялся в случаях, когда значения показателей состояния пациента, например таких как «наличие удаленных метастаз» (присутствуют/отсутствуют), «степень дифференциации раковой опухоли» (высокая/низкая) и т. д., были представлены в номинальной шкале и разбивали естественным образом пациентов на группы. В указанных случаях с помощью  $t$ -теста проводилось сравнение значимости отличия средних значений полученных главных компонент по выделенным подгруппам пациентов.

## 2. Результаты исследования

Оригинальные изображения в формате RGB были конвертированы в цветовое пространство Lab, количество исходных цветов было уменьшено до 24 наиболее представительных с помощью алгоритма median cut [21]. Этот алгоритм используется в области обработки изображений для эффективного редуцирования цветового пространства. Вначале трехмерные расширенные цветовые матрицы типа *ССС* с фиксированным межпиксельным расстоянием  $d$  содержали  $24^3 = 13\,824$  элемента. С учетом дальнейшего приведения матрицы к нижнему треугольному виду количество используемых элементов матрицы для одного межпиксельного расстояния сократилось до  $N = 2600$ . Поскольку одновременно использовались три типа равно-

сторонних треугольников с длинами сторон  $D = \{1, 2, 3\}$ , окончательное количество анализируемых элементов для каждого изображения равнялось  $2600 \times 3 = 7800$ . Далее была составлена сборная таблица признаков, состоящая из 7800 столбцов (по количеству признаков на каждом изображении) и 68 строк (по количеству пациентов). Каждая строка этой таблицы представляла собой векторизованную матрицу совместной встречаемости изображения. После применения метода главных компонент для дальнейшего анализа были выбраны 27 компонент для рутинных изображений и 38 – для изображений с маркером D2-40, покрывающих 95 % вариативности исходных признаков. Эти результаты позволяют предположить, что структурная вариативность изображений с маркером D2-40 существенно выше по сравнению с рутинными.

После корреляции с клиническими данными 27 главных компонент признаков рутинных изображений выявлено 43 корреляции с уровнем значимости  $p < 0,01$ . Та же процедура, примененная к 38 главным компонентам признаков другого типа изображений, выявила 47 значимых связей и записей в таблице состояния больных. При детальном изучении значимых корреляций выяснилось, что не во всех случаях найденные связи между определенными гистологическими структурами и клиническими показателями могут быть легко интерпретированы, несмотря на предполагаемый интерес с научной и практической точек зрения. Например, при анализе корреляций между главными компонентами признаков рутинных изображений и данными состояния больных выглядят многообещающими следующие показатели: развитие удаленных метастаз ( $p < 0,001$ ), степень дифференциации раковой опухоли ( $p < 0,007$ ), количество выкидышей ( $p < 0,0001$ ) и количество проведенных курсов химиотерапии ( $p < 0,000002$ ,  $r = -0,543$ ). Визуализация некоторых структур приведена в верхней строчке рис. 2.

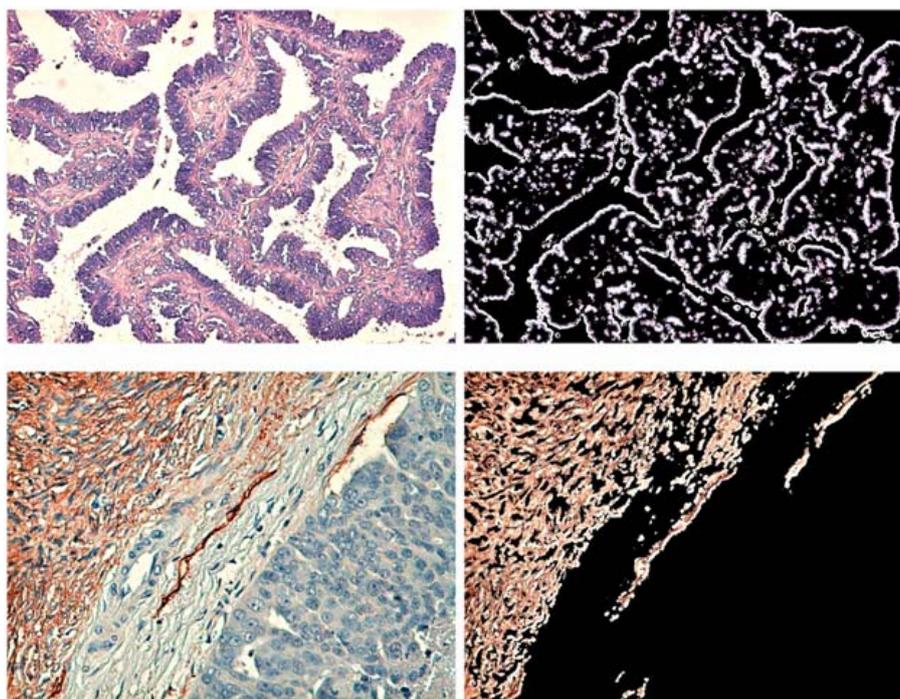


Рис. 2. Исходные изображения (левая колонка) и найденные ключевые структуры (правая колонка) для рутинных (верхняя строка) и D2-40 (нижняя строка) изображений

Обратная зависимость длины границы с количеством обследований может быть объяснена фактом, что у более «молодых» опухолей, которые поддаются химическому лечению, структура ткани более развита по сравнению со «старыми» опухолями, которые сразу удаляются. Изображения найденных связей для тканей, обработанных эндотелиальным маркером D2-40, демонстрируют похожие признаки. На нижней строчке рис. 2 показана одна из таких зависимостей. Доля прокрашенных маркером D2-40 структур сильно коррелирует с уровнем дифференциации опухоли ( $p < 0,009$ ), временем выживаемости пациента ( $p < 0,010$ ) и наличием рецидива ( $p < 0,017$ ).

Также вычислялись совместные матрицы встречаемости типа *IID* на конвертированных в оттенки серого изображениях. Несмотря на несколько найденных многообещающих корреляций для этого типа дескрипторов, в целом полученные данные противоречивы. В частности, при обратном отображении элементов матрицы *IID* на исходное серое изображение одновременно выделялись структуры с различным биологическим смыслом. Причина этого видится в неизбежном при таком подходе редуцировании диапазона различных цветов в одно значение уровня яркости серого цвета.

### Заключение

Представленный метод может рассматриваться как многообещающий инструмент для автоматической идентификации и визуализации структур на гистологических изображениях, релевантных данным о состоянии пациентов.

Поскольку в методе не предусмотрено внутренних средств для интерпретации найденных связей, необходим экспертный анализ результатов.

Дальнейшее усовершенствование метода предполагает развитие автоматической процедуры отбора элементов матрицы для отображения на исходные изображения после расчета главных компонент.

Работа выполнена в рамках проекта МНТЦ В-1682.

Авторы выражают благодарность сотрудникам Минского городского клинического онкологического диспансера М.В. Фридману и С.Е. Шелкович за предоставленные данные и консультации по медицинским вопросам.

### Список литературы

1. Schwab, M. Encyclopedia of Cancer / M. Schwab. – N.Y. : Academic Press, 2009. – 3235 p.
2. Hayat, M. Methods of Cancer Diagnosis, Therapy and Prognosis. In 6 vol. / M. Hayat. – Springer, 2009–2010.
3. Wootton, R. Image Analysis in Histology: Conventional and Confocal Microscopy / R. Wootton, D. Springall, J. Polak. – Cambridge : Cambridge University Press, 1995. – 425 p.
4. Histopathological image analysis : A review / M.N. Gurcan [et al.] // IEEE Reviews in Biomedical Engineering. – 2009. – Vol. 2. – P.147–171.
5. Histopathological image analysis using model-based intermediate representations and color texture: Follicular lymphoma grading / O. Sertel [et al.] // Journal of Signal Processing Systems. – 2009. – Vol. 55, № 1. – P.169–183.
6. Stack, M.S. Ovarian Cancer (Cancer Treatment and Research) / M.S. Stack, D.A. Fishman. – N.Y. : Springer, 2009. – 409 p.
7. Bamberger, E. Angiogenesis in epithelial ovarian cancer (review) / E. Bamberger, C. Perrett // Molecular Pathology. – 2002. – № 55. – P. 348–359.
8. Computer-aided image processing of angiogenic histological samples in ovarian cancer / M. Sprindzuk [et al.] // Journal of Clinical Medicine Research. – 2009. – Vol. 1, № 5. – P. 249–261.
9. Folkman, J. What is the evidence that tumors are angiogenesis dependent? / J. Folkman // Journal of the National Cancer Institute. – 1990. – Vol. 82, № 1. – P. 4–6.
10. Hsu, W. Image mining: Trends and developments / W. Hsu, M. Lee, J. Zhang // Journal of Intelligent Information Systems. – 2002. – Vol. 19, № 1. – P. 7–23.
11. Herold, J. Multivariate image mining / J. Herold, C. Loyek, T.W. Nattkemper // Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. – 2011. – Vol. 1, № 1. – P. 2–13.
12. Perner, P. Image mining: Issues, framework, a generic tool and its application to medical image diagnosis / P. Perner // Engineering Applications of Artificial Intelligence. – 2002. – Vol. 15, № 2. – P. 205–216.
13. Kovalev, V. Mining lung shape from x-ray images / V. Kovalev, A. Prus, P. Vankevich // Machine Learning and Data Mining in Pattern Recognition (MLDM-2009). – Germany, 2009. – Vol. 5632. – P. 554–568.

14. Kovalev, V. Histological image mining for exploring textural differences in cancerous tissue / V. Kovalev, I. Safonau, A. Prus // Swedish Symposium on Image Analysis (SSBA-2010). – Sweden, 2010. – P. 113–116.
15. Image indexing using color correlograms / J. Huang [et al.] // IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition. – USA, 1997. – P. 762–768.
16. Kovalev, V. Color co-occurrence descriptors for querying-by-example / V. Kovalev, S. Volmer // Int. Conf. on Multimedia Modelling. – Switzerland, 1998. – P. 32–38.
17. Julesz, B. Foundations of Cyclopean Perception / B. Julesz. – Cambridge, Massachusetts : The MIT Press, 2006. – 426 p.
18. Cortical regions involved in visual texture perception: a fMRI study / L.L. Beason-Held [et al.] // Cognitive Brain Research. – 1998. – № 7. – P. 111–118.
19. Petrou, M. Three-dimensional nonlinear invisible boundary detection / M. Petrou, V. Kovalev, J. Reichenbach // IEEE Trans. Image Processing. – 2006. – Vol. 15, № 10. – P. 3020–3032.
20. Kovalev, V. Detection of structural differences between the brains of schizophrenic patients and controls / V. Kovalev, M. Petrou, J. Suckling // Psychiatry Research: Neuroimaging. – 2003. – № 124. – P. 177–189.
21. Heckbert, P. Color image quantization for frame buffer display / P. Heckbert // Proc. of the 9th annual conf. on computer graphics and interactive techniques (SIGGRAPH '82). – USA, 1982. – P. 297–307.

Поступила 10.02.12

*Объединенный институт проблем  
информатики НАН Беларуси,  
Минск, Сурганова, 6  
e-mail: vassili.kovalev@gmail.com,  
dmitruk@newman.bas-net.by*

**V.A. Kovalev, A.A. Dmitruk, I.U. Safonau**

**A METHOD FOR IDENTIFICATION AND VISUALIZATION  
OF HISTOLOGICAL IMAGE STRUCTURES RELEVANT  
TO THE CANCER PATIENT CONDITIONS**

A method is suggested for identification and visualization of histology image structures relevant to the key characteristics of the state of cancer patients. The method is based on a multistep procedure which includes calculating image descriptors, extracting their principal components, correlating them to known object properties and mapping disclosed regularities all the way back up to the corresponding image structures they found to be linked with. As a result, a number of associations between the patients' conditions and morphological image structures were found.

УДК 004.93'1; 004.932

Д.В. Прадун, А.А. Мамчич

## АЛГОРИТМ АВТОМАТИЧЕСКОГО ОПРЕДЕЛЕНИЯ ТЕНЕЙ С ИСПОЛЬЗОВАНИЕМ ЭНТРОПИЙНОГО МЕТОДА БИНАРИЗАЦИИ ИЗОБРАЖЕНИЙ

*Предлагается алгоритм автоматического определения теней, который основан на представлении цифрового изображения в цветовом  $YUV$ -пространстве с последующим анализом гистограмм и использованием энтропийного метода определения порога бинаризации для выделения теневых пикселей. Разработанный алгоритм предусматривает параллельную модификацию с применением многопроцессорных вычислительных систем без усложнения программной реализации.*

### Введение

Процесс выявления и удаления теней на цветных изображениях представляет собой актуальную задачу при анализе космической информации. Так, наличие затемненных участков на аэро- или космических снимках может создавать ряд нежелательных особенностей (артефактов), которые значительно усложняют процесс распознавания и сегментации расположенных на них объектов. Поэтому в настоящее время разработка эффективных методов определения и удаления теней является чрезвычайно важной проблемой в области обработки цифровых изображений.

Существующие алгоритмы поиска и удаления теней основаны, главным образом, на анализе таких характеристик цифровых изображений, как цвет, яркость и текстура [1]. В работах [2, 3] предлагается подход, который заключается в получении специального изображения, инвариантного к освещению. С учетом того что на таком снимке отсутствуют тени, процесс их нахождения сводится к сравнению векторных полей градиентов исходного и инвариантного изображений. Как отмечается в [4], результат работы данного метода получается заметно размытым, что допускает его использование только в качестве предварительной обработки для большинства задач компьютерного зрения. При определении теней в алгоритмах плотного оптического потока возможно выделение цветовых составляющих каналов в  $RGB$ -пространстве с последующим приравниванием яркостей пикселей к среднему константному значению [5]. Существенным недостатком такого подхода является низкое качество его работы на широком диапазоне изображений с учетом того, что пороговые величины результирующей константной яркости различаются для множества снимков. С целью определения теней также применяются алгоритмы, основанные на известном методе кластеризации  $K$ -средних ( $K$ -means) [6]. Несмотря на приемлемую точность поиска, эти алгоритмы имеют высокую чувствительность к выбору начальных точек кластеров и медленно работают с большими объемами данных.

В настоящей статье предлагается алгоритм поиска теней, который основан на представлении цифрового изображения в цветовом  $YUV$ -пространстве с последующим анализом гистограмм и использованием энтропийного метода определения порога бинаризации для выделения теневых пикселей. Результат работы представляется в виде бинарного изображения, содержащего маску теней. Достоинствами разработанного алгоритма является высокая точность поиска, а также вычислительная эффективность с возможностью параллельной реализации.

### 1. Общая постановка задачи и схема алгоритма

Анализируя изображения с имеющимися тенями (рис. 1, а, в), можно сделать вывод, что основной характеристикой пикселей, принадлежащих теневым объектам, является низкий уровень их яркости по сравнению с другими объектами изображений. Поэтому для эффективного выделения таких пикселей фактически необходимо провести бинаризацию с выделением объектов, яркость которых ниже заданного порога. Кроме того, специфика изображений, получаемых в результате аэро- и космической съемки, заключается в наличии шума и помех при съемке, что, в

свою очередь, требует выполнения определенных действий над снимками с целью удаления подобных искажающих факторов. Для этого исходное изображение предварительно фильтруется или сглаживается, в результате чего получается улучшенный снимок с ярко выраженными однородными областями.

Общую схему алгоритма выявления теней можно представить следующим образом: пусть на цветном  $RGB$ -изображении размером  $M \times N$  имеются тени объектов, полученные в результате аэро- или космической съемки. Кроме того, исходное изображение предварительно отфильтровано с помощью алгоритма максимального потока сети с целью выделения однородных областей (рис. 1, б, з) [7, 8]. Для выполнения более эффективного и быстрого анализа изображение переводится в цветовую  $YUV$ -модель, которая состоит из следующих компонент:  $Y$  – яркостная составляющая пиксела;  $U$  и  $V$  – цветоразностные составляющие пиксела, необходимые для восстановления требуемого цвета при обратной конвертации.

Цветовая  $YUV$ -модель применяется в основном в телевидении, а также при хранении и обработке видеоданных. Основной компонентой данной модели является яркостная составляющая  $Y$ , которая представляет собой полутоновое изображение исходного снимка. Обработка данной компоненты выполняется в предложенном алгоритме выявления теней.

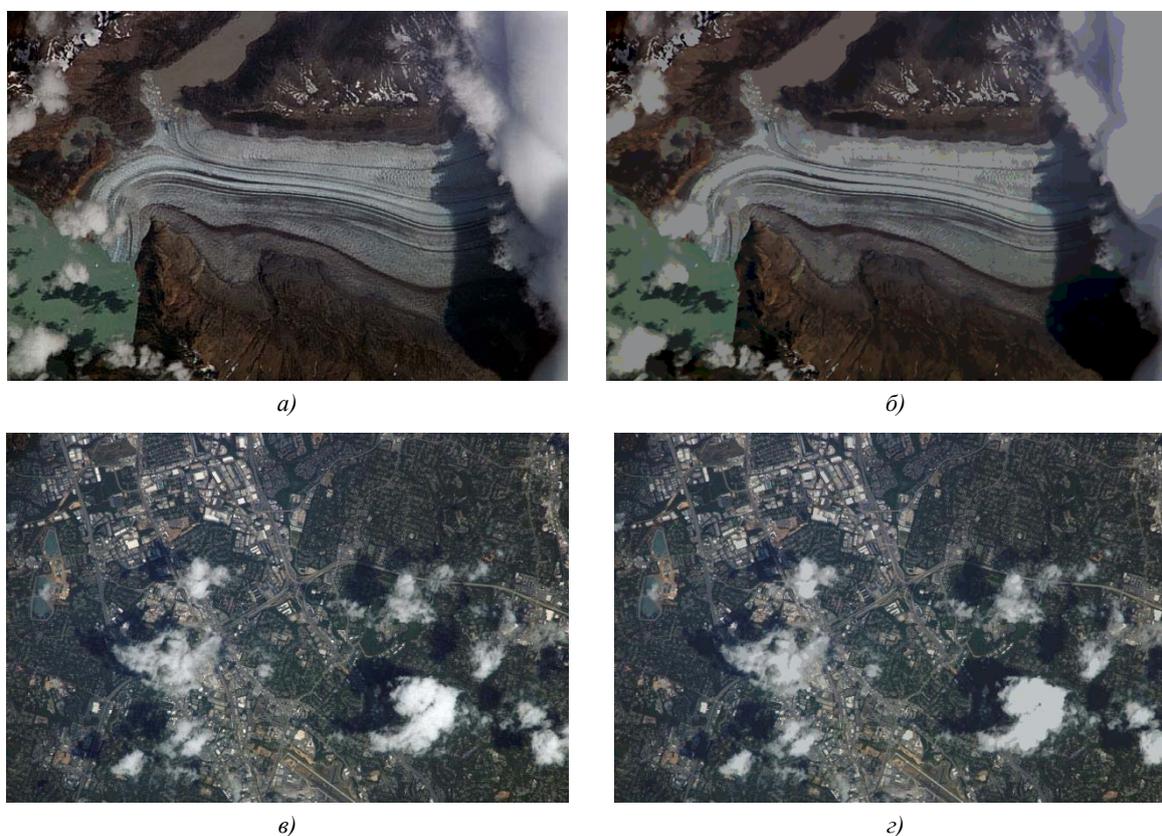


Рис. 1. Изображения с наличием теней: а), в) исходные снимки; б), з) фильтрация алгоритмом максимального потока

Для преобразования  $RGB$ -изображения в  $YUV$ -изображение используются следующие формулы:

$$\begin{aligned} Y &= 0,299 \cdot R + 0,587 \cdot G + 0,114 \cdot B; \\ U &= -0,14713 \cdot R - 0,2886 \cdot G + 0,436 \cdot B; \\ V &= 0,615 \cdot R - 0,51499 \cdot G - 0,10001 \cdot B. \end{aligned} \quad (1)$$

Отметим, что в большинстве случаев для определенного  $Y$ -слоя изображения диапазон значений пикселей данной составляющей меньше чем  $[0; 255]$ , что чаще всего усложняет дальнейший

процесс анализа яркостной составляющей снимка. Поэтому следующим шагом алгоритма является так называемое расширение гистограммы  $Y$ -компоненты до диапазона  $[0; 255]$  по формуле

$$\tilde{x}_{i,j} = 255 \cdot \frac{x_{i,j} - x_{\min}}{x_{\max} - x_{\min}}, \quad (2)$$

где  $x_{i,j}$  и  $\tilde{x}_{i,j}$  – старое и новое значения яркости пиксела  $Y$ -компоненты в точке  $(i, j)$ ,  $i=1, \dots, M$ ,  $j=1, \dots, N$ , соответственно;  $x_{\max}$  и  $x_{\min}$  – максимальное и минимальное значения яркости  $Y$ -компоненты изображения. Гистограммы изображений на рис. 1, полученные в результате подобных преобразований, показаны на рис. 2.

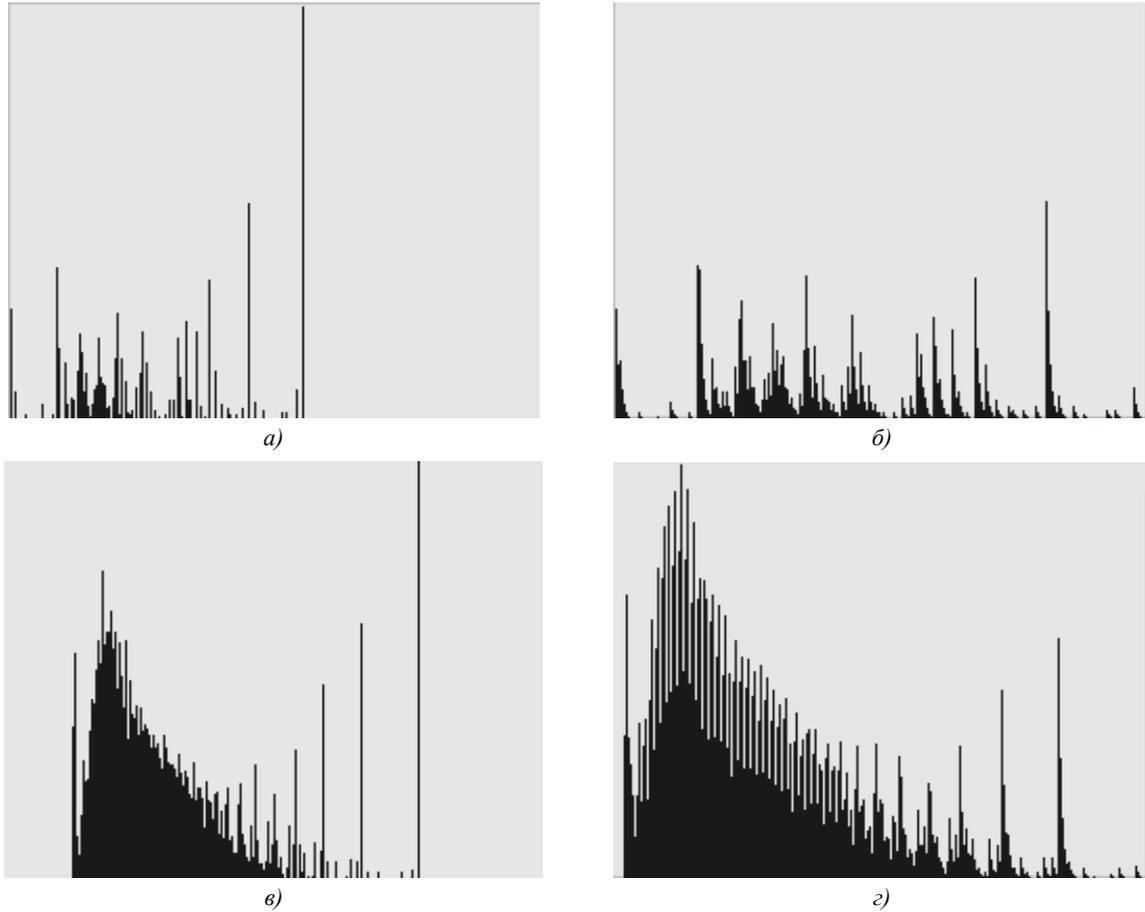


Рис. 2. Гистограммы изображений на рис. 1: а), в) исходные; б), г) после видоизменения

После того как значения яркостной составляющей  $YUV$ -модели определены в диапазоне  $[0; 255]$ , строится маска теней исходя из значений  $Y$ -компоненты. Для этого сначала вычисляется порог бинаризации изображения  $T_{bin}$  с помощью энтропийного метода Капура [9, 10]. Данный метод бинаризации основывается на вычислении значений энтропий переднего плана и заднего фона изображения по формулам

$$T_{bin} = \arg \max[H_b(T) + H_f(T)];$$

$$H_b(T) = - \sum_{g=0}^T \frac{p(g)}{P_b(T)} \cdot \log \frac{p(g)}{P_b(T)}; \quad H_f(T) = - \sum_{g=T+1}^G \frac{p(g)}{P_f(T)} \cdot \log \frac{p(g)}{P_f(T)}, \quad (3)$$

где  $G$  – максимально возможное значение интенсивности пикселей;  $T$  – порог бинаризации, выбираемый в заданном диапазоне  $[0; G]$  (в нашем случае  $G = 255$ );  $P_b(T)$  и  $P_f(T)$  – функции

распределения вероятности для переднего плана и заднего фона соответственно, вычисляемые по формуле

$$P_b(T) = \sum_{i=0}^T p(i), \quad P_f(T) = \sum_{i=T+1}^G p(i). \quad (4)$$

Здесь  $p(i)$  – оценка вероятности, вычисляемая как  $p(g) = \frac{K_g}{K}$ . При этом  $K_g$  – число пикселей, интенсивность которых равна  $g$ ;  $K = M \times N$  – общее число пикселей.

Результаты бинаризации с помощью метода Капура используются для определения пикселей, принадлежащих теням, по следующей схеме:

$$mask_{i,j} = \begin{cases} 255, & \text{если } \tilde{x}_{i,j} < \frac{T_{bin}}{255} \cdot avg; \\ 255, & \text{если } \tilde{x}_{i,j} < \frac{T_{bin}}{255} \cdot avg_{window}; \\ 0 & \text{в противном случае,} \end{cases} \quad (5)$$

где  $mask_{i,j}$  – значение маски теней в точке  $(i, j)$ ,  $i = 1, \dots, M$ ,  $j = 1, \dots, N$ ;  $avg$  – среднее значение яркости  $Y$ -слоя;  $avg_{window}$  – среднее значение яркости в заданном окне  $w \times w$ .

Следует отметить, что значение  $avg_{window}$  вычисляется только для тех пикселей, которые не были определены как принадлежащие теням изображения при сравнении их яркостной составляющей с глобальным средним значением  $avg$ . Поэтому размеры окна  $w \times w$  итерационно уменьшаются от большего к меньшему для более точного анализа окрестности пикселей. В предложенном алгоритме значение размера окна  $w$  изменяется от 21 до 3 с шагом 3. По сравнению с предложенными размерами окна в [1] были выбраны сравнительно небольшие значения  $w$ , так как в ином случае при анализе некоторых изображений возникает опасность появления на маске теней большого количества маленьких кластеров размером 3-5 пикселей, а также слияния теневых пикселей с нетеневыми, что в дальнейшем будет мешать обработке графических данных (рис. 3). Кроме того, результаты исследований показали, что применение метода Капура для определения порога бинаризации предпочтительнее других известных алгоритмов. Так, при использовании кластерных методов бинаризации Отсу и Киттлера, которые известны как наиболее эффективные и известные методы приведения изображений к двоичному виду [10], заметно определение слишком низкого порога, что приводит к потере теневых объектов и крайне негативно влияет на дальнейший процесс обработки (рис. 4).

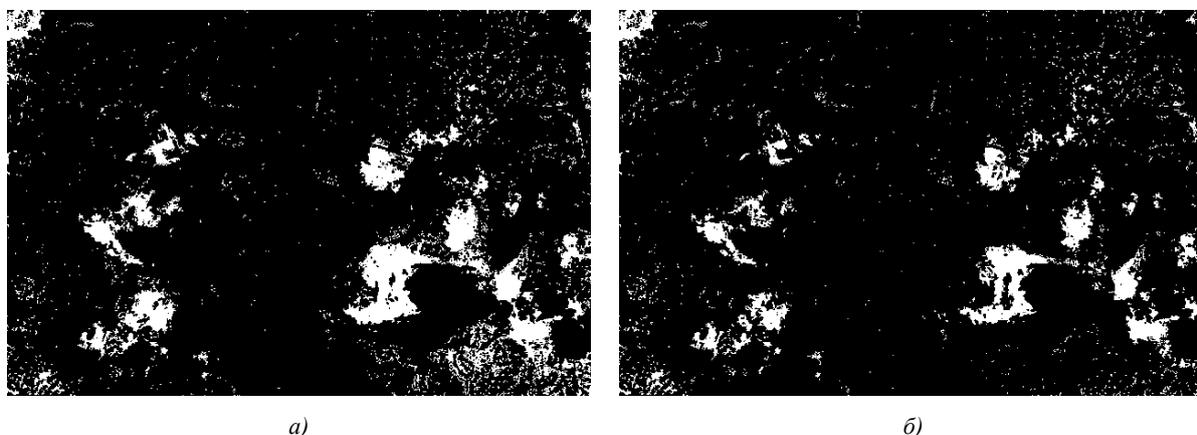


Рис. 3. Маски теней для изображения на рис. 1, в: а) использование больших размеров окна; б) использование предложенных значений  $w$

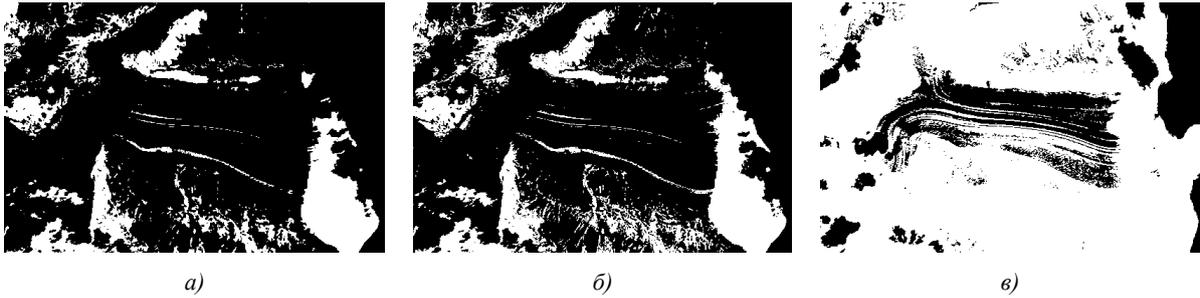


Рис. 4. Определение теней разными методами бинаризации изображения на рис. 1, *а)* методом Капура; *б)* методом Отсу; *в)* методом Киттлера

## 2. Результаты работы алгоритма

Исходя из описанной ранее схемы алгоритма можно заметить, что при итерационном сравнении значения  $Y$ -составляющей со значениями  $avg$  и  $avg_{window}$  и уменьшении значений размеров окна можно определить пиксели, которые принадлежат теневым объектам изображения (рис. 5). Такие пиксели могут быть исключены из дальнейшего процесса обработки графической информации для обеспечения более эффективного анализа исходных данных. Кроме того, достоинство алгоритма состоит в том, что его параллельная модификация с использованием многопроцессорных вычислительных систем не потребует дополнительных преобразований самого алгоритма с точки зрения сложности его реализации.

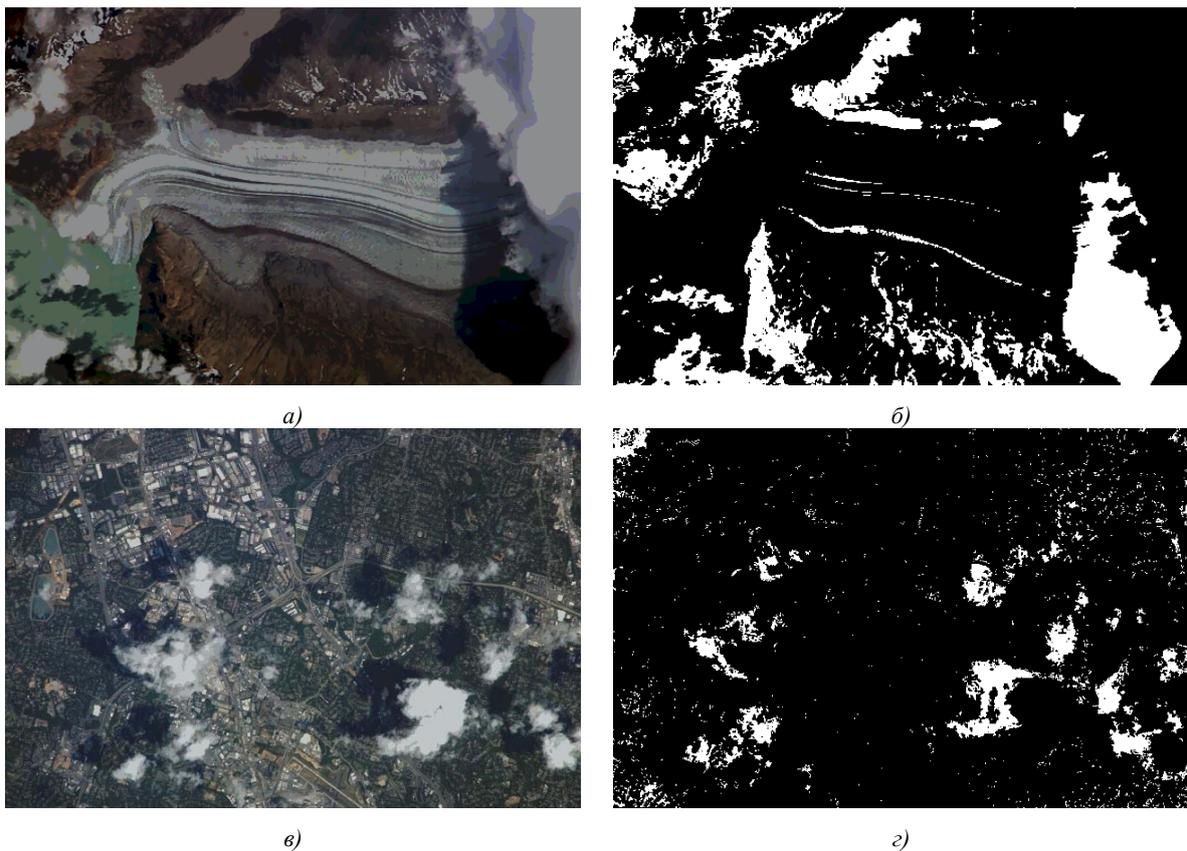


Рис. 5. Результаты поиска теней на изображениях:  
*а), в)* исходные сглаженные снимки; *б), з)* построенные маски теней

Несмотря на хорошие результаты поиска теней, можно увидеть, что наряду с теньевыми пикселями в маске теней содержатся пиксели, которые на исходном изображении сложно оп-

ределить как принадлежащие теням. Это связано с тем, что часто на изображениях располагаются слабоосвещенные объекты или объекты темного цвета, которые по своим яркостным характеристикам схожи с теневыми пикселями. Поэтому необходим последующий анализ полученных масок теней с целью исключения пикселей, которые реально не относятся к теням. В работе [1] как вариант предлагается использовать анализ контуров теневых объектов исходя из того, что тени обладают более гладкими гранями, нежели все остальные объекты изображений. С другой стороны, можно применять определенные методы распознавания и кластеризации выделенных в маске теней пикселей для анализа того, к каким видам объектов и к кластерам с какими характеристиками они относятся.

Заметим, что данный алгоритм эффективен при его использовании именно для изображений, где наличие теней очевидно. В ситуациях, когда обрабатываются снимки без наличия теневых пикселей, результаты построения маски теней являются неудовлетворительными (рис. 6). Поэтому предварительно анализируемый снимок необходимо оценить с точки зрения наличия теней, что может быть выполнено самим оператором системы распознавания, в которой предложенный алгоритм будет использоваться.

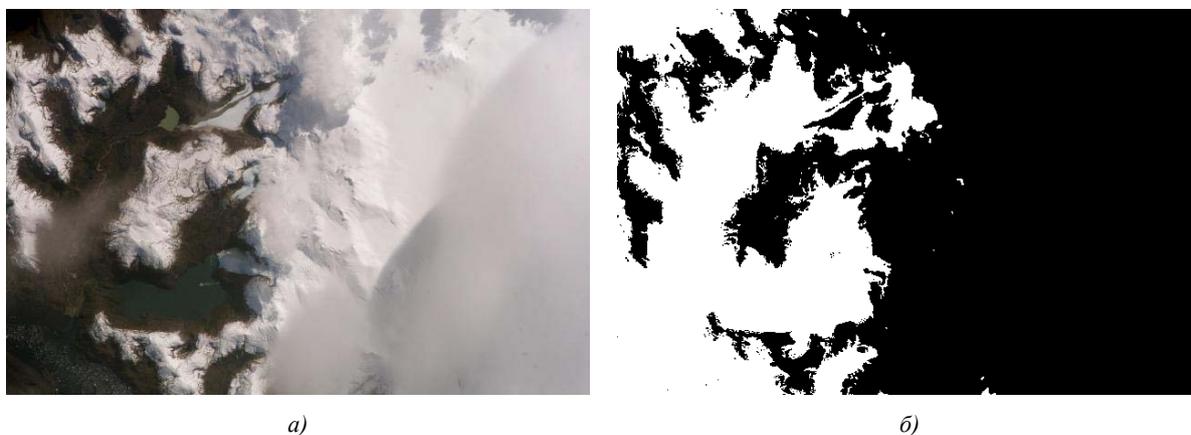


Рис. 6. Поиск теней на изображениях без явного наличия теневых пикселей:  
а) исходный снимок; б) результат ошибочного нахождения теней

### Заключение

Предложенный алгоритм поиска теней на космических и аэроснимках позволяет выявлять теневые пиксели на изображениях с помощью построения соответствующих масок. Используя энтропийный метод бинаризации и итерационный анализ заданной окрестности пикселей, алгоритм выполняет адаптивное определение принадлежности пикселей теневым объектам на изображении. Кроме того, он предусматривает параллельную модификацию с использованием многопроцессорных вычислительных систем без усложнения программной реализации. На данном этапе разработанная технология выявления теней обладает рядом недостатков, заключающихся в основном в отнесении к теневым объектам пикселей, изначально не принадлежащих к таковым. Поэтому дальнейшая модификация алгоритма будет направлена на решение данной проблемы с помощью кластеризационных и классификационных методов обработки изображений, которые позволят определить, какие из выбранных в маске теней пикселей действительно являются теневыми.

### Список литературы

1. Shadow detection and removal from a single image / C. Blajovici [et al.] // 19<sup>th</sup> Summer School on Image Processing, Szeged, Hungary, 7–16 July, 2011 / University of Szeged. – Szeged, Hungary, 2011. – P. 1–6.
2. Finlayson, G. Removing Shadows from Images / G. Finlayson, S. Hordley, M. Drew // European Conference on Computer Vision. – 2002. – Vol. 2353. – P. 823–836.

3. Entropy Minimization for Shadow Removal / G. Finlayson [et al.] // International Journal of Computer Vision. – 2009. – Vol. 85 (1). – P. 35–57.
4. Кононов, В. Применение уравнения Пуассона в задачах обработки изображений / В. Кононов, В. Конушин // Сетевой журнал «Компьютерная графика и мультимедиа» [Электронный ресурс]. – 2009. – Режим доступа : [http://www.cgm.computergraphics.ru/files/cgm/pdf/cgm\\_17\\_2202.pdf](http://www.cgm.computergraphics.ru/files/cgm/pdf/cgm_17_2202.pdf). – Дата доступа : 15.02.2012.
5. Харченко, К.В. Удаление теней в изображениях для обработки плотного оптического потока / К.В. Харченко // Материалы Междунар. науч.-техн. конф. САИТ-2011, Киев, 23–28 мая 2011 г. – Киев : УНК «ИПСА» НТУУ «КПИ», 2011. – С. 394.
6. Xu, L. Shadow Removal from a Single Image / L. Xu, F. Qi, R. Jiang // Proc. of the Sixth Intern. Conf. on Intelligent Systems Design and Applications (ISDA 2006). – Jinan, China, 2006. – Vol. 2. – P. 1049–1054.
7. Прадун, Д.В. Фильтрация и кластеризация мультиспектральных изображений с помощью алгоритма максимального потока в сети на основе вычисления градиента / Д.В. Прадун, Б.А. Залесский // Информатика. – 2010. – № 3(27). – С. 73–79.
8. Pradun, D.V. Block-Parallel Clustering of Multispectral Images Using Max Flow Network Algorithm / D.V. Pradun // Proc. of the 11<sup>th</sup> Intern. Conf. PRIP'2011, Minsk, May 18–20, 2011 / Belarusian State University of Informatics and Radioelectronics. – Minsk, 2011. – P. 76–79.
9. Kapur, J.N. A new method for gray-level picture thresholding using entropy of the histogram / J.N. Kapur, P.K. Sahoo, A.K.C. Wong // Graph. Models Image Process. – 1985. – Vol. 29. – P. 273–285.
10. Sezgin, M. Survey over image thresholding / M. Sezgin, B. Sankur // Journal of Electronic Imaging. – 2004. – Vol. 13, № 1. – P. 146–165.

Поступила 02.03.12

*Объединенный институт проблем  
информатики НАН Беларуси,  
Минск, Сурганова, 6  
e-mail: lexatam@newman.bas-net.by*

**D.V. Pradun, A.A. Mamchich**

### **THE ALGORITHM OF AUTOMATIC SHADOW DETECTION USING ENTROPIC THRESHOLDING METHOD**

An algorithm for automatic shadow detection is suggested, which is based on the representation of digital image in the color space  $YUV$ , succeeding analysis of histograms and entropic threshold method for detection of the shadow pixels. The algorithm admits parallel modification for multiprocessor computing systems without complication of the program implementation.

УДК 004.93'1; 004.932

А.И. Кравчонок

## АЛГОРИТМ БЫСТРОГО ВЫЧИСЛЕНИЯ ОПТИЧЕСКОГО ПОТОКА ПРИ ПОМОЩИ SSE2-ИНСТРУКЦИЙ ПРОЦЕССОРОВ СЕМЕЙСТВА x86

*Представлен алгоритм быстрого вычисления оптического потока при помощи SSE2-инструкций на персональном компьютере. Алгоритм имеет константную сложность в зависимости от радиуса окна оптического потока, применяет SSE2 SIMD-инструкции на всех этапах вычислений, при работе на многоядерных процессорах использует параллельный режим работы. Алгоритм позволяет значительно ускорить вычисление оптического потока, что делает возможным его применение в режиме реального времени на персональных компьютерах.*

### Введение

Оптический поток – это смещение каждой точки данного изображения по отношению к предыдущему изображению на видеопоследовательности. Вычисление оптического потока – одна из центральных проблем в компьютерном зрении. Различные методы решения данной задачи находят применение во многих приложениях, таких как навигация роботов [1], отслеживание объектов (людей, транспортных средств) [2–4], видеонаблюдение и контроль доступа [5–7], сжатие видеопоследовательностей [8–12], стереовосстановление [1, 13–22] и др. Многие приложения нуждаются в вычислении оптического потока в режиме реального времени, однако достичь такой производительности на персональном компьютере (ПК) все еще сложно. Существующие методы быстрого вычисления плотного поля оптического потока [1, 13, 23, 24] не позволяют использовать режим реального времени на персональных компьютерах при достаточно больших размерах изображения и достаточно больших возможных смещениях объектов на изображениях, не жертвуя при этом точностью вычислений и не используя различные упрощения и приближения [11, 12, 25].

При этом необходимо заметить, что наиболее часто применяемыми разрешениями видеопоследовательностей в настоящее время являются  $320 \times 240$ ,  $640 \times 480$ ,  $720 \times 576$  при использовании максимально возможных смещений пикселей изображения в пределах  $\pm 8$ ,  $\pm 16$  по осям  $X$  и  $Y$ . Можно также заметить, что с повышением разрешения зачастую необходимо увеличивать и значения максимально возможных смещений пикселей изображения. Вычислительной мощности современных процессоров персональных компьютеров при использовании известных алгоритмов недостаточно для вычисления оптического потока в режиме реального времени при указанных разрешениях и максимально возможных смещениях пикселей изображения. Тенденции в развитии современных процессоров указывают на то, что в ближайшем будущем повышение их производительности будет достигаться за счет увеличения количества их вычислительных ядер, которые могут выполнять вычисления в параллельном режиме. Таким образом, на данный момент актуальной является разработка алгоритма вычисления оптического потока, который может эффективно выполняться в параллельном режиме на многоядерных процессорах.

В статьях Сана [23, 24] предлагается алгоритм вычисления оптического потока, который позволяет получить плотную карту оптического потока при помощи быстрой корреляции и техники поиска минимального 2D- либо 3D-пути в 3D-объеме куба коэффициентов корреляции. Быстрая корреляция вычисляется при помощи алгоритма бегущего окна, который инвариантен к размеру окна корреляции. Автор рассматривает такие меры сходства окон, как нормированная центрированная корреляция (ZNCC – zero mean normalized cross-correlation), сумма квадратов разности (SSD – sum of squared difference), сумма модулей разности (SAD – sum of absolute difference). Однако даже при таком подходе алгоритм не позволяет достичь производительности реального времени для сколь-нибудь существенного количества возможных сдвигов (области допустимых скоростей движущихся объектов).

В статьях [5, 6, 26] предлагается использовать алгоритм Сана для вычисления оптического потока в задаче поиска и отслеживания движущихся объектов в режиме реального

времени. Для ускорения вычислений применяются команды MMX (multimedia extensions) и SSE (streaming SIMD extensions). Авторам удается достичь производительности реального времени при использовании небольших разрешений видеопоследовательностей и небольшом количестве рассматриваемых возможных сдвигов объектов, они применяют MMX- и SSE-команды для ускорения лишь наиболее трудоемких частей алгоритма вычисления оптического потока.

В статье [25] представляется алгоритм вычисления оптического потока почти в режиме реального времени на обычном оборудовании без специальных аппаратных средств. Однако алгоритм применяет пирамиду изображений, что в некоторых задачах (например, обнаружения и отслеживания быстрых объектов небольшого размера) может снижать точность результатов.

Многие быстрые алгоритмы стереовосстановления могут быть применены для вычисления оптического потока, среди них алгоритмы Фогераса и соавт. [1], Канады и соавт. [13], Сана [14–19], а также алгоритмы, описанные в статьях [21, 22]. Однако практически все из них используют эпиполярное ограничение, которое значительно сокращает количество возможных сдвигов при вычислении соответствия окон корреляции и позволяет достичь производительности реального времени только в задаче стереовосстановления. В задаче вычисления оптического потока применить эпиполярное ограничение не представляется возможным, так как оно не выполняется – в поле зрения камеры объекты могут двигаться в произвольных направлениях. Например, если в задаче стереовосстановления рассматриваются возможные смещения точек вдоль эпиполярных линий в пределах  $\pm 16$ , то необходимо рассмотреть только 33 сдвига, при таком же диапазоне смещений по осям  $X$  и  $Y$  для оптического потока необходимо вычислить уже 1089 оценок для соответствующих смещений. Таким образом, задача вычисления оптического потока в отличие от задачи стереовосстановления является вычислительно значительно более сложной и все указанные выше алгоритмы не позволяют решать ее в режиме реального времени для разумной области возможных смещений объектов.

Многие современные алгоритмы сжатия видео (MPEG-1/2/4/7 от moving picture experts group и H.26x-1/2/3/4 от video coding experts group) применяют оценку движения при помощи сопоставления блоков. Сопоставление блоков фактически дает несколько разреженную карту оптического потока, так как вектор смещения вычисляется не для всех точек изображения, а для некоторого количества непересекающихся блоков (чаще всего квадратных, наиболее часто используются размеры  $4 \times 4$ ,  $8 \times 8$ ,  $16 \times 16$ ), на которое разбивается изображение. Алгоритм сопоставления блоков имеет значительно меньшую вычислительную сложность по сравнению с оптическим потоком, так как вычисляет векторы смещения для значительно меньшего количества точек и при реализации на SSE2 может работать в режиме реального времени [8, 9]. Оптический поток имеет большую вычислительную сложность, так как вычисления производятся не для отдельных точек, а для всех точек изображения, поэтому применение алгоритмов оценки движения при помощи SSE2 не позволяет достичь скорости реального времени. Однако можно заметить сильную избыточность вычислений при применении алгоритмов оценки смещения блоков для вычисления оптического потока за счет пересечения соседних окон, чего не наблюдается при оценке смещения непересекающихся блоков.

На данный момент широкое применение оптического потока ограничивается его высокой вычислительной сложностью, поэтому многие авторы применяют для вычисления оптического потока приближительные методы [11, 12, 25]. Зачастую при вычислении оптического потока учитывают специфику задачи и вычисляют его не для всех точек изображения, а только для движущихся областей либо некоторых ключевых точек изображения [3, 7, 27–30]. При этом, вычисляя вместо плотной карты векторов движения только смещения отдельных областей или точек, не всегда удается достичь производительности реального времени без дополнительных ухищрений и ограничений.

В настоящей статье предлагается алгоритм быстрого вычисления оптического потока при помощи SSE2-команд процессоров семейства x86, который может выполняться в режиме реального времени на персональных компьютерах.

## 1. Эффективная реализация алгоритма оптического потока

Среди различных методик вычисления оптического потока можно выделить несколько наиболее часто применяемых [25, 31]:

- дифференциальный подход (differential) [31–33];
- алгоритмы поиска наилучшего соответствия областей (region-based matching) [31];
- частотные алгоритмы (energy based) [31];
- фазовые алгоритмы (phase based) [31].

Частотные алгоритмы дают малую точность вычислений [31]. Фазовые алгоритмы имеют высокую вычислительную сложность [31], а градиентные методы оптического потока, например метод Лукаса – Канаде [33], достигают высокой точности для сцен с малыми пиксельными смещениями (менее 1–2 пикселей на кадр), при этом хуже работают для больших смещений [11, 34]. Однако, например, в задачах обнаружения и отслеживания движущихся объектов на видеопоследовательностях, а также в задачах стереовосстановления смещения объектов от кадра к кадру могут быть большими (вплоть до нескольких десятков пикселей).

Из описанных методов наиболее простым и легко реализуемым является алгоритм поиска наилучшего соответствия областей, который широко применяется для решения задач компенсации движения при сжатии видео [10–12], поиска движущихся объектов [35, 36], стереорекострукции [1, 13–19, 21, 22]. Значительным плюсом алгоритма поиска наилучшего соответствия областей является возможность его реализации при помощи SSE2-команд, что значительно ускоряет вычисления [8, 9].

При существующих вычислительных мощностях персональных компьютеров вычисление оптического потока в режиме реального времени является сложной задачей, поэтому для ее решения были разработаны различные подходы:

*подвыборка* – вычисление оптического потока не во всех точках изображения. Такой подход могут использовать, например, алгоритмы на основе сопоставления блоков изображений [10–12, 35, 36], а также алгоритмы, вычисляющие оптический поток только в областях интереса [3, 7, 27–30];

*иерархический метод* – последовательное вычисление оптического потока на пирамиде изображений [33, 34];

*различные аппроксимации оптического потока* – например трехшаговый поиск (TSS – tree-step search), четырехшаговый поиск (FSS – four-step search), логарифмический поиск (LS – logarithmic search). Вычисления ускоряются за счет оценки не всех возможных сдвигов, а только тех, которые наиболее вероятно (в соответствии с некоторым предположением) дадут лучшую оценку [11, 12];

*отсечения* – проверка выполнения некоторого условия при вычислении оптического потока, которое позволяет отсеять заведомо плохие оценки до выполнения всех вычислений [12, 21, 22, 37];

*вычисление оптического потока при помощи бегущей суммы* – вычисление оценок для сдвигов данного окна выполняется на основе оценок, полученных для окна, соседнего с ним, что позволяет избежать большого количества лишних вычислений [1, 13, 23, 24].

Описанные подходы либо не дают точного вычисления оптического потока (подвыборка, иерархический метод, аппроксимации), либо не позволяют достичь скорости реального времени (отсечения, бегущая сумма).

Для ускорения вычисления оптического потока можно применять SIMD (single instruction, multiple data)-команды, например SSE2, что позволяет значительно ускорить вычисления [5, 6, 8, 9, 29, 30]. Возможны различные варианты использования подобных команд в зависимости от применяемого алгоритма вычисления оптического потока. Использование SSE2-команд для описываемого в статье алгоритма позволяет значительно ускорить вычисление оптического потока.

## 2. Алгоритм поиска наилучшего соответствия областей для вычисления оптического потока

В алгоритме поиска наилучшего соответствия областей вычисление оптического потока из первого изображения во второе осуществляется путем подбора для каждого пиксела второго

изображения наилучшего соответствия среди пикселей первого изображения в пределах заданного интервала возможных сдвигов. Оценки схожести пикселей вычисляются путем сравнения фиксированной окрестности пиксела (чаще всего квадратной или прямоугольной формы) на втором изображении со сдвинутой окрестностью в первом. Для каждого пиксела изображения осуществляются всевозможные сдвиги его окрестности в некоторых заранее заданных пределах, и для каждого сдвига подсчитывается оценка соответствия окрестности из второго изображения и той части первого изображения, на которое она легла. Существуют различные оценки для определения наилучшего соответствия двух частей изображений, чаще всего применяются следующие [1, 38]:

– корреляция (correlation between the two blocks – COR):

$$COR = \sum_{j=1}^M \sum_{i=1}^N (P_1(i, j) \times P_2(i, j));$$

– сумма квадратов разности (sum of squared difference – SSD):

$$SSD = \sum_{j=1}^M \sum_{i=1}^N (P_1(i, j) - P_2(i, j))^2;$$

– сумма модулей разности (sum of absolute difference – SAD):

$$SAD = \sum_{j=1}^M \sum_{i=1}^N |P_1(i, j) - P_2(i, j)|,$$

где  $P_1$  и  $P_2$  – прямоугольные части некоторых изображений размера  $M \times N$ , а  $P_1(i, j)$  и  $P_2(i, j)$  – яркости пикселей  $P_1$  и  $P_2$  с соответствующими координатами.

При поиске наилучшего соответствия первая мера максимизируется, а вторая и третья минимизируются.

С вычислительной точки зрения наилучшей мерой является SAD вследствие следующих причин:

- имеет более простую формулу;
- обладает возможностью эффективной реализации при помощи SSE2-команд;
- результат вычисления модуля разности занимает меньше места в памяти, чем квадрат разности и помещается в один байт, что позволяет обрабатывать параллельно большее количество пикселей.

При использовании SAD качество найденного оптического потока получается несколько худшим, чем при использовании других мер [13, 38], однако применение SAD при незначительном ухудшении качества оптического потока позволяет значительно повысить скорость вычислений [13]. Поэтому при вычислении оптического потока в качестве меры различия областей изображений будем использовать SAD.

Пусть  $I_1$  и  $I_2$  – два последовательных изображения размера  $IW \times IH$ ,  $p_1(x, y)$  – пиксели изображения  $I_1$ ,  $p_2(x, y)$  – пиксели изображения  $I_2$ , где  $0 \leq x < IW$ ,  $0 \leq y < IH$ , а  $I_1(x, y)$  и  $I_2(x, y)$  – яркости пикселей соответствующих изображений. Обозначим через  $W_1^r(x, y)$  прямоугольную окрестность пиксела  $p_1(x, y)$  размера  $(2r + 1) \times (2r + 1)$ :

$$W_1^r(x, y) = \{p_1(x', y') \mid x - r \leq x' \leq x + r, y - r \leq y' \leq y + r\}.$$

Окрестность  $W_2^r(x, y)$  пиксела  $p_2(x, y)$  определяется аналогично. Определим  $R$  как максимально возможное смещение пиксела  $p_2(x, y)$  изображения  $I_2$  по отношению к предыдущему изображению последовательности  $I_1$ . Тогда  $S = \{(u, v) \mid -R \leq u, v \leq R\}$  – множество возмож-

ных векторов движения  $(u, v)$  пиксела  $p_2(x, y)$ . Пусть  $W_1^r(x+u, y+v)$  и  $W_2^r(x, y)$  – окрестности радиуса  $r$  пикселей  $p_1(x+u, y+v)$  и  $p_2(x, y)$ , где  $(u, v) \in S$ . Определим меру отличия двух окрестностей  $W_1^r(x+u, y+v)$  и  $W_2^r(x, y)$ :

$$SAD_{(x,y)}(u, v) = \sum_{j=-r}^r \sum_{i=-r}^r |I_1(x+i+u, y+j+v) - I_2(x+i, y+j)|.$$

Тогда будем искать для пиксела  $p_2(x, y)$  его вектор движения  $(\hat{u}, \hat{v})$  как вектор  $(u, v)$ , для которого оценка  $SAD_{(x,y)}(u, v)$  минимальна:

$$(\hat{u}, \hat{v}) = \arg \min_{(u,v) \in S} SAD_{(x,y)}(u, v).$$

В такой постановке задачи вычисления векторов движения возможно использование только окрестности пикселей нечетного диаметра, что не всегда удобно. При необходимости использования окрестности пиксела четного диаметра будем задавать окрестность

$$W_1^r(x, y) = \{p_1(x', y') \mid x-r \leq x' < x+r, y-r \leq y' < y+r\}.$$

Окрестность четного диаметра  $W_2^r(x, y)$  пиксела  $p_2(x, y)$  определяется аналогично. Тогда вычисление  $SAD_{(x,y)}(u, v)$  для окрестностей с четным диаметром необходимо выполнять следующим образом:

$$SAD_{(x,y)}(u, v) = \sum_{j=-r}^{r-1} \sum_{i=-r}^{r-1} |I_1(x+i+u, y+j+v) - I_2(x+i, y+j)|.$$

С помощью указанных формул оптический поток можно вычислить не для всех пикселей изображения  $I_2$ , а только для тех, для которых при любом возможном сдвиге  $(u, v) \in S$ ,  $S = \{(u, v) \mid -R \leq u, v \leq R\}$ , и заданном радиусе окна оптического потока  $r$  существуют все пиксели окрестности  $W_1^r(x+u, y+v)$ , т. е. для пикселей  $p_2(x, y) \in P$ , где  $P = \{p_2(x, y) \mid R+r \leq x < IW - R - r, R+r \leq y < IH - R - r\}$  (рис. 1).

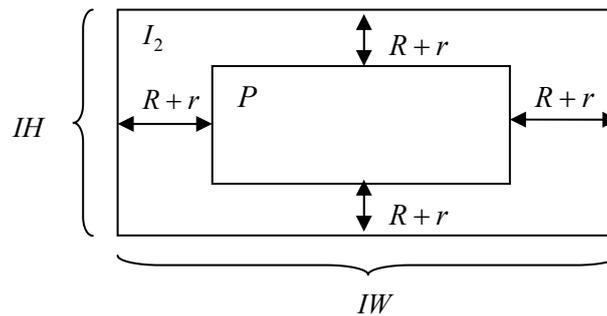


Рис. 1. Множество пикселей  $P$  изображения  $I_2$ , для которых можно вычислить оптический поток

Будем вычислять оптический поток только для множества пикселей  $P$ , так как для любого пиксела  $p_2(x, y) \notin P$  можно найти сдвиг  $(u', v') \in S$ , при котором часть окрестности  $W_1^r(x+u', y+v')$  будет выходить за пределы изображения.

Среди алгоритмов вычисления оптического потока на основе поиска наилучшего соответствия областей можно выделить алгоритмы, основанные на применении бегущего окна, сложность которых не зависит от размера окна оптического потока [1, 13, 23, 24]. Описание

эффективного вычисления оконных сумм на основе бегущего окна для изображений дано в [39]. Пусть  $I$  – изображение, для которого необходимо вычислить локальные суммы элементов по окну размера  $(2r + 1) \times (2r + 1)$ :

$$SUM(x, y) = \sum_{j=y-r}^{y+r} \sum_{i=x-r}^{x+r} I(i, j). \quad (1)$$

Обозначим через  $SUMV(x, y)$  сумму элементов окна  $1 \times (2r + 1)$ , образующих вертикальный столбец:

$$SUMV(x, y) = \sum_{j=y-r}^{y+r} I(x, j).$$

Перепишем формулу (1) следующим образом:

$$SUM(x, y) = \sum_{i=x-r}^{x+r} SUMV(i, y).$$

Заметим, что сумма  $SUMV(x, y)$  может быть вычислена на основе суммы  $SUMV(x, y - 1)$ :

$$SUMV(x, y) = SUMV(x, y - 1) - I(x, y - r - 1) + I(x, y + r), \quad (2)$$

а сумма  $SUM(x, y)$  – на основе суммы  $SUM(x - 1, y)$ :

$$SUM(x, y) = SUM(x - 1, y) - SUMV(x - r - 1, y) + SUMV(x + r, y). \quad (3)$$

Тогда локальные суммы элементов по окну можно находить при помощи алгоритма с константной сложностью, вычисляя в несколько этапов:

- начальные суммы  $SUMV(x, r + 1)$ ,  $x = \overline{1, IW}$ ;
- все суммы  $SUMV(x, y)$ ,  $x = \overline{1, IW}$ ,  $y = \overline{r + 2, (IH - r)}$ , при помощи бегущего окна (см. формулу (2));
- начальные суммы  $SUM(r + 1, y)$ ,  $y = \overline{r + 1, (IH - r)}$ ;
- все суммы  $SUM(x, y)$ ,  $x = \overline{r + 2, (IW - r)}$ ,  $y = \overline{r + 2, (IH - r)}$  при помощи бегущего окна (см. формулу (3)).

Вычисления, описанные выше, можно производить построчно, определяя  $SUMV(x, y)$  на основе формулы (2) не для всего изображения сразу, а только для одной строки, после чего можно вычислять  $SUM(x, y)$  на основе формулы (3) только для этой строки. После этого алгоритм переходит к следующей строке и выполняет для нее вычисления таким же образом. Построчное выполнение алгоритма требует меньшего объема памяти для хранения промежуточных вычислений, таких как бегущие вертикальные суммы  $SUMV(x, y)$ .

С помощью описанного алгоритма бегущего окна можно вычислять оценки  $SAD_{(x,y)}(u, v)$  за константное время в зависимости от размера окна оптического потока. К алгоритмам вычисления оптического потока, использующим указанный подход, можно отнести алгоритмы Фогераса и соавт. [1], алгоритм Канаде и соавт. [13], Сана [14–19, 23, 24]. Основным отличием алгоритма Сана от похожих алгоритмов Фогераса и соавт. [1] и Канады и соавт. [13] является реализация основного цикла алгоритма по сдвигам, а не по пикселям, т. е. в алгоритмах Фогераса и соавт., а также Канады и соавт. для каждого пикселя изображения сразу вычисляются все оценки для всевозможных сдвигов окрестности пикселя, после чего вычисления продолжают для следующих пикселей. В алгоритме Сана первоначально фиксируется сдвиг, для которого вычисляются оценки для окрестностей всех пикселей при дан-

ном сдвиге, после чего осуществляется переход к следующему сдвигу. Подобные небольшие отличия в организации циклов вычисления оценок для различных сдвигов окрестностей пикселей приводят к различной программной реализации алгоритмов, особенно при использовании SSE2-инструкций.

### 3. Реализация алгоритма вычисления оптического потока при помощи SSE2

SSE2 – дополнительный набор инструкций, позволяющий ускорить обработку мультимедиаданных (звука, видеопотоков, изображений). В настоящее время он поддерживается практически всеми процессорами как фирмы Intel, так и AMD. SSE2 включает в себя набор 128-битных регистров, а также дополнительные команды, которые позволяют выполнять некоторые арифметические операции параллельно. Например, за одну операцию можно параллельно складывать, вычитать либо выполнять другие операции над 16 байтами, 8 словами либо 4 двойными словами.

Одним из методов вычисления оптического потока на основе SSE2-команд является применение алгоритма сопоставления блоков, используемого для кодирования видеопоследовательностей (см. табл. 4, алгоритм 1) [9]. Сопоставление блоков применяется для окрестности каждого пикселя изображения, что приводит к значительно большим вычислительным затратам по сравнению с исходным алгоритмом, где сопоставление производится только для достаточно крупных блоков изображения ( $4 \times 4$ ,  $8 \times 8$ ,  $16 \times 16$ ). Очевидно, что время работы подобного алгоритма неудовлетворительно для вычислений в режиме реального времени. В процессорах Intel Pentium 4 появились новые SSE3-команды, в частности LDDQU, которые позволяют ускорить вычисления в алгоритме сопоставления блоков [8], но даже при их использовании скорость вычисления оптического потока не достигает реального времени.

Ускорить вычисления в указанных выше алгоритмах можно при помощи организации вычислений на основе бегущего окна. Однако специфика данных SSE2-алгоритмов позволяет получить на их основе только алгоритм с линейной сложностью вычислений в зависимости от размера окна оптического потока (см. табл. 4, алгоритм 2). Вычисления векторов смещения производятся по столбцам сверху вниз, при этом результаты оценок для всех смещений данного пикселя сохраняются в массиве и используются для вычисления оценок следующего пикселя в столбце.

Указанные выше алгоритмы хорошо работают с окнами оптического потока размера  $4 \times 4$ ,  $8 \times 8$ ,  $16 \times 16$ , однако их сложно адаптировать для работы с окнами произвольного размера.

Высокую скорость вычислений оптического потока можно достичь, используя алгоритмы бегущего окна с константной сложностью. Реализация алгоритма Фогераса и соавт. для стереовосстановления при помощи MMX-инструкций была предложена в статье [22]. Предлагаемый в данной статье алгоритм отличается от указанных выше тем, что он основан на алгоритмах Сана [14–19, 23, 24] и реализуется при помощи MMX- и SSE-команд на базе метода, отличного от SIMD-реализации алгоритма Фогераса и соавт.

Реализация алгоритмов Сана [14–19, 23, 24] вычисления оптического потока при помощи MMX- и SSE-команд описана в статьях [5, 6], однако применение SIMD-команд в указанных статьях ограничивается только наиболее трудоемкой частью алгоритма, в которой производится вычисление корреляции на основе оконных сумм, полученных без применения MMX- и SSE-команд. Авторы статей [5, 6] описывают реализацию вычисления оптического потока при помощи корреляции, однако применение SAD для вычисления оптического потока дает преимущество в более быстром вычислении оценок схожести блоков изображений (модуль разности вычисляется при помощи SSE2 значительно проще, чем корреляция), а также позволяет обрабатывать сразу по 16 элементов одновременно при вычислении модуля разности и по 8 элементов при суммировании, что ускоряет выполнение алгоритма за счет более высокой степени параллелизма и не требует использования вещественных чисел. Также при применении SAD, в отличие от корреляции, не нужно вычислять квадратные корни, что является достаточно трудоемкой операцией. В данной статье предлагается эффективная реализация алгоритма Сана на основе SSE2-команд, отличная от упомянутой выше.

Алгоритм Сана условно состоит из двух частей:

1) вычисление куба оценок для всех смещений каждого пиксела второго изображения относительно первого;

2) вычисление при помощи динамического программирования оптического потока на основе поиска максимальной поверхности в кубе.

Для получения быстрого алгоритма вычисления оптического потока будет использоваться алгоритм Сана со следующими модификациями:

1. В качестве меры отличия областей изображений используется SAD.

2. Оценки, вычисленные для всех пикселов при некотором фиксированном сдвиге, не сохраняются в массиве (кубе) всех оценок, а оцениваются сразу, при этом сохраняются только наилучшие оценки для каждого пиксела.

3. Реализация всех этапов работы алгоритма основана на применении SSE2-инструкций.

Зададим максимальный размер окна оптического потока  $16 \times 16$ , для того чтобы любая возможная оценка SAD для окрестности некоторого пиксела помещалась в двухбайтовую ячейку памяти. Пусть  $S = \{(u, v) \mid -R \leq u, v \leq R\}$  – множество возможных сдвигов. Пронумеруем все сдвиги из множества  $S$  и пусть  $S_i = (u', v')$  – сдвиг с порядковым номером  $i$ , где  $1 \leq i \leq (R+1)^2$ . Пусть  $M$  – двумерный массив оценок SAD и  $M(x, y)$  – наилучшая полученная на данный момент оценка SAD для пиксела с координатами  $(x, y)$  изображения  $I_2$ . Пусть  $MU, MV$  – двумерные массивы, содержащие сдвиги по осям  $X$  и  $Y$ , при которых получены наилучшие на данный момент оценки SAD, хранящиеся в массиве  $M$ :  $MU(x, y) = \bar{u}$ ,  $MV(x, y) = \bar{v}$ , где  $(\bar{u}, \bar{v})$  – сдвиг, при котором получена оценка  $M(x, y)$ .

Получим следующий алгоритм вычисления оптического потока:

1. Пусть  $i = 1$ ,  $M(x, y) = MAX$ , где  $0 \leq x < IW$ ,  $0 \leq y < IH$ , а  $MAX$  – некоторая константа, заведомо превышающая любую возможную оценку SAD для данного размера окна оптического потока.

2. Зафиксируем сдвиг  $S_i$ .

3. Вычислим модуль разности двух изображений  $I_1$  и  $I_2$  с учетом сдвига  $S_i$  и сохраним в двумерный массив  $D$ :

$$D(x, y) = |I_1(x + u', y + v') - I_2(x, y)|,$$

где  $(x, y) \in PD$ ,  $PD = \{(x, y) \mid R \leq x < IW - R, R \leq y < IH - R\}$  – множество координат пикселов изображения  $I_2$ , для которых можно вычислить модуль разности с пикселями изображения  $I_1$  при любом сдвиге из множества  $S$ .

4. При помощи вертикального бегущего окна размера  $1 \times (2r + 1)$ , где  $1$  – ширина окна, а  $(2r + 1)$  – высота окна, вычислим вертикальные суммы модулей разности по столбцам массива  $D$  и сохраним в двумерный массив  $V$ :

$$V(x, y) = \sum_{k=-r}^r D(x, y + k),$$

где  $(x, y) \in PV$ ,  $PV = \{(x, y) \mid R \leq x < IW - R, R + r \leq y < IH - R - r\}$  – множество координат элементов массива  $D$ , для которых можно вычислить вертикальные суммы по столбцам при любом сдвиге из множества  $S$  и данном радиусе окна оптического потока  $r$ .

5. При помощи горизонтального бегущего окна размера  $(2r + 1) \times 1$ , где  $(2r + 1)$  – ширина окна, а  $1$  – высота окна, вычислим горизонтальные суммы по строкам массива  $V$  и сохраним в двумерный массив  $H$ :

$$H(x, y) = \sum_{k=-r}^r V(x + k, y),$$

где  $(x, y) \in PH$ ,  $PH = \{(x, y) | R + r \leq x < IW - R - r, R + r \leq y < IH - R - r\}$  – множество координат элементов массива  $V$ , для которых можно вычислить горизонтальные суммы по строкам при любом сдвиге из множества  $S$  и данном радиусе окна оптического потока  $r$ .

6. Сравним полученные оценки SAD, сохраненные в массиве  $H$ , с оценками, хранящимися в массиве  $M$ : если  $H(x, y) < M(x, y)$ , то  $M(x, y) = H(x, y)$ ,  $MU(x, y) = u'$ ,  $MV(x, y) = v'$ .

7.  $i = i + 1$ ; если  $i \leq (R + 1)^2$ , идем на шаг 2.

8. Конец.

Таким образом, после выполнения алгоритма массивы  $MU, MV$  будут содержать векторы движения пикселей изображения  $I_2$ , а массив  $H$  – оценки SAD для соответствующих пикселей.

Представленный выше алгоритм использует SSE2-команды на всех этапах вычисления оптического потока, в частности, для получения:

- модуля разности изображений со сдвигом;
- вертикальных бегущих сумм (как вертикальных, так и горизонтальных);
- горизонтальных бегущих сумм.

Опишем подробно реализацию каждого шага алгоритма при помощи SSE2-инструкций. Так как в процессе вычисления оптического потока бегущая сумма вычисляется много раз – столько же, сколько рассматривается сдвигов, то скорость ее вычисления, как и скорость вычисления модуля разности изображений, должна быть как можно выше. Процесс вычисления бегущей суммы состоит из двух этапов. Первоначально вычисляется вертикальная бегущая сумма, затем – горизонтальная. Если максимальное окно оптического потока не превосходит  $16 \times 16$ , то вертикальное и горизонтальное суммирование кадров, представляющих собой модуль разности двух исходных кадров со сдвигом, можно выполнять в элементах типа WORD, так как максимальная сумма элементов блока размера  $16 \times 16$  не превосходит  $255 \times 16 \times 16$ , что помещается в ячейку памяти WORD. В таком случае при вычислении вертикальной и горизонтальной бегущих сумм при помощи SSE2-команд можно добиться параллельного вычисления восьми бегущих сумм.

*Вычисление модуля разности двух кадров со сдвигом при помощи SSE2-команд.* Для вычисления оптического потока необходимо определить, сколько модулей разностей изображений со сдвигом и сколько сдвигов рассматривается при вычислении оптического потока. Таким образом, при интервале возможных сдвигов  $\pm 16$  по осям  $X$  и  $Y$  необходимо вычислить 1089 модулей разности исходных кадров со сдвигом. При размере изображений, например,  $720 \times 576$  функция вычисления модуля разности и модуля разности со сдвигом должна быть хорошо оптимизирована, чтобы занимать как можно меньше времени. Чтобы уложиться в режим реального времени (40 мс на обработку одного кадра) при вычислении оптического потока, вычисления одного модуля разности изображений должно занимать время, в несколько раз меньшее, чем  $40/1089 = 0,036$  мс.

Существует несколько способов реализации эффективного вычисления модуля разности при помощи SSE2-команд:

1. Применение арифметики с насыщением [40]. Пусть в регистрах R1 и R2 находятся 16 байтовых элементов двух изображений, тогда модуль разности этих 16 элементов определяется следующим образом:

```
R3 = _mm_subs_epu8 (R1,R2);
R4 = _mm_subs_epu8 (R2,R1);
R4 = _mm_or_si128 (R4,R3).
```

2. Применение команд поиска минимума и максимума. Для тех же регистров R1 и R2 модуль разности 16 байтовых элементов можно искать таким образом:

```
R3 = _mm_min_epu8(R1,R2);
R4 = _mm_max_epu8(R1,R2);
R4 = _mm_subs_epu8(R4,R3).
```

Модуль разности изображений вычисляется со сдвигом второго изображения относительно первого. Для чтения данных со сдвигом из изображения используется SSE2-команда MOVDQU (интриндик `_mm_loadu_si128`), которая позволяет загрузить из памяти в SSE2-регистр сразу

16 байт. При поддержке процессором SSE3-команд вместо MOVDQU-команды возможно использование LDDQU-команды (интринсик `_mm_lddqu_si128`), которая позволяет быстрее читать данные по адресам в памяти, не выровненным на границу в 16 байт. Если организовать вычисление модуля разности специальным образом (использовать изображения с шириной, кратной 16, и максимальный сдвиг, кратный 16), то данные со сдвигом из изображения могут читаться MOVDQA-командой для выровненных данных (интринсик `_mm_load_si128`), что также увеличит быстродействие.

Сравнительное быстродействие вычислений модуля разности двух кадров со сдвигом как при помощи SSE2-команд, так и без них приведено в табл. 1.

Таблица 1  
Быстродействие операций поиска модуля разности кадров

Операция	Размер изображения	Время, мс
Вычисление модуля разности без SSE2-команд	320×240	0,077
	640×480	0,340
	720×576	0,490
Вычисление модуля разности с SSE2-командами	320×240	0,006
	640×480	0,039
	720×576	0,053

*Вычисление вертикальных бегущих сумм по столбцам изображения при помощи SSE2-команд.* Будем учитывать тот факт, что результатом вычисления модуля разности двух изображений со сдвигом является байтовый массив, который будем называть изображением модуля разности кадров. При этом вертикальные бегущие суммы по столбцам могут не помещаться в байтовое значение, поэтому хранить их необходимо в двухбайтовых ячейках памяти (при высоте окна оптического потока не более 256 пикселей).

Рассмотрим первые 16 столбцов изображения модуля разности кадров:  $S_1, \dots, S_{16}$ . Пусть  $S_n(i)$  –  $i$ -й элемент столбца с номером  $n$  сверху, а  $H = 2 \times r + 1$  – высота вертикального окна, в котором необходимо вычислить сумму элементов столбца, равную высоте окна оптического потока, тогда  $SUM_n(i)$  – сумма элементов  $i$ -го окна бегущей суммы:

$$SUM_n(i) = \sum_{k=i}^{i+H-1} S_n(k).$$

Каждая сумма  $SUM_n(i+1)$  за исключением первой суммы  $SUM_n(1)$  вычисляется на основе предыдущей суммы по формуле

$$SUM_n(i+1) = SUM_n(i) - S_n(i) + S_n(i+H).$$

Тогда вычисление вертикальных бегущих сумм для рассматриваемых 16 столбцов  $S_1, \dots, S_{16}$  при помощи SSE2-инструкций осуществляется следующим образом:

1.  $i = 1$ ,  $SUM_1 = 0$ ,  $SUM_2 = 0$ .
2. Копируем в SSE2-регистр  $REG_1$  16  $i$ -х элементов рассматриваемых 16 столбцов:

$$REG_1 = S_1(i), \dots, S_{16}(i).$$

3. Преобразуем 16 байтовых элементов, находящихся в регистре  $REG_1$ , в 16 двухбайтовых элементов и помещаем первые 8 в регистр  $REG_2$  и вторые 8 – в регистр  $REG_3$ .

4. Аккумулируем полученные элементы в двух регистрах  $SUM_1$  и  $SUM_2$  :

$$SUM_1(i) = SUM_1(i) + REG_2(i) ;$$

$$SUM_2(i) = SUM_2(i) + REG_3(i) .$$

5.  $i = i + 1$ ; если  $i \leq H$  , идем на шаг 2.

6. Сохраняем 16 полученных вертикальных сумм из регистров  $SUM_1$  и  $SUM_2$  в массив  $V$  .

7. Копируем в регистр  $REG_1$  следующие 16  $i$ -х элементов:

$$REG_1 = S_1(i), \dots, S_{16}(i) .$$

8. Преобразуем 16 байтовых элементов, находящихся в регистре  $REG_1$  , в 16 двухбайтовых элементов и помещаем первые 8 в регистр  $REG_2$  и вторые 8 – в регистр  $REG_3$  .

9. Копируем в регистр  $REG_4$  16 элементов с индексом  $(i - H)$  :

$$REG_4 = S_1(i - H), \dots, S_{16}(i - H) .$$

10. Преобразуем 16 байтовых элементов, находящихся в регистре  $REG_4$  , в 16 двухбайтовых элементов и помещаем первые 8 в регистр  $REG_5$  и вторые 8 – в регистр  $REG_6$  .

11. Вычисляем новые суммы на основе имеющихся 16 сумм при помощи бегущего окна:

$$SUM_1(i) = SUM_1(i) - REG_5(i) + REG_2(i) ;$$

$$SUM_2(i) = SUM_2(i) - REG_6(i) + REG_3(i) .$$

12. Сохраняем 16 полученных вертикальных сумм из регистров  $SUM_1$  и  $SUM_2$  в массив  $V$  .

13.  $i = i + 1$ ; если  $i \leq IH$  , то идем на шаг 7.

14. Конец.

Шаги 1–6 алгоритма выполняют вычисление первых 16 сумм, на основе которых при помощи бегущего окна вычисляются все остальные суммы (шаги 7–14). Вычисление вертикальных сумм для остальных столбцов изображения выполняется аналогичным образом. Сравнительное быстродействие вычисления вертикальных бегущих сумм как при помощи SSE2-команд, так и без них отражено в табл. 2.

Таблица 2  
Быстродействие операции поиска вертикальных бегущих сумм

Операция	Размер изображения	Время, мс
Вычисление вертикальных бегущих сумм без SSE2-команд	320×240	0,286
	640×480	1,549
	720×576	2,117
Вычисление вертикальных бегущих сумм с SSE2-командами	320×240	0,010
	640×480	0,058
	720×576	0,080

*Вычисление горизонтальных бегущих сумм по строкам изображения при помощи SSE2-команд.* Данный шаг алгоритма сложно реализовать при помощи SSE2-инструкций, применяя метод, аналогичный вычислению вертикальных бегущих сумм, так как SSE2-команды чтения элементов массива работают только с горизонтально расположенными данными. Для реализации алгоритма горизонтальной бегущей суммы по аналогии с вертикальной бегущей

суммой необходимо было бы выполнять чтение сразу восьми элементов массива, расположенных вертикально, что невозможно осуществить при помощи одной SSE2-команды. Для решения данной задачи можно предложить несколько подходов:

1. Транспонирование массива и применение алгоритма вертикальной бегущей суммы.

2. Чтение из массива прямоугольной части размера  $8 \times 8$  при помощи восьми SSE2-команд чтения и транспонирование полученного квадрата с помощью SSE2-команд. Последующие вычисления производятся со строками транспонированного квадрата, представляющими собой транспонированные столбцы по восемь элементов исходного массива.

3. Первоначальное преобразование изображений, для которых будет вычисляться оптический поток таким образом, чтобы можно было осуществить параллельные вычисления восьми бегущих горизонтальных сумм одновременно. После вычисления оптического потока на преобразованных изображениях необходимо обратное преобразование полученных данных, которое даст правильный массив векторов движения.

Первый вариант требует транспонирования всех изображений модуля разности двух кадров, полученных для всех рассматриваемых сдвигов. Например, для интервала сдвигов  $\pm 16$  по осям  $X$  и  $Y$  это потребует выполнения  $1089$  транспонирований изображений (общее количество операций чтения и записи элементов будет равно  $1089 \times IW \times IH \times 2$ ), что занимает значительное время и не позволяет реализовать вычисления достаточно быстро.

Второй вариант фактически повторяет первый, только с транспонированием при помощи SSE2-команд. При проходе по всему изображению с вычислением горизонтальных бегущих сумм необходимо будет транспонировать практически все квадратики  $8 \times 8$ , на которые разбивается изображение. При этом данная процедура повторяется заново при каждом новом сдвиге, т. е., например, для интервала сдвигов  $\pm 16$  по осям  $X$  и  $Y$  она повторяется  $1089$  раз (транспонирование квадрата  $8 \times 8$  требует  $24$  SSE2-инструкции, тогда общее количество операций составляет  $1089 \times IW \times IH \times (24 / 64)$ ). Таким образом, при данном подходе общее количество операций уменьшается благодаря эффективной процедуре транспонирования, реализованной при помощи SSE2-команд, однако число излишних вычислений остается большим.

Третий вариант предусматривает смещение восьми частей изображения таким образом, что в каждой строке полученного изображения будут содержаться данные восьми частей изображения, при этом вычисление горизонтальных бегущих сумм можно выполнять параллельно, используя только SSE2-команды чтения горизонтально расположенных данных в рассматриваемой строке. Этот подход содержит дополнительные этапы предварительного смещения частей изображения и этап постобработки по восстановлению изображения из смешанных частей. Данные этапы выполняются только по одному разу и не занимают много вычислительного времени (общее количество операций чтения и записи элементов будет равно  $IW \times IH \times 2$ ). При этом остальные этапы вычисления оптического потока (вычисление модуля разности и вычисление вертикальных бегущих сумм), как и раньше, выполняются в параллельном режиме по описанным выше алгоритмам, лишь с небольшими отличиями. При использовании данного варианта вычисления горизонтальных бегущих сумм количество операций хотя и увеличивается за счет общего увеличения размера обрабатываемого изображения, однако остается наименьшим из описанных трех вариантов.

Для вычисления горизонтальной бегущей суммы при помощи SSE2 будем использовать метод предварительного разбиения изображения на восемь частей и их перемешивания. Тогда вычислять горизонтальную сумму можно таким же образом, как в реализации горизонтальной суммы без SSE, только читать и обрабатывать необходимо сразу восемь WORD-элементов. Каждый из этих восьми элементов будет относиться к разным частям изображения и, таким образом, можно будет вычислять параллельно восемь бегущих сумм для восьми разных частей изображения.

Для вычисления восьми горизонтальных бегущих сумм необходимо разбить исходные изображения на восемь частей. При этом, так как бегущая сумма вычисляется только для множества пикселей  $P$ , необходимо разбивать именно эту область на части. Необходимо также учесть, что для вычисления крайних пикселей каждой части этой области необходимо иметь данные о пикселях, выходящих за пределы этой части.

Разобьем область  $P$  изображения  $I_2$  на восемь одинаковых частей  $P_0, \dots, P_7$  (рис. 2). Обозначим через  $WP_i$  область  $P_i$ , расширенную на  $R + r$  в каждую сторону (рис. 3).

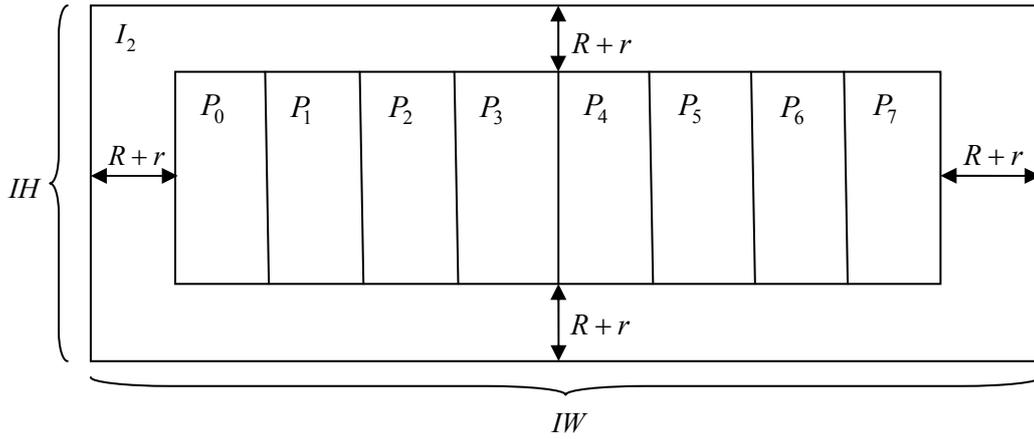


Рис. 2. Разбиение множества пикселей  $P$  изображения  $I_2$  на части

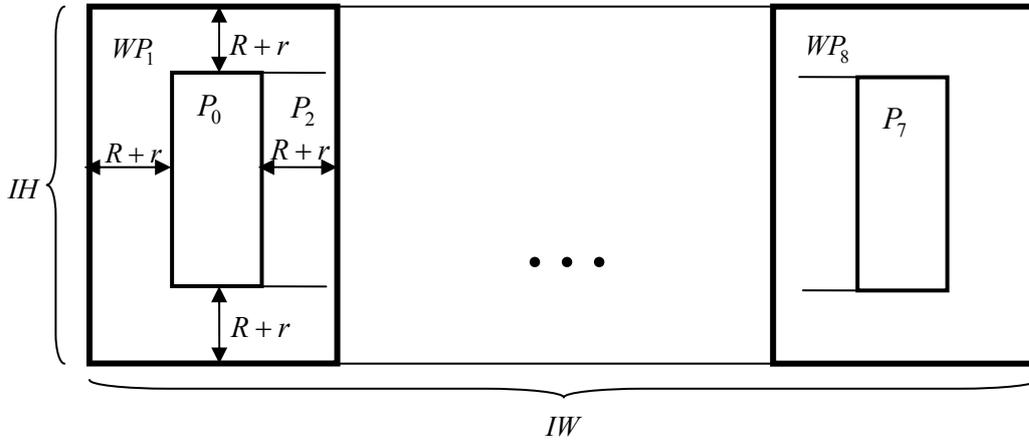


Рис. 3. Расширенные области  $P_i$  изображения  $I_2$

Таким образом, получим восемь пересекающихся областей  $WP_i, i=0, \dots, 7$ . Если  $I_2$  имеет размеры  $IW \times IH$ , область изображения  $P$  будет иметь размер  $PW \times PH = (IW - 2 \times (R+r)) \times (IH - 2 \times (R+r))$ , а каждая область  $P_i$  будет иметь размер  $(PW/8) \times PH$ . Следовательно, каждая расширенная область изображения  $WP_i$  будет иметь размер  $(\frac{PW}{8} + 2 \times (R+r)) \times (PH + 2 \times (R+r))$ . Обозначим через  $NC = \frac{PW}{8} + 2 \times (R+r)$  количество столбцов в расширенной области  $WP_i$ ,  $C_i(k)$  –  $k$ -й столбец расширенной области  $WP_i$  (нумерация столбцов начинается с нуля). Сформируем из расширенных областей  $WP_i$  изображения  $I_2$  новое изображение  $I'_2$  следующим образом. Будем располагать последовательно столбцы расширенных областей  $WP_i$ : сначала все первые столбцы, потом вторые и т. д. Пусть  $S_i$  –  $i$ -й столбец изображения  $I'_2$  (нумерация столбцов начинается с нуля), тогда

$$S_i = C_{i-8 \times \lfloor \frac{i}{8} \rfloor} \left( \left\lfloor \frac{i}{8} \right\rfloor \right).$$

Столбцы  $S_i$  образуют изображение  $I'_2$ , размер которого составит  $(8 \times NC) \times IH$ . Таким образом, изображение  $I'_2$  имеет ширину  $IW'_2 = 8 \times NC = IW + 7 \times 2 \times (R+r)$ , что больше, чем у изображения  $I_2$ , на  $2 \times (R+r) \times 7$  столбцов. Например, при размере кадра видеопоследовательности  $720 \times 576$ , радиусе окна оптического потока  $r = 8$  и максимально возможном смещении пиксела

$R=16$  ширина преобразованного кадра составит  $720 + 7 \times 2 \times (8 + 16) = 1056$ , а общий размер преобразованного кадра –  $1056 \times 576$ . Описанное выше преобразование необходимо производить как для изображения  $I_2$ , так и для изображения  $I_1$ . Однако можно заметить, что при обработке видеопоследовательностей достаточно выполнять одно преобразование изображения для каждого нового кадра видеопоследовательности, так как второй кадр, необходимый для вычисления оптического потока, уже будет преобразован на предыдущем шаге. Так как соседние столбцы исходных изображений  $I_1$  и  $I_2$  лежат в преобразованных изображениях  $I'_1$  и  $I'_2$  с шагом в восемь столбцов, становится возможным читать при помощи одной SSE2-операции чтения данных элементы сразу восьми различных столбцов изображения и оперировать при вычислении горизонтальной бегущей суммы сразу восемью элементами одновременно. Это позволяет вычислять бегущие горизонтальные суммы аналогично вычислению вертикальных бегущих сумм одновременно для восьми различных частей исходных изображений  $I_1$  и  $I_2$ . Также заметим, что количество операций, необходимых для вычисления оптического потока на полученном изображении, возрастает непропорционально увеличению площади изображения. Для полученного изображения  $1056 \times 576$  по сравнению с исходным изображением  $720 \times 576$  количество операций возрастает в 1,16 раза, тогда как площадь первого больше, чем площадь второго, в 1,46 раза.

Сравнительное быстроедействие вычислений горизонтальных бегущих сумм как при помощи SSE2-команд, так и без них приведено в табл. 3.

Таблица 3  
Быстроедействие операций поиска горизонтальных бегущих сумм

Операция	Размер изображения	Время, мс
Вычисление горизонтальных бегущих сумм без SSE2-команд	320×240	0,158
	640×480	0,730
	720×576	0,956
Вычисление горизонтальных бегущих сумм с SSE2-командами	320×240	0,017
	640×480	0,071
	720×576	0,099

Оптический поток для полученных после преобразования изображений  $I'_1$  и  $I'_2$  вычисляется при помощи описанного выше алгоритма с небольшими отличиями:

– при вычислении модуля разности двух кадров со сдвигом каждый сдвиг по оси  $X$  необходимо увеличивать в восемь раз;

– при вычислении горизонтальной бегущей суммы можно использовать алгоритм, аналогичный вычислению вертикальных бегущих сумм на основе SSE2, что позволяет вычислять параллельно по восемь бегущих сумм;

– после вычисления оптического потока на преобразованных изображениях необходимо выполнить обратное преобразование оптического потока.

Необходимо заметить, что потери производительности за счет увеличения размеров изображений по отношению к исходным меньше, чем выигрыш, получаемый за счет параллельной обработки при вычислении горизонтальных бегущих сумм.

#### 4. Результаты тестирования алгоритма быстрого вычисления оптического потока при помощи SSE2-команд

Работа алгоритма тестировалась на примере реальной видеопоследовательности, содержащей изображения различных движущихся объектов (транспорта, людей) при разрешении кадров  $320 \times 240$ ,  $640 \times 480$ ,  $720 \times 576$  (рис. 4) на персональном компьютере с процессором Intel Core i5 750. Кадры видеопоследовательности при более низких разрешениях были получены масштабированием кадров исходной видеопоследовательности с разрешением  $720 \times 576$ .

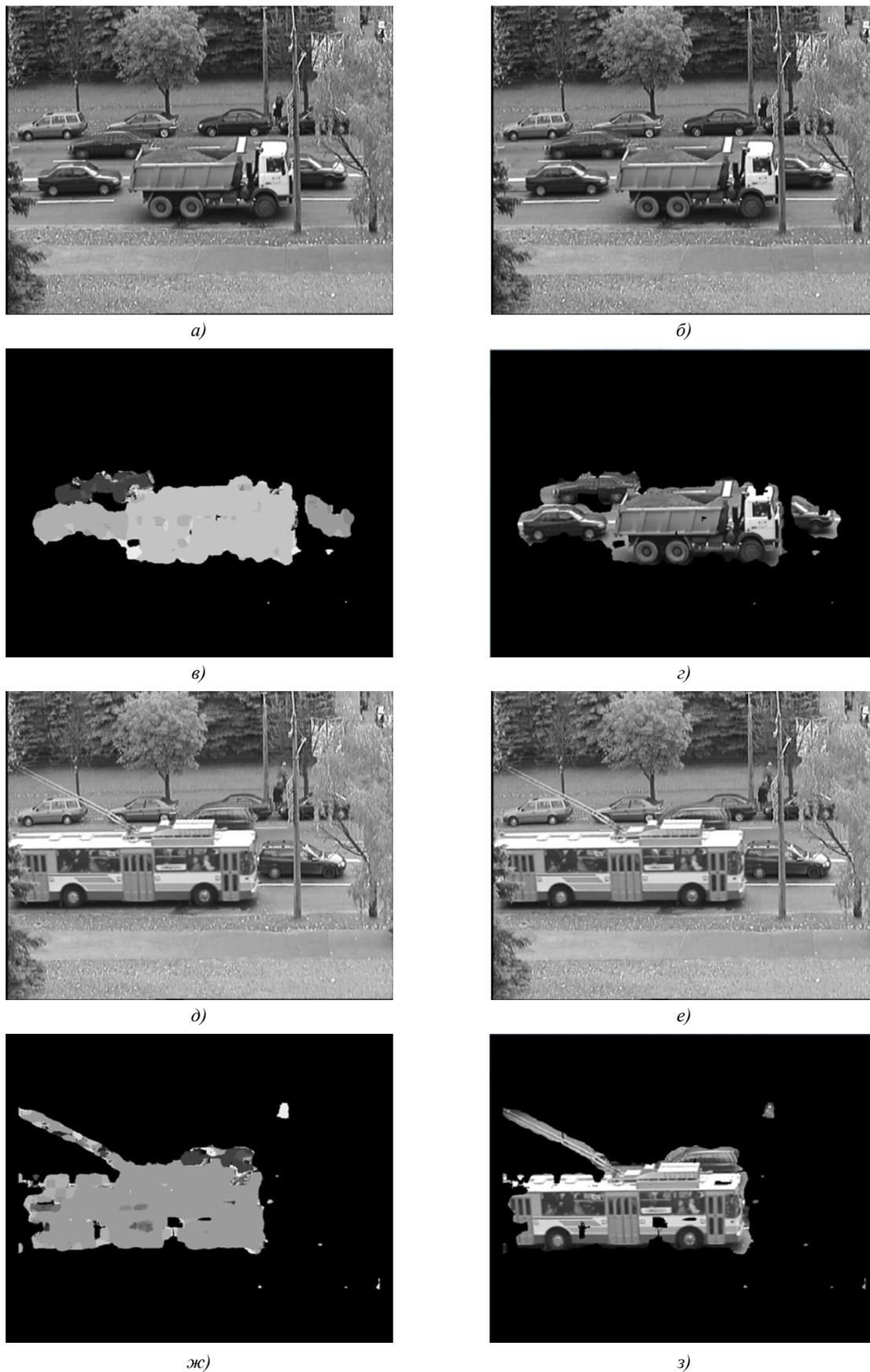


Рис. 4. Результаты работы алгоритма вычисления оптического потока: а), д) первый кадр пары; б), е) второй кадр пары; в), ж) вычисленный оптический поток; г), з) обнаруженные движущиеся объекты

Результаты быстродействия алгоритма на персональном компьютере с процессором Intel Core i5 750 с использованием одного ядра приведены в табл. 4. Можно заметить, что при разрешении  $320 \times 240$ , окне оптического потока  $16 \times 16$  и максимальных возможных сдвигах  $\pm 8$  алгоритм практически достигает быстродействия реального времени (15 кадров/с).

Таблица 4  
Быстродействие алгоритмов вычисления оптического потока с окном  $16 \times 16$

Алгоритмы оптического потока	Размер кадра видеопоследовательности	Интервал сдвигов	Время работы на кадр, мс
Алгоритм 1 оценки движения при помощи SSE2-команд	320×240	±8	291
	320×240	±16	1034
	640×480	±16	5028
	720×576	±16	6424
Алгоритм 2 оценки движения с линейной сложностью при помощи SSE2-команд	320×240	±8	102
	320×240	±16	332
	640×480	±16	1604
	720×576	±16	2213
Алгоритм Сана вычисления оптического потока без помощи SSE2-команд	320×240	±8	168
	320×240	±16	636
	640×480	±16	3096
	720×576	±16	4324
Быстрый алгоритм вычисления оптического потока при помощи SSE2-команд (горизонтальная бегущая сумма без SSE2-команд)	320×240	±8	78
	320×240	±16	289
	640×480	±16	1356
	720×576	±16	1860
Быстрый алгоритм вычисления оптического потока при помощи SSE2-команд (преобразование изображений)	320×240	±8	<b>49</b>
	320×240	±16	<b>176</b>
	640×480	±16	<b>768</b>
	720×576	±16	<b>1050</b>

Для реализации параллельной версии алгоритма была использована библиотека OpenMP. Результаты быстродействия параллельной версии алгоритма на персональном компьютере с процессором Intel Core i5 750 с использованием четырех ядер приведены в табл. 5.

Таблица 5  
Быстродействие параллельной версии алгоритма вычисления оптического потока с окном  $16 \times 16$  (четыре ядра процессора)

Алгоритм оптического потока	Размер кадра видеопоследовательности	Интервал сдвигов	Время работы на кадр, мс
Параллельная версия быстрого алгоритма вычисления оптического потока при помощи SSE2-команд (преобразование изображений)	320×240	±8	<b>25</b>
	320×240	±16	<b>85</b>
	640×480	±16	<b>252</b>
	720×576	±16	<b>328</b>

### Заключение

Предложен алгоритм быстрого вычисления оптического потока при помощи SSE2 SIMD-инструкций процессоров семейства x86. Алгоритм имеет константную сложность ( $O(1)$ ) в зависимости от радиуса окна оптического потока и применяет SSE2 SIMD-команды на всех этапах вычисления оптического потока (в том числе для вычисления горизонтальных и вертикальных бегущих сумм), что значительно ускоряет его работу. Применение SAD позволяет при помощи SSE2-команд одновременно обрабатывать большее количество пикселей изображения по сравне-

нию с корреляцией и SSD, а также быстрее вычислять меру отличия блоков изображений вследствие более простой формулы. При работе на многоядерных процессорах алгоритм использует параллельный режим работы, что позволяет ускорить вычисления. Скорость работы алгоритма на процессоре Intel Core i5 750 в режиме параллельного выполнения на четырех ядрах при разрешении видеопоследовательности  $320 \times 240$ , окне оптического потока  $16 \times 16$  и максимальных возможных сдвигах  $\pm 8$  по осям  $X$  и  $Y$  (всего вычисляется  $17 \times 17 = 289$  сдвигов) достигает 40 кадров в секунду, а при максимальных возможных сдвигах  $\pm 16$  (всего вычисляется  $33 \times 33 = 1089$  сдвигов) – 11 кадров в секунду. Это позволяет применять алгоритм для вычисления оптического потока в режиме реального времени на персональном компьютере. Алгоритм может быть полезен для решения таких задач, как обнаружение и отслеживание движущихся объектов на видеопоследовательностях.

### Список литературы

1. Faugeras, O. Real time correlation-based stereo: Algorithm, implementations and applications / O. Faugeras, B. Hotz, H. Mathieu // Technical Report RR-2013. – France : INRIA, 1993.
2. Yokoyama, M. A Contour-Based Moving Object Detection and Tracking / M. Yokoyama, T. Poggio // 2<sup>nd</sup> Joint IEEE Intern. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance. – Breckenridge, Colorado, USA, 2005. – P. 271–276.
3. Denman, S. An Adaptive Optical Flow Technique for Person Tracking Systems / S. Denman, V. Chandran, S. Sridharan // Pattern Recognition Letters. – 2007. – Vol. 28, № 10. – P. 1232–1239.
4. Kang, S. Estimation of Moving Information for Tracking of Moving Objects / S. Kang, J. Park, S. Jeong // Journal of Mechanical Science and Technology. – 2001. – Vol. 15, № 3. – P. 300–308.
5. Sadykhov, R. Fast cross correlation algorithm for optical flow estimation / R. Sadykhov, D. Lamovsky // Proc. of Nordic Signal Processing Symposium (NORSIG'2006). – Reykjavik, Iceland, 2006. – P. 322–325.
6. Sadykhov, R. Estimation of the cross correlation based optical flow for video surveillance / R. Sadykhov, D. Lamovsky // Computing. – 2006. – Vol. 5, № 3. – P. 112–117.
7. Tian, Y. Robust Salient Motion Detection with Complex Background for Real-time Video Surveillance / Y. Tian, A. Hampapur // IEEE Computer Society Workshop on Motion and Video Computing. – Breckenridge, Colorado, USA, 2005. – Vol. 2. – P. 30–35.
8. Intel Corp. Block-Matching in Motion Estimation Algorithms Using Streaming SIMD Extensions 3. – USA : Intel Applications Notes, 2003.
9. Intel Corp. Motion Estimation with Intel® Streaming SIMD Extensions 4 (Intel® SSE4) [Electronic resource]. – Mode of access : <http://software.intel.com/en-us/articles/motion-estimation-with-intel-streaming-simd-extensions-4-intel-sse4/>. – Date of access : 09.09.11.
10. Tran, T. Performance Enhancement of Motion Estimation Using SSE2 Technology / T. Tran, H. Cho, S. Cho // World Academy of Science, Engineering and Technology. – 2008. – Vol. 40. – P. 168–171.
11. Chen, Z. Efficient Block Matching Algorithm for Motion Estimation / Z. Chen // Intern. Journal of Signal Processing. – 2009. – Vol. 5, № 2. – P. 133–137.
12. Chen, Y. Fast block matching algorithm based on the winner-update strategy / Y. Chen, Y. Hung, C. Fuh // IEEE Transactions on Image Processing. – 2001. – Vol. 10, № 8. – P. 1212–1222.
13. Kanade, T. Development of a video-rate stereo machine / T. Kanade, H. Kato, S. Kimura // In Proc. of Intern. Robotics and Systems Conference (IROS'95). – Pittsburgh, Pennsylvania, USA, 1995. – Vol. 3. – P. 95–100.
14. Sun, A. Fast Stereo Matching Method / A. Sun // Proc. of Digital Image Computing: Techniques and Applications. – Auckland, New Zealand, 1997. – P. 95–100.
15. Sun, C. Multi-Resolution Rectangular Subregioning Stereo Matching Using Fast Correlation and Dynamic Programming Techniques / C. Sun // CMIS Report 98/246. – Australia : CSIRO, 1998.
16. Sun, C. Multi-Resolution Stereo Matching Using Maximum-Surface Techniques / C. Sun // Proc. of Digital Image Computing: Techniques and Applications. – Perth, Australia, 1999. – P. 195–200.
17. Sun, C. Fast Stereo Matching Using Rectangular Subregioning and 3D Maximum-Surface Techniques / C. Sun // Intern. Journal of Computer Vision. – 2002. – Vol. 47, № 1. – P. 99–117.

18. Sun, C. Fast Algorithms for Stereo Matching and Motion Estimation / C. Sun // Proc. of Australia-Japan Advanced Workshop on Computer Vision. – Adelaide, Australia, 2003. – P. 38–48.
19. Sun, C. Fast Panoramic Stereo Matching Using Cylindrical Maximum Surfaces / C. Sun, S. Peleg // IEEE Transactions on Systems, Man, and Cybernetics. Part B. – 2004. – Vol. 34, № 1. – P. 760–765.
20. Hirschmüller, H. Real-Time Correlation-Based Stereo Vision with Reduced Border Errors / H. Hirschmüller, P. Innocent, J. Garibaldi // International Journal of Computer Vision. – 2002. – Vol. 47, № 1–3. – P. 229–246.
21. Stefano, L. A Fast Area-Based Stereo Matching Algorithm / L. Stefano, M. Marchionni, S. Mattoccia // Image and Vision Computing. – 2004. – Vol. 22, № 12. – P. 983–1005.
22. Stefano, L. A PC-based Real-Time Stereo Vision System / L. Stefano, M. Marchionni, S. Mattoccia // Machine Graphics & Vision. – 2004. – Vol. 13, № 3. – P. 197–220.
23. Sun, C. Fast Optical Flow Using Cross Correlation and Shortest-Path Techniques / C. Sun // Proc. of Digital Image Computing: Techniques and Applications. – Perth, Australia, 1999. – P. 143–148.
24. Sun, C. Fast optical flow using 3d shortest path techniques / C. Sun // Image and vision computing. – 2002. – Vol. 20, № 13/14. – P. 981–991.
25. Zhao, P. Near Real-Time Optical Flow / P. Zhao, M. Spetsakis // Proc. of the 14th Intern. Conf. on Vision Interface. – Ottawa, ON, Canada, 2001. – P. 47–55.
26. Садыхов, Р.Х. Инструментальная система для обработки видеoinформации / Р.Х. Садыхов, Д.В. Ламовский // Доклады БГУИР. – 2007. – № 4 (20). – С. 175–180.
27. Chang, M. Optical Flow Measurement Based on Boolean Edge Detection and Hough Transform / M. Chang, I. Kim, J. Park // Intern. Journal of Control, Automation and Systems. – 2003. – Vol. 7, № 5. – P. 788–798.
28. Lu, N. Motion Detection Based On Accumulative Optical Flow and Double Background Filtering / N. Lu, J. Wang, L. Yang // Proc. of World Congress on Engineering. – London, UK, 2007. – P. 602–607.
29. Krauchonak, A. Detection of Moving Objects on Videosequences Based on Region Growing Optical Flow / A. Krauchonak // Proc. of the 10th International Conference «Pattern Recognition and Image Analysis: New Informational Technologies» (PRIA-10-2010), St. Petersburg, Russian Federation, December 5–12, 2010. – St. Petersburg, 2010. – Vol. 1. – P. 223–226.
30. Kravchonok, A. Detection of moving objects in video sequences by the computation of optical flow based on region growing / A. Kravchonok // Pattern Recognition and Image Analysis. – 2011. – Vol. 21, № 2. – P. 283–286.
31. Barron, J. Performance of optical flow techniques / J. Barron, D. Fleet, S. Beauchemin // Intern. Journal of Computer Vision. – 1994. – Vol. 12, № 1. – P. 43–77.
32. Horn, B. Robot Vision / B. Horn. – MIT Press. Cambridge, 1986. – 509 p.
33. Lucas, B. An Iterative Image Registration Technique with an Application to Stereo Vision / B. Lucas, T. Kanade // Proc of Intern. Joint Conference on Artificial Intelligence. – Vancouver, BC, Canada, 1981. – P. 674–679.
34. Anandan, P. A Computational Framework and an Algorithm for the Measurement of Visual Motion / P. Anandan // Intern. Journal of Computer Vision. – 1989. – Vol. 2. – P. 283–310.
35. Bartolini, F. Motion Estimation and Tracking for Urban Traffic Monitoring / F. Bartolini, V. Cappellini, C. Giani // Proc of IEEE Intern. Conf. on Image Processing. – Lausanne, Switzerland, 1996. – P. 787–790.
36. Stefano, L. Vehicle Detection and Tracking Using the Block Matching Algorithm / L. Stefano, E. Viarani // Proc. of Intern. Multiconference on Circuits, Systems, Communications and Computer. – Athens, Greece, 1999. – P. 4491–4496.
37. Mattoccia, S. Efficient and optimal block matching for motion estimation / S. Mattoccia, F. Tombari, L. Stefano // 14th IAPR Intern. Conf. on Image Analysis and Processing (ICIAP 2007). – Modena, Italy, 2007. – P. 705–710.
38. Toivonen, T. A New Algorithm for Fast Full Search Block Motion Estimation Based on Number Theoretic Transforms / T. Toivonen // Proc. of the 9th Intern. Workshop on Systems, Signals, and Image Processing. – Manchester, United Kingdom, 2002. – P. 90–94.

39. McDonnell, M. Box-filtering techniques / M. McDonnell // Computer Graphics and Image Processing. –1981. – Vol. 17, № 1. – P. 65–70.

40. Intel Corp. Absolute-Difference Motion Estimation for Intel® Pentium® 4 Processors [Electronic resource]. – Mode of access : <http://software.intel.com/en-us/articles/absolute-difference-motion-estimation-for-intel-pentiumr-4-processors/>. – Date of access : 31.08.10.

Поступила 26.01.12

*Объединенный институт проблем  
информатики НАН Беларуси,  
Минск, Сурганова, 6  
e-mail: alpha\_storm@mail.ru*

**A.I. Kravchonok**

**FAST OPTICAL FLOW COMPUTATION ALGORITHM BASED  
ON SSE2-INSTRUCTIONS OF X86 PROCESSOR FAMILY**

An algorithm for fast computation of optical flow based on SSE2-instructions of a PC is proposed. The algorithm has a constant complexity of the radius of the optical flow window, applies SSE2 SIMD-commands at all computation stages while working on multi-core processors and uses a parallel mode of operation. The algorithm can significantly speed up the computation of optical flow, which makes it applicable for real time mode of personal computers.

УДК 004.272.25, 004.622

Г.А. Прокопович

## ПРИМЕНЕНИЕ ГЕТЕРОАССОЦИАТИВНЫХ НЕЙРОННЫХ СЕТЕЙ ДЛЯ ЗАПИСИ И ВОССТАНОВЛЕНИЯ ИНФОРМАЦИИ

*Рассматривается проблема ассоциативного представления данных на примере рекуррентных искусственных нейронных сетей типа Хопфилда. Предложены: модификация сети «двунаправленная ассоциативная память», способ секционирования входных данных, основанный на их равномерном разбиении с дальнейшим последовательным отображением друг в друга, а также соответствующий метод их записи в гетероассоциативную память. Полученная нейросетевая архитектура способна сохранять, ассоциативно восстанавливать и распознавать большие объемы информации.*

### Введение

Одной из важнейших задач для современных вычислительных устройств является сохранение информации на носителе таким образом, чтобы впоследствии ее можно было извлечь, обладая лишь некоторым ключевым признаком этой информации. Как правило, предлагаются две различные формы сравнения входного ключа. С одной стороны, если среди совокупности данных требуется отыскать такие, которые на основе некоторой заданной меры близости соответствуют ключевой информации, то такая постановка задачи близка к постановке задачи распознавания образов. С другой стороны, задачи поиска информации по известному ключу, который сам содержит некоторую часть искомой информации, связаны с задачами оптимизации работы баз данных [1], а также способами ассоциативного представления данных [2].

Среди различных методов ассоциативной записи и восстановления информации можно выделить динамические нейросетевые архитектуры, обладающие обратной связью. К наиболее известным относятся однослойные сети Хопфилда [3–5] и рекуррентные многослойные перцептроны – сети Элмана [6]. Широкое практическое применение получили сети Элмана, которые используются для управления динамическими объектами [7, 8] и сжатия информации [9]. Однако для изучения принципов работы ассоциативной памяти чаще используются сети Хопфилда, которые демонстрируют, каким образом в принципе может быть организована устойчивая память, состоящая из коллектива недостаточно надежных нейроподобных элементов [9–11].

Характерной особенностью сетей Хопфилда является то, что процесс извлечения информации в таких сетях осуществляется по итеративному принципу и заключается в том, что на каждой новой итерации через обратные связи с элементами задержки  $\Delta t$  происходит обработка данных, полученных на предыдущем шаге. Для устойчивых сетей с обратными связями такая циркуляция информации происходит до тех пор, пока значения элементов сети не установятся в состояние равновесия, т. е. пока они не приобретут стационарные значения.

Для сети Хопфилда, в которой используется симметричная матрица весовых коэффициентов с нулевой главной диагональю, т. е.  $w_{ij} = w_{ji}$  и  $w_{ii} = 0$ , такими стационарными значениями – аттракторами – являются устойчивые стационарные точки или устойчивые предельные циклы длины два [12]:

$$y_i(t+k) = y_i(t), \quad i = \overline{1, n}, \quad k = 1, 2. \quad (1)$$

Тем не менее сети Хопфилда обладают определенными недостатками, которые значительно ограничивают их практическое применение. Как известно, чтобы восстановить (извлечь) необходимый образ с помощью обученной сети Хопфилда, необходимо иметь ключевой образ, совпадающий по размеру с искомым. Это приводит к невозможности реализации ассоциативного процесса восстановления информации в полном смысле слова, т. е. по ее отдельному фрагменту. В результате недостающие участки входного вектора находятся в состоянии неопределенности. Например, в многочисленных источниках [10, 11, 13] часто приводится задача

восстановления графических образов при их частичном искажении (рис. 1, а). Если память сети Хопфилда не переполнена и количество наложенного шума не превышает некоторого определенного порога, то она восстановит исходный образ (рис. 1, б). Однако сеть не способна качественно восстановить образ по его некоторой известной части (рис. 1, в), так как ключевой образ (рис. 1, г) будет слабо коррелировать с исходным. В то же время человеческий мозг прекрасно справляется с поставленной задачей и способен вспомнить (восстановить) большие объемы информации лишь по малой части ключевой информации [2].



Рис. 1. Различные варианты входных образов

Рассмотрим бинарный вектор  $\mathbf{L}_p = (l_1, l_2, \dots, l_p)$ , состоящий из  $p$  элементов, с помощью которого можно зашифровать некоторую информацию. Чтобы записать указанный информационный вектор  $\mathbf{L}_p$  с помощью сети Хопфилда, она должна состоять из  $n = p$  нейронов. Весовая матрица синаптических связей  $W_{n \times n}$  представляет собой квадратную матрицу размером  $n \times n$ , где  $n$  – длина входного вектора. Тогда с ростом длины  $p$  входного вектора  $\mathbf{L}_p$  общий объем памяти, необходимый для моделирования сети Хопфилда, будет расти не линейно, а квадратично. Таким образом, имеет место ограничение сверху на объем памяти, необходимый для хранения весовой матрицы  $W_{n \times n}$ . В связи с этим может сложиться такая ситуация, когда размер вектора  $\mathbf{L}_p$  значительно превысит число нейронов  $n$  во входном слое сети. Следующим недостатком сети Хопфилда является то, что для восстановления информационного вектора  $\mathbf{L}_p$  потребуется ключевой вектор, длина которого совпадает с длиной искомого вектора.

Таким образом, несмотря на то что искусственные нейронные сети Хопфилда успешно демонстрируют ассоциативное восстановление, они не способны реализовывать ассоциативную обработку информации в той мере, какую демонстрируют нервные клетки головного мозга человека.

Описанная задача является типовой и востребована при реализации поиска в реляционных базах данных. Поэтому была поставлена задача разработать нейросетевую архитектуру, способную ассоциативно извлекать и обрабатывать информационные последовательности  $\mathbf{L}_p$ , размер которых значительно превышает число  $n$  нейронов входного слоя, т. е.  $p \gg n$ .

### 1. Способ декомпозиции входных данных и модификация сети ДАП

Рассмотрим искусственную нейронную сеть типа двунаправленной ассоциативной памяти (ДАП), разработанную Б. Коско [14, 15]. Сеть ДАП, как и широко известная сеть Хопфилда, способна к обобщению, вырабатывая правильные реакции даже на искаженные входы. Однако последняя является автоассоциативной. Это означает, что входной образ может быть завершен или исправлен, но не может быть ассоциирован с другим образом. Данный факт является результатом одноуровневой структуры ассоциативной памяти, в которой выходной вектор генерируется на тех же нейронах, на которые поступает входной вектор. В результате восстановленный образ, являющийся точечным аттрактором пространства состояний сети Хопфилда по (1), принципиально не может быть стимулом для генерации новой информации.

В противоположность этому в двухслойной сети ДАП входной вектор поступает на один набор нейронов  $\mathbf{X}^{(n)} = (x_1, x_2, \dots, x_n)^T$ , а соответствующий выходной вектор с помощью матри-

цы весов  $W_{m \times n}$  вырабатывается на другом наборе  $Y^{(m)} = (y_1, y_2, \dots, y_m)^T$ . Таким образом, реализуется *гетероассоциативная память*. Поэтому обученная сеть на поступающий двоичный ключевой вектор, обладающий общими характерными признаками, вырабатывает на выходе один и тот же вектор-отклик, т. е. наблюдается процесс адресации памяти по ее содержанию. Процесс обучения сети ДАП реализуется в форме суммирования внешних произведений всех векторных пар обучающего набора согласно правилу Хебба [3].

Дополнительными преимуществами сетей данного типа являются: реализация распределенной памяти; быстрый безытерационный способ обучения; способность добавлять в память ограниченное число новых образов, не разрушая старые, и относительно простая архитектура для ее программно-аппаратной реализации [13, 16, 17].

Теоретически информационная емкость сети ДАП (количество образов, которые она может запомнить) в два раза меньше, чем у сети Хопфилда. Это связано со структурой сети ДАП, в процессе функционирования которой происходит не только извлечение вектор-отклика  $Y$ , но и восстановление самого вектор-стимула  $X$  (если он в допустимых пределах отличался от эталонного значения). Однако если стимул и отклик представить в виде двух составных частей одной информационной последовательности  $L_p$ , то ее максимальная длина  $p = n + m$ , а в случае, когда  $n = m$ , получаем, что  $p = 2n$ . Другими словами, размер запоминаемого образа в два раза превышает размер входного слоя нейронной сети.

Тем не менее результаты приведенных рассуждений недостаточно удовлетворяют условию поставленной задачи. Поэтому было решено изменить алгоритм работы и архитектуру сети ДАП таким образом, чтобы ее стационарные значения изменялись по формуле (1), в которой  $k \geq 2$ . С этой целью автором было предложено ввести в архитектуру сети ДАП дополнительную обратную связь  $t_B$  (рис. 2), а также новый способ секционирования входных данных.

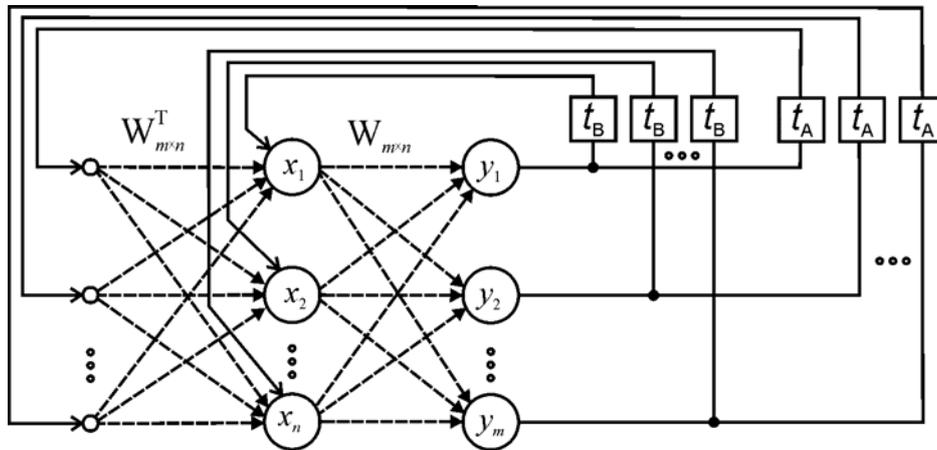


Рис. 2. Модифицированная сеть ДАП

Допустим, имеется бинарный вектор  $X = (x_1, x_2, \dots, x_p)^T$ , в котором содержится некая информационная последовательность  $L_p$ . Далее разделим вектор  $X$  на  $k$  векторов  $x_i$  одинаковой длины таким образом, чтобы выполнялись следующие условия:

$$\begin{cases} x_i \neq x_j, & i \neq j, \quad i, j = \overline{1, k}; \\ p \bmod k = 0. \end{cases} \quad (2)$$

Другими словами, исходный информационный вектор  $X$  представлен как строго определенная последовательность уникальных вектор-столбцов  $x_i$  меньшей, но одинаковой длины  $X = [x_1; x_2; \dots; x_k]$  (рис. 3, а, б). Максимальное число  $k$ , на которое может быть разбит исходный вектор  $X$ , зависит от длины регулярных последовательностей нулей и единиц, встречаю-

щихся в исходном бинарном векторе. Далее запишем (закодируем) каждый из полученных векторов  $\mathbf{x}_i$ , которые здесь и далее будем называть информационными блоками, с помощью сети ДАП таким образом, чтобы его собственным ключом был предшествующий ему вектор  $\mathbf{x}_{i-1}$ . Таким образом, предложенный способ секционирования входных данных заключается в кодировании вектора  $\mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; \dots; \mathbf{x}_k]$  как замкнутой последовательности пар уникальных векторов  $\mathbf{x}_1 \Rightarrow \mathbf{x}_2, \mathbf{x}_2 \Rightarrow \mathbf{x}_3, \dots, \mathbf{x}_{k-1} \Rightarrow \mathbf{x}_k$  и  $\mathbf{x}_k \Rightarrow \mathbf{x}_1$  (рис. 3, в). Другими словами, если входными данными для обучения сети ДАП служит набор  $[\mathbf{x}_1; \mathbf{x}_2; \dots; \mathbf{x}_{k-1}; \mathbf{x}_k]$ , то в качестве выходных данных требуется использовать набор  $[\mathbf{x}_2; \mathbf{x}_3; \dots; \mathbf{x}_k; \mathbf{x}_1]$ , т.е. обучающая выборка будет состоять из набора следующих векторов:

$$\begin{cases} \mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; \dots; \mathbf{x}_{k-1}; \mathbf{x}_k]; \\ \mathbf{Y} = [\mathbf{x}_2; \mathbf{x}_3; \dots; \mathbf{x}_k; \mathbf{x}_1]. \end{cases} \quad (3)$$

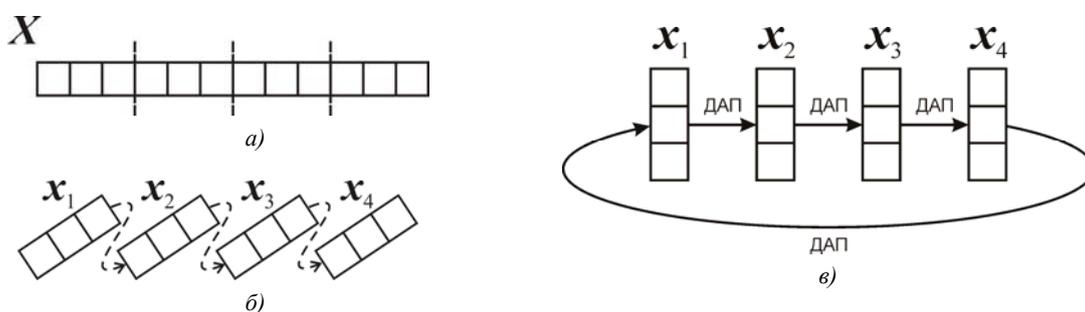


Рис. 3. Запись вектора  $\mathbf{X}$  по предложенному методу: а) и б) подготовка; в) запись

Предложенный алгоритм функционирования сети ДАП отличается от классического тем, что после стабилизации значений выходов нейронов обоих слоев добавляется вторая обратная связь  $t_B$ , которая идет от второго слоя к первому напрямую, не взаимодействуя с весовой квадратной матрицей  $W_{n \times n}$ . В результате предложенный алгоритм позволяет запоминать детерминированную последовательность  $\mathbf{L}_p$ , равную

$$p = k \cdot n \quad (4)$$

элементам, где  $k$  – число информационных блоков;  $n$  – длина входного и выходного слоев ДАП.

Следовательно, если число входных и выходных нейронов в сети ДАП  $m = n = const$ , а число информационных блоков может быть увеличено (если полученные новые подобразы не коррелируют между собой), то  $k$  тоже будет увеличиваться. Другими словами, будет расти объем запоминаемой информации  $\mathbf{L}_p$ . Стоит еще раз уточнить, что  $\mathbf{L}_p$  – это не объем общей памяти сети ДАП, а информационная последовательность нулей и единиц, состоящая из полученного набора подобразов. Под объемом памяти  $V_{\text{ДАП}}$  сети ДАП понимается максимально возможное число запоминаемых образов, которое определяется выражением

$$V_{\text{ДАП}} \leq \frac{l}{4 \cdot \log_2 l}, \quad l = \min(n, m), \quad (5)$$

где  $l$  – количество нейронов в наименьшем слое. В рассматриваемом случае  $l = m = n$ , тогда  $V_{\text{ДАП}}$  равняется объему сети Хопфилда [5].

Введем новую качественную характеристику – информационную эффективность сети  $I$ , с помощью которой можно оценить предложенный гетероассоциативный метод представления

информации. Следуя [18, 19], определим  $I$  как количество бит информации, приходящейся на один синапс, т. е.

$$I = \frac{L}{n^2}, \quad (6)$$

где  $L$  – информация, извлекаемая из сети;  $n^2$  – общее количество элементов сети, способных к модификации (синаптические связи). Вычислим информационную эффективность при записи одного образа, т. е. при фиксированном размере матрицы весов  $W_{n \times n}$ , с помощью сети Хопфилда:  $I_{\text{Хопф}} = \frac{L}{n^2} = \frac{n}{n^2} = \frac{1}{n}$  и с помощью модифицированной ДАП:  $I_{\text{ДАП}} = \frac{p}{n^2} = \frac{k \cdot n}{n^2} = \frac{k}{n}$ , где  $n$  – число нейронов во входном слое;  $k$  – число подобразов, на которые разбивается исследуемый образ. Таким образом, если ставится задача сохранить один образ с помощью ассоциативной нейронной сети, то  $I_{\text{ДАП}} = k \cdot I_{\text{Хопф}}$ .

В результате использования предложенных способ секционирования входных данных и модифицированного алгоритма функционирования сети ДАП последняя способна реализовывать отображение специального вида:  $\mathbf{x}_{i+1} = F(\mathbf{x}_i)$ , причем  $\mathbf{x}_1 = F(\mathbf{x}_k)$ , где  $k$  – количество информационных блоков, на которые был разбит первоначальный вектор  $\mathbf{X}$ . После проведенного процесса обучения сеть ДАП способна восстанавливать и идентифицировать запомненный на этапе обучения вектор  $\mathbf{X}$  по любому известному ключевому вектору  $\mathbf{x}_i$ . Поэтому процессы записи и извлечения информации, закодированной в виде последовательности уникальных векторов  $\mathbf{x}_i$ , являются возможными только в процессе динамического режима работы сети.

Так как итерационный процесс извлечения информации является циклическим, значения выходов описанной системы, попав на аттрактор, будут бесконечно изменяться согласно формуле (1). Поэтому предложенный метод регламентирует и правило останова. Процесс распознавания (извлечения) считается успешно завершенным, если за конечное число итераций на выходе сети ДАП появится вектор, идентичный ключевому, который послужил стимулом к возникновению данного динамического режима. Если описанное правило не выполняется, т. е. значения выходов сети ДАП находятся в хаотическом режиме, то считается, что искомый образ не был записан в память либо ключевой вектор не обладает достаточным количеством начальной информации.

## 2. Извлечение и идентификация данных

С помощью модифицированной сети ДАП, а также предложенных способа секционирования и метода записи входных данных появляется возможность реализовать устройства, способные выполнять одну из следующих функций: извлечение либо идентификацию данных, представленных информационной последовательностью  $\mathbf{L}_p$ . Основой подобных устройств, к числу которых относятся устройства извлечения и идентификации данных, может служить не только описанная выше модифицированная сеть ДАП, но и любая другая реализация гетероассоциативной памяти.

На рис. 4, а изображена блок-схема алгоритма работы устройства, предназначенного для ассоциативного извлечения (восстановления) по любой известной части заранее сохраненной информационной последовательности  $\mathbf{L}_p$ , в качестве которой, например, могут выступать как графические изображения, так и текстовые файлы, предварительно закодированные в бинарный вид. Одним из главных требований для записи указанной информации  $\mathbf{L}_p$  является описанный выше этап предобработки, цель которого заключается в ее декомпозиции на некоторое число информационных блоков, соответствующих условию (2). Полученные информационные блоки должны быть записаны в модифицированную сеть ДАП по предложенному методу (3).

Процесс извлечения информации  $\mathbf{L}_p$  должен начинаться с ввода любого из известных информационных блоков, представленного вектором  $\mathbf{x}_{\text{нач}}$ . Затем значения данного вектора

присваиваются временному вектору  $x$ , который записывается в результирующий массив  $X$  и с помощью сети ДАП извлекает первый вектор-отклик  $y$ . На следующей итерации значения полученного вектор-отклика  $y$  присваиваются значениям нового вектор-стимула  $x$ , который извлекает из сети ДАП очередной информационный блок. Таким образом, пока выполняется условие  $x_{нач} \neq y$ , из сети ДАП последовательно извлекаются информационные блоки, принадлежащие искомой информационной последовательности  $L_p$ . После того как будет извлечен ключевой вектор  $x_{нач}$ , следует этап проверки на адекватность работы устройства. В данном случае используется проверка на наличие числа  $k$  информационных блоков. Для этого была введена дополнительная переменная-счетчик  $c$ . В качестве альтернативы можно предложить, чтобы  $k$ -й информационный блок содержал некую наперед заданную последовательность символов, являющуюся признаком конца информационной последовательности  $L_p$ .

На рис. 4, б изображена блок-схема алгоритма работы устройства, предназначенного для регистрации в потоке сенсорных данных определенного класса образов, которые могут описывать критические значения некоторых физических процессов. Подобными сенсорными данными могут быть графические изображения, акустические сигналы, данные химических анализаторов и т. д. Таким образом, задачей устройства является причисление входного вектора к одному из двух классов (например, штатный и нештатный режимы), причем близость либо корреляция между образами каждого из классов может полностью отсутствовать, т. е. сигналы могут быть различной природы. Главное, чтобы их информационные блоки отвечали условиям (2).

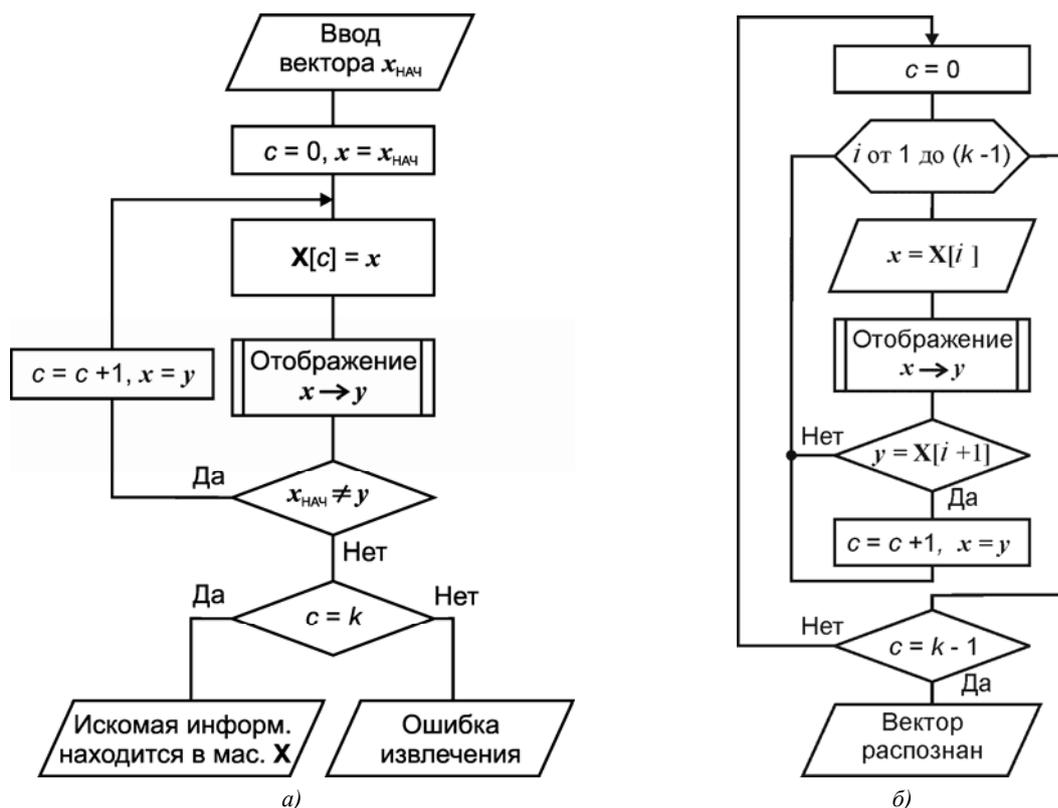


Рис. 4. Блок-схемы алгоритмов функционирования предлагаемых устройств

Принцип работы предлагаемого устройства идентификации сенсорных данных заключается в подсчете количества ассоциируемых пар информационных блоков исследуемого вектора  $X$ . Если информационные блоки принадлежат к известному образу, то, начиная с первого, они будут последовательно отображаться друг в друга и счетчик  $c$  будет равен  $(k - 1)$ . В данном случае считается, что исследуемый вектор  $X$  соответствует одному из известных эталонов, при этом устройство выдаст сигнал о его распознавании.

### 3. Примеры и эксперименты по записи и извлечению информации

На типовых примерах записи и извлечения информации рассмотрим преимущества модифицированной сети ДАП. Для начала проанализируем пример записи и восстановления бинарного вектора с помощью предложенного модифицированного способа. Допустим, у нас имеется 16-битный вектор-столбец  $\mathbf{X} = (1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1 \ 0)^T$ , т. е. его длина  $p = 16$ . Максимально допустимым числом частей  $k$ , на которое его можно разделить согласно условию (3), является четыре. Это объясняется тем, что исходный вектор  $\mathbf{X}$  нельзя разделить на 3-битные векторы без остатка, а при разделении его на 2-битные среди них окажутся одинаковые. Таким образом, получим следующие четыре вектора:  $\mathbf{x}_1 = (1 \ 1 \ 0 \ 0)^T$ ,  $\mathbf{x}_2 = (1 \ 1 \ 1 \ 1)^T$ ,  $\mathbf{x}_3 = (1 \ 0 \ 0 \ 1)^T$  и  $\mathbf{x}_4 = (1 \ 0 \ 1 \ 0)^T$ .

Чтобы записать полученную последовательность  $\mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; \mathbf{x}_3; \mathbf{x}_4]$  с помощью предложенного метода, реализуем двухслойную сеть ДАП, в каждом слое которой будет находиться по четыре нейрона. Воспользуемся обобщенной формулой генерации матрицы синаптических связей  $W_{m \times n}$  по следующей формуле [5–7]:

$$W_{m \times n} = \sum_{i=1}^k (2 \cdot \mathbf{Y}_i^{(m)} - 1) \otimes (2 \cdot \mathbf{X}_i^{(n)} - 1)^T, \quad (7)$$

где  $k$  – число пар обучаемой выборки  $\{\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_k\}$  и  $\{\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_k\}$ ;  $n$  и  $m$  – число входных и выходных нейронов соответственно; операция в скобках необходима для перевода бинарных векторов в биполярные.

Учитывая, что  $m = n = 4$ ,  $k = 4$ ,  $\mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; \mathbf{x}_3; \mathbf{x}_4]$  и  $\mathbf{Y} = [\mathbf{y}_1; \mathbf{y}_2; \mathbf{y}_3; \mathbf{y}_4]$ , получим

$$\begin{aligned} W_{4 \times 4} &= \left( \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \otimes [1 \ 1 \ -1 \ -1] + \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \otimes [1 \ 1 \ 1 \ 1] + \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix} \otimes [1 \ -1 \ -1 \ 1] + \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} \otimes [1 \ -1 \ 1 \ -1] \right) = \\ &= \left( \begin{bmatrix} 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 \end{bmatrix} + \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 \end{bmatrix} + \begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \end{bmatrix} + \begin{bmatrix} 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ -1 & 1 & -1 & 1 \\ -1 & 1 & -1 & 1 \end{bmatrix} \right) = \\ &= \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & -4 \\ 0 & 0 & -4 & 0 \\ 0 & 4 & 0 & 0 \end{bmatrix}. \end{aligned}$$

Таким образом, в значениях полученной весовой матрицы  $W_{4 \times 4}$  синаптических связей нейронов содержится информация об искомом векторе  $\mathbf{X}$ . Далее, учитывая функцию генерации вектор-отклика сети ДАП [5–7]

$$\mathbf{Y}^{(m)} = f(W_{m \times n} \cdot \mathbf{X}^{(n)} - \mathbf{B}^{(m)}), \quad (8)$$

где  $f(x)$  – функция активации (например, знаковая функция сигнум);  $\mathbf{B}^{(m)}$  – вектор смещения (как правило, равный нулю), восстановим всю последовательность, начиная с ввода третьего участка  $\mathbf{x}_3 = (1 \ 0 \ 0 \ 1)^T$ :

$$\mathbf{x}'_4 = \operatorname{sgn} \left( \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & -4 \\ 0 & 0 & -4 & 0 \\ 0 & 4 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \right) = \operatorname{sgn} \left( \begin{bmatrix} 4 \\ 0 \\ 4 \\ 0 \end{bmatrix} \right) = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \mathbf{x}_4;$$

$$\mathbf{x}'_1 = \operatorname{sgn} \left( \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & -4 \\ 0 & 0 & -4 & 0 \\ 0 & 4 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \right) = \operatorname{sgn} \left( \begin{bmatrix} 4 \\ 4 \\ 0 \\ 0 \end{bmatrix} \right) = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \mathbf{x}_1;$$

$$\mathbf{x}'_2 = \operatorname{sgn} \left( \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & -4 \\ 0 & 0 & -4 & 0 \\ 0 & 4 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \right) = \operatorname{sgn} \left( \begin{bmatrix} 4 \\ 4 \\ 4 \\ 4 \end{bmatrix} \right) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \mathbf{x}_2,$$

и, наконец, проверим ключевой образ

$$\mathbf{x}'_3 = \operatorname{sgn} \left( \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & -4 \\ 0 & 0 & -4 & 0 \\ 0 & 4 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right) = \operatorname{sgn} \left( \begin{bmatrix} 4 \\ 0 \\ 0 \\ 4 \end{bmatrix} \right) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \mathbf{x}_3.$$

В результате восстановлена вся искомая последовательность  $\mathbf{X}=[\mathbf{x}_1; \mathbf{x}_2; \mathbf{x}_3; \mathbf{x}_4]$ , причем восстановление можно было начинать с любой известной части.

Рассмотрим второй пример, в котором требуется разработать устройство ассоциативного хранения несжатой графической информации – монохромных изображений. При этом существуют аппаратные ограничения на число параллельных каналов ввода-вывода и объем используемой памяти – 8 Мб. Таким образом, если на один элемент весовой матрицы  $W$  выделить по 8 бит, то максимальный ее размер составит  $1024 \times 1024$  элементов.

С помощью современных аппаратных средств можно реализовать сеть Хопфилда, которая будет в состоянии запоминать монохромные изображения размером  $32 \times 32$  (рис. 5, а). Точно такого же размера можно реализовать сеть ДАП, которая тоже будет способна запоминать и восстанавливать монохромные изображения размером  $32 \times 32$ . Однако благодаря предложенному методу гетероассоциативной записи и извлечения последовательности образов изображения, показанные на рис. 5, б–д, можно закодировать как подобразы одного большого образа. В результате обученная по предложенному методу сеть ДАП способна при вводе одного из известных подобразов 5, б–д восстановить искомый образ, размеры которого будут уже  $64 \times 64$  (рис. 5, е). Таким образом, в приведенном примере сеть ДАП способна хранить и восстанавливать (при наличии любой из четырех составных частей) монохромное изображение 5, е, которое в четыре раза превышает по размеру изображение 5, а.

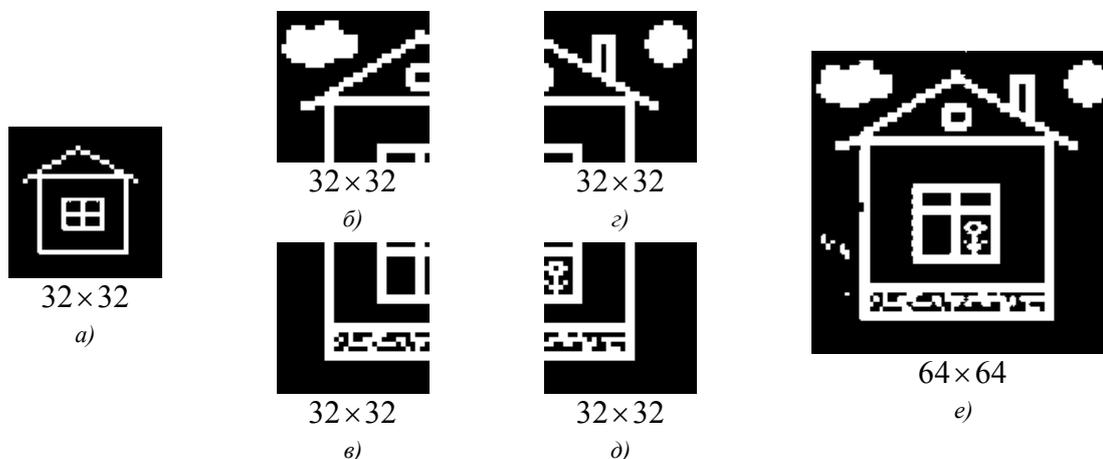


Рис. 5. Восстановление графических изображений сетями Хопфилда и модифицированной ДАП

На следующем примере рассмотрим обратную задачу. Имеется монохромное изображение размером  $120 \times 160$ , требуется определить минимальный размер весовой матрицы  $W_{n \times n}$ , способной к восстановлению указанного изображения с помощью предложенного метода по любой его известной части. Для этого необходимо провести ряд экспериментов, заключающихся в делении монохромного изображения (рис. 6, а) на  $k$  одинаковых по размеру участков. Ниже приведены 15 возможных способов деления, причем на рис. 6, б указан седьмой способ, т. е.  $2 \times 4$ :

- |                       |                       |                        |                         |                         |
|-----------------------|-----------------------|------------------------|-------------------------|-------------------------|
| 1) $1 \times 1 = 1$ ; | 4) $2 \times 2 = 4$ ; | 7) $2 \times 4 = 8$ ;  | 10) $3 \times 5 = 15$ ; | 13) $6 \times 4 = 24$ ; |
| 2) $1 \times 2 = 2$ ; | 5) $1 \times 5 = 5$ ; | 8) $2 \times 5 = 10$ ; | 11) $4 \times 4 = 16$ ; | 14) $5 \times 5 = 25$ ; |
| 3) $3 \times 1 = 3$ ; | 6) $3 \times 2 = 6$ ; | 9) $3 \times 4 = 12$ ; | 12) $4 \times 5 = 20$ ; | 15) $6 \times 5 = 30$ . |

Затем полученные подобразы записываются по предложенному методу в сеть ДАП.

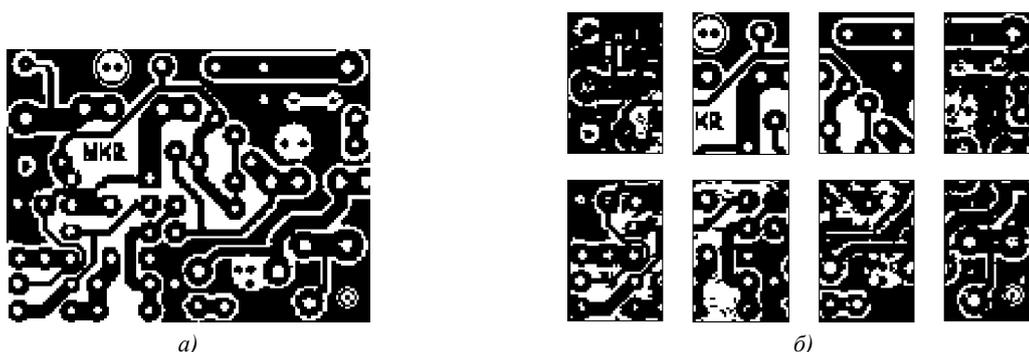


Рис. 6. Пример записи и распознавания монохромного изображения

На графике (рис. 7) отчетливо видна зависимость качества распознавания отдельных подобразов от длины  $n$  входного вектора сети ДАП ( $n = p/k$ ,  $p = const$  – длина информационной последовательности). Из формулы (5) видно, что от числа  $n$  зависит максимально возможное число подобразов. Сплошной линией на графике указана относительная погрешность распознавания  $\delta_k$  при различных способах разбиения исходного изображения:

$$\delta_k = \frac{1}{n} \sum_{i=1}^k h(x_i, \tilde{x}_i) \cdot 100\%, \quad (9)$$

где  $h(x_i, \tilde{x}_i)$  – расстояние Хемминга между эталонными  $x_i$  подобразами из обучающей выборки и соответствующими подобразами  $\tilde{x}_i$ , полученными в результате распознавания. Деление на  $n$  необходимо для устранения влияния размера входного слоя на значение погрешности.

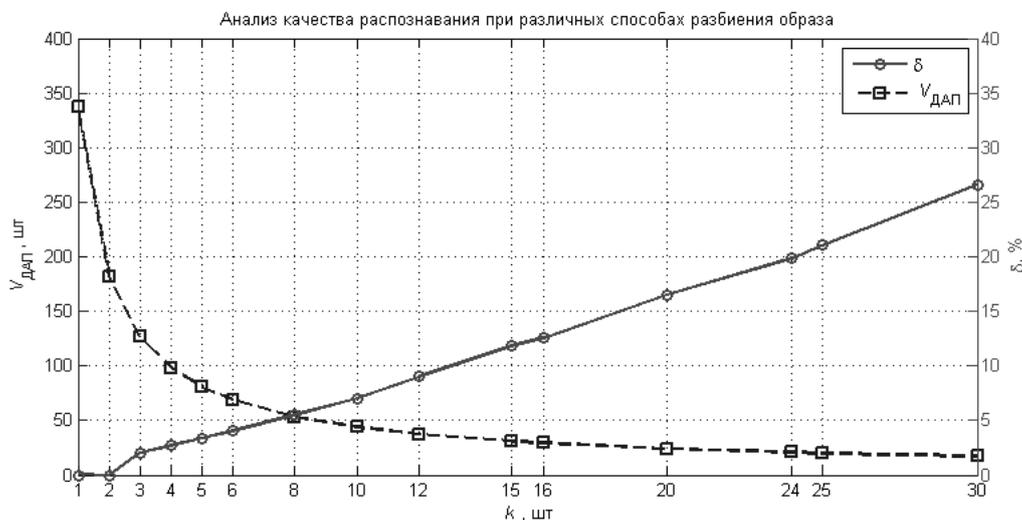


Рис. 7. Экспериментальные данные различных способов записи монохромного изображения

Эксперименты на конкретном изображении (см. рис. 6, *a*) показали, что из-за сильной корреляции между подобразами с ростом числа  $k$  пропорционально растет и относительная погрешность набора подобразов  $\delta_k$ . Это в конечном счете приводит к разрушению первоначального образа, т. е. образ полностью не восстанавливается. На рис. 6, *b* изображен пример, когда исходное изображение было поделено на восемь одинаковых по размеру частей и было восстановлено с ошибкой в 5%. Хотя, судя по графику, при  $k=5$  максимально возможным числом подобразов, которое можно записать в ДАП, является 50.

Благодаря своей нейросетевой архитектуре предложенная модель памяти способна добавлять новые и удалять старые образы. Это свойство достигается благодаря безытерационному способу обучения ДАП [16, 17]. Например, из памяти обученной сети ДАП с весовой матрицей  $W_{n \times n}$  требуется удалить информационный блок  $\mathbf{x}_i$ , тогда необходимо разорвать ассоциативную связь между последовательными блоками  $\mathbf{x}_{i-1} \Rightarrow \mathbf{x}_i$  и  $\mathbf{x}_i \Rightarrow \mathbf{x}_{i+1}$  и создать новую ассоциативную пару  $\mathbf{x}_{i-1} \Rightarrow \mathbf{x}_{i+1}$ :

$$W'_{n \times n} = W_{n \times n} - \mathbf{x}_i \otimes \mathbf{x}_{i-1}^T - \mathbf{x}_{i+1} \otimes \mathbf{x}_i^T + \mathbf{x}_{i+1} \otimes \mathbf{x}_{i-1}^T. \quad (10)$$

Для добавления в сеть ДАП после  $i$ -го участка нового информационного блока  $\mathbf{x}_j$  необходимо воспользоваться следующей формулой:

$$W''_{n \times n} = W_{n \times n} - \mathbf{x}_{i+1} \otimes \mathbf{x}_i^T + \mathbf{x}_j \otimes \mathbf{x}_i^T + \mathbf{x}_{i+1} \otimes \mathbf{x}_j^T. \quad (11)$$

В результате для выполнения операций над определенным информационным блоком в предложенном методе организации памяти не требуется искать его расположение среди других блоков (представленных в виде списка), необходимо только создать новую или удалить старую ассоциативную связь.

### Заключение

Предложена модификация нейронной сети ДАП, а также способ секционирования и записи входных данных, позволяющие не только ассоциативно идентифицировать (распознавать) ранее запомненные образы, но и извлекать (восстанавливать) из памяти определенные последовательности соответствующих ассоциативных пар. Описываемая итерационная модель нейросетевой гетероассоциативной памяти способна эффективно оперировать большими объемами информации. Так, количество элементов информационной последовательности  $L_p$ , состоящей из нулей и единиц, при фиксированном размере весовой матрицы синаптических связей  $W_{n \times n}$  может достигать длины  $p = k \cdot n$ , где  $n$  – размер входного слоя нейронной сети ДАП, а  $k$  – количество уникальных информационных блоков запоминаемого образа.

Среди основных положительных свойств полученной модели памяти можно выделить реализацию поиска информации по ее содержанию и хранение информации в виде структурированных последовательностей. Также можно отметить, что процесс выборки информации представляет собой динамический процесс, подобный процессам, протекающим в различных физических системах непрерывного действия. Недостатком предложенной модели памяти является ее небольшая емкость, связанная с ограниченной способностью сети ДАП.

Благодаря итерационному методу записи и извлечения, а также распределенному методу хранения информации на множестве взаимодействующих нейроподобных элементов рассматриваемая модель обладает сложной нелинейной динамикой, свойства которой еще предстоит изучить.

### Список литературы

1. Кузнецов, С.Д. Основы баз данных : учебное пособие / С.Д. Кузнецов. – 2-е изд., испр. – М. : Интернет-университет информационных технологий; БИНОМ. Лаборатория знаний, 2007. – 484 с.

2. Кохонен, Т. Ассоциативные запоминающие устройства / Т. Кохонен. – М. : Мир, 1982. – 384 с.
3. Hopfield, J. Neural networks and physical systems with emergent collective computational abilities / J. Hopfield // Proc. of National Academy of Sciences. – 1982. – Vol. 79, № 8. – P. 2554–2558.
4. Hopfield, J.J. Computing with neural circuits / J.J. Hopfield, D.W. Tank // A model. Science. – 1986. – Vol. 233. – P. 625–633.
5. The capacity of the Hopfield associative memory / R.J. McEliece [et al.] // IEEE Transactions on Information Theory. – 1987. – Vol. 33, № 4. – P. 461–482.
6. Elman, J.L. Finding structure in time / J.L. Elman // Cognitive Science. – 1990. – Vol. 14. – P. 179–211.
7. Narendra, K.S. Identification and control of dynamical systems using neural networks / K.S. Narendra, K. Parthasarathy // IEEE Trans. Neural Networks. – 1990. – Vol. 1. – P. 4–27.
8. Improved Elman networks and applications for controlling ultrasonic motors / X.H. Shi [et al.] // Applied Artificial Intelligence: An International Journal. – 2004. – Iss. 7, vol. 18. – P. 603–629.
9. Хайкин, С. Нейронные сети: полный курс / С. Хайкин. – 2-е изд., пер. с англ. – М. : Вильямс, 2006. – 1104 с.
10. Фролов, А.А. Нейронные модели ассоциативной памяти / А.А. Фролов, И.П. Муравьев. – М. : Наука, 1987. – 160 с.
11. Ежов, А.А. Нейрокомпьютинг и его применения в экономике и бизнесе / А.А. Ежов, С.А. Шумский. – М. : МИФИ, 1998. – 223 с.
12. Меламед, И.И. Нейронные сети и комбинаторная оптимизация / И.И. Меламед // Автоматика и телемеханика. – 1994. – № 11. – С. 3–40.
13. Уоссермен, Ф. Нейрокомпьютерная техника: теория и практика / Ф. Уоссермен. – М. : Мир, 1990. – 240 с.
14. Kosko, B. Constructing an associative memory / B. Kosko // Byte. – 1987. – Vol. 12. – P. 137–144.
15. Kosko, B. Bi-directional associative memories / B. Kosko // IEEE Transactions on Systems, Man and Cybernetics. – 1988. – Vol. 18, № 1. – P. 49–60.
16. Прокопович, Г.А. Адаптивный нейросетевой классификатор / Г.А. Прокопович // Информатика. – 2009. – № 23. – С. 68–81.
17. Прокопович, Г.А. Нейросетевой блок памяти для адаптивной работы сложных технических систем в динамической среде / Г.А. Прокопович // Информатика. – 2010. – № 26. – С. 54–65.
18. Palm, G. On the information storage capacity of local learning rules / G. Palm // Neural Comp. – 1992. – Vol. 4. – P. 703–711.
19. Sommer, F.T. Improved bidirectional retrieval of sparse patterns stored by Hebbian learning / F.T. Sommer, G. Palm // Neural Networks. – 1999. – Vol. 12 (2). – P. 281–297.

**Поступила 17.02.12**

*Объединенный институт проблем  
информатики НАН Беларуси,  
Минск, Сурганова, 6  
e-mail: bigznich@tut.by*

**R.A. Prakaovich**

**HETEROASSOCIATIVE NEURAL NETWORKS APPLICATION  
FOR DATA RECORDING AND RECOVERY**

The problem of an associative representation of the data is considered on an example of recurrent neural networks of Hopfield type. An original heteroassociative input data representing method is proposed, which is based on data splitting into equal parts, followed by sequential mapping of these parts into each other. The obtained neural network architecture is able for recording and associative recovery of large volumes of information.

УДК 004.383.3

**М.И. Вашкевич, И.С. Азаров, А.А. Петровский****ПОДАВЛЕНИЕ ЭФФЕКТА АКУСТИЧЕСКОЙ ОБРАТНОЙ СВЯЗИ  
В СЛУХОВЫХ АППАРАТАХ С ИСПОЛЬЗОВАНИЕМ  
НЕРАВНОПОЛОСНОГО БАНКА ФИЛЬТРОВ**

*Предлагается метод подавления эффекта акустической обратной связи, возникающего в слуховых аппаратах. Метод основан на применении адаптивной фильтрации в частотных полосах сигнала. Отличительной особенностью предлагаемого метода является то, что при его использовании не возникает дополнительной задержки в прямом пути распространения сигнала. За счет выполнения декомпозиции сигнала на неравнополосные частотные составляющие метод обеспечивает более высокую скорость сходимости и более высокий уровень ослабления сигнала обратной связи по сравнению с широкополосной адаптивной фильтрацией.*

**Введение**

Цифровые слуховые аппараты позволяют повысить разборчивость речи для людей с нарушением слуха за счет усиления полезного сигнала. Однако утечка сигнала из динамика слухового аппарата в микрофон, называемая акустической обратной связью (АОС), значительно ограничивает максимальный коэффициент усиления, который может быть использован в слуховом аппарате, при условии сохранения его устойчивости. АОС вызывает эффект завывания и значительно ухудшает качество звука, когда коэффициент усиления близок к максимальному. В ряде случаев максимального усиления недостаточно, чтобы компенсировать потерю слуха.

Для уменьшения влияния АОС предложено несколько методов, использующих адаптивные алгоритмы. Наибольшее распространение в практических реализациях слуховых аппаратов получили адаптивные алгоритмы на основе метода наименьших средних квадратов благодаря их простоте и малой вычислительной сложности [1]. Тем не менее эти алгоритмы не всегда позволяют обеспечить достаточную устойчивость для широкополосного сигнала. Эффективным способом устранения данного недостатка является использование в системе подавления эффекта АОС банка фильтров. При помощи банка фильтров сигнал может быть расщеплен на субполосные составляющие, и процесс адаптации может выполняться отдельно в каждой субполосе. За счет этого достигается большая устойчивость системы, а также увеличивается скорость сходимости адаптивного алгоритма. Дополнительным преимуществом такого подхода является возможность уменьшения порядка адаптивных фильтров и сокращение вычислительной сложности за счет децимации субполосных сигналов.

В работе [2] предлагается выполнять субполосную декомпозицию сигнала с помощью банка фильтров WOLA (weighed overlap-add) для улучшения производительности системы подавления эффекта АОС (рис. 1). На схеме индексы  $n$  и  $m$  используются для обозначения временных отсчетов исходного (широкополосного) сигнала и отсчетов субполосных сигналов соответственно. Для обозначения векторных величин на схеме используются жирные линии. Изменяющаяся во времени функция  $\mathbf{G}(m)$  решает задачи компенсации потери слуха, подавления шума и компрессии динамического диапазона. Коэффициент  $g_c$  применяется для общего усиления выходного уровня звука, он может изменяться пользователем слухового аппарата посредством регулятора громкости.

Недостатком данного подхода является то, что банк фильтров вносит ощутимую задержку (порядка 16 мс) в прямой путь распространения сигнала между микрофоном и динамиком. Это нежелательно по нескольким причинам. Во-первых, при задержке более 10 мс возникает эффект гребенчатого фильтра (comb filter effect) [3], который воспринимается человеческим ухом как эхо и является результатом того, что звук, обработанный слуховым аппаратом, накладывается в

среднем ухе на необработанный звук. Во-вторых, собственный голос говорящего достигает улитки внутреннего уха посредством костной звукопроводимости и через отверстие в слуховом аппарате с минимальной задержкой, где взаимодействует с задержанным и усиленным сигналом от слухового аппарата, что вызывает эффект окрашивания звука (coloration effect) [4].

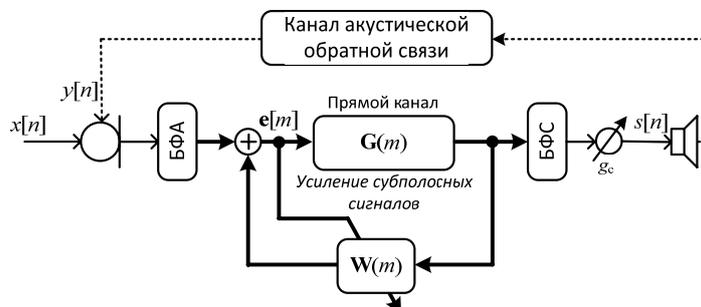


Рис. 1. Схема подавления АОС с использованием банка фильтров (БФА – банк фильтров анализа, БФС – банк фильтров синтеза)

Отсутствие дополнительной задержки в прямом пути можно обеспечить за счет параллельного включения системы подавления эффекта АОС (рис. 2). Однако данная система не позволяет выполнять адаптивную фильтрацию в субполосах сигнала  $s[n]$ , поскольку в этом случае в сигнал компенсации  $\hat{y}[n]$  будет внесена задержка, обусловленная банками анализа/синтеза.

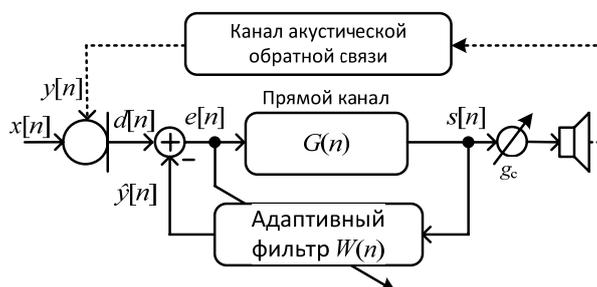


Рис. 2. Система подавления АОС, не вносящая задержки в прямой путь распространения сигнала

В настоящей работе предлагается метод субполосного подавления эффекта АОС, который не вносит дополнительной задержки в прямой путь распространения сигнала в слуховом аппарате. Работа опирается на последние результаты в области адаптивной фильтрации, которые показывают, что неравнополосные адаптивные структуры в ряде случаев превосходят равнополосные по таким параметрам, как скорость сходимости и/или модельная ошибка, благодаря их повышенной гибкости [5]. В отличие от [2] для субполосной декомпозиции сигнала в работе используется передискретизированный неравнополосный косинусно-модулированный банк фильтров (КМБФ) [6].

## 1. Субполосное подавление эффекта акустической обратной связи

### 1.1. Устранение задержки банков фильтров анализа/синтеза

Основная идея предлагаемого подхода заключается в том, чтобы устранить задержку, вносимую в компенсирующий сигнал  $\hat{y}[n]$  банками фильтров, за счет добавления в адаптивный фильтр функции предсказания. Данный подход оправдан, поскольку эффект АОС приводит к появлению периодической помехи на определенной частоте, а, как известно, детерминированные (периодические) сигналы предсказываются с высокой точностью.

Рассмотрим алгоритм адаптивной фильтрации на основе метода наименьших средних квадратов. Пусть  $S(z)$  – это  $z$ -образ входного сигнала, а  $D(z)$  –  $z$ -образ обучающего сигнала, появляющегося на выходе линейной системы с передаточной функцией  $W(z)$ , тогда в результате работы адаптивного алгоритма находится оценка передаточной функции  $\hat{W}(z)$ . Если задержать входной сигнал на  $r$  отсчетов, то оцениваемая система будет выполнять функцию линейного предсказания на  $r$  отсчетов, а оценка передаточной функции примет вид  $z^r \hat{W}(z)$ . Допустим, что банк фильтров синтеза вносит задержку в компенсирующий сигнал, равную  $r$  отсчетам, тогда входной и обучающий сигналы можно синхронизировать, используя фильтр с передаточной функцией  $z^r \hat{W}(z)$  (рис. 3).

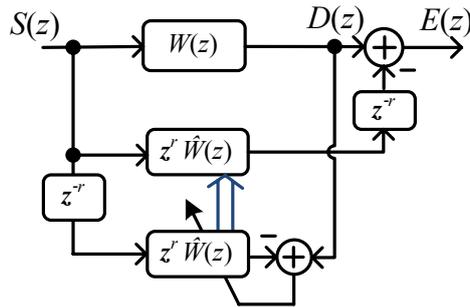


Рис. 3. Устранение задержки при помощи адаптивной фильтрации (широкой стрелкой показана операция копирования коэффициентов адаптивного фильтра)

Используя изложенный выше принцип устранения задержки, предложим следующую схему субполосного подавления эффекта АОС (рис. 4).

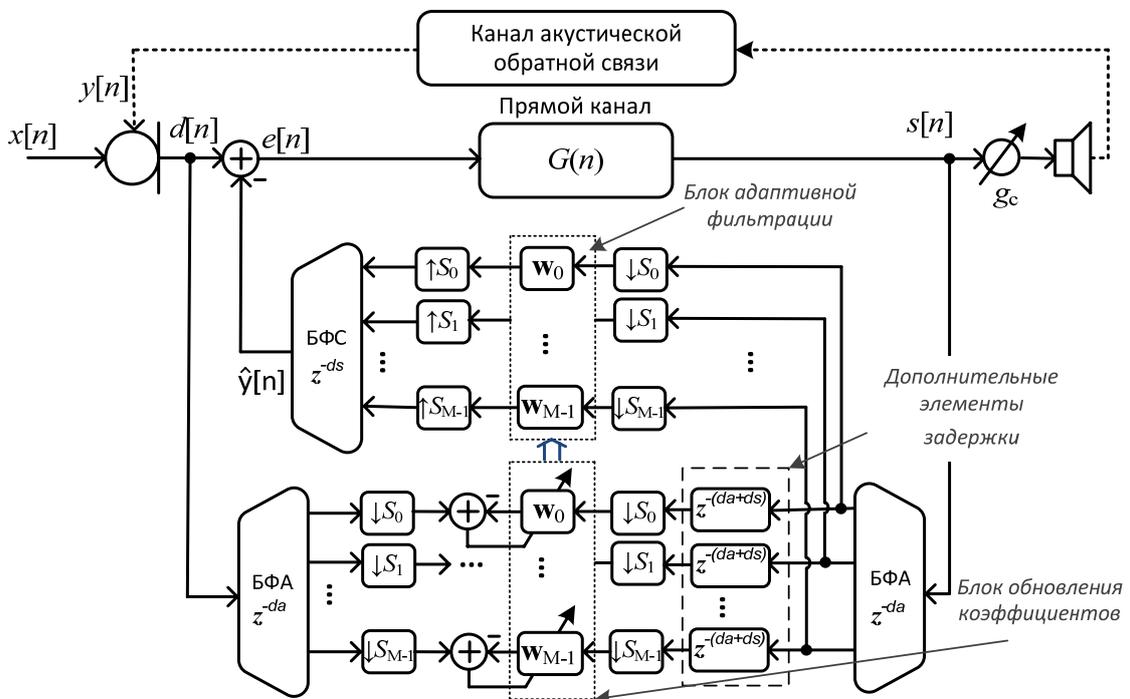


Рис. 4. Схема субполосного подавления акустической обратной связи на основе неравнополосного КМБФ

При помощи банков фильтров анализа выполняется расщепление сигналов  $d[n]$  и  $s[n]$  на  $M$  спектральных компонент. Поскольку спектры сигналов внутри каналов занимают более узкие полосы частот, выполняется переход на более низкую частоту дискретизации. В банке фильтров синтеза исходная частота дискретизации восстанавливается. В каждом канале банка фильтров оценивается свой вектор коэффициентов адаптивного фильтра (на схеме соответствующие векторы обозначены  $\mathbf{w}_0, \dots, \mathbf{w}_{M-1}$ ). Банки фильтров вносят следующие задержки:  $da$  – задержка в банке фильтров анализа;  $ds$  – задержка в банке фильтров синтеза. С помощью изложенного выше принципа для синхронизации желаемого сигнала  $d[n]$  с входным  $s[n]$  применяются дополнительные элементы задержки.

### 1.2. Адаптивная фильтрация субполосных сигналов

Особенность алгоритма адаптивной фильтрации, используемого в системе подавления эффекта АОС, заключается в том, что обновление коэффициентов фильтров и сама фильтрация происходят раздельно: в блоке обновления коэффициентов и блоке адаптивной фильтрации. В каждом канале оценивается свой набор коэффициентов адаптивного фильтра. Процедура оценки одинакова для всех каналов и отличается лишь значениями параметров, таких как порядок фильтра, коэффициент потерь и шаг адаптации. В блоке обновления коэффициентов используется стандартный алгоритм наименьших квадратов (для упрощения записи индекс номера канала опущен):

1. Каждому коэффициенту фильтра  $w[l]$ ,  $l = 0, 1, \dots, L-1$ , присваивается нулевое значение,  $L$  – порядок адаптивного фильтра.

2. Вычисляется выходной отсчет фильтра:  $\hat{y}[m] = \sum_{l=0}^{L-1} w[l]s[m-l]$ , где  $m$  – номер текущего отсчета входного сигнала,  $s[m]$  – входной сигнал.

3. Вычисляется оценка ошибки:  $e[m] = d[m] - \hat{y}[m]$ , где  $d[m]$  – желаемый сигнал.

4. Обновляются весовые коэффициенты:  $w[l] = \delta w[l] + 2\mu e[m]x[m-l]$ , где  $0 < \delta < 1$  – коэффициент потерь. Параметр  $\mu$  представляет собой шаг адаптации алгоритма. Увеличивается номер текущего отсчета:  $m = m + 1$ . Алгоритм переходит к шагу 2.

В блоке адаптивной фильтрации выполняется только второй шаг алгоритма, а коэффициенты  $w[l]$  копируются из блока обновления коэффициентов.

На рис. 5 показаны сигнал на входе слухового аппарата и выходные сигналы широкополосной и субполосной систем подавления эффекта АОС. Из приведенных графиков можно сделать вывод, что субполосная система имеет более высокую скорость сходимости и нежелательные осцилляции в выходном сигнале значительно меньше, чем в случае использования широкополосной системы подавления эффекта АОС.

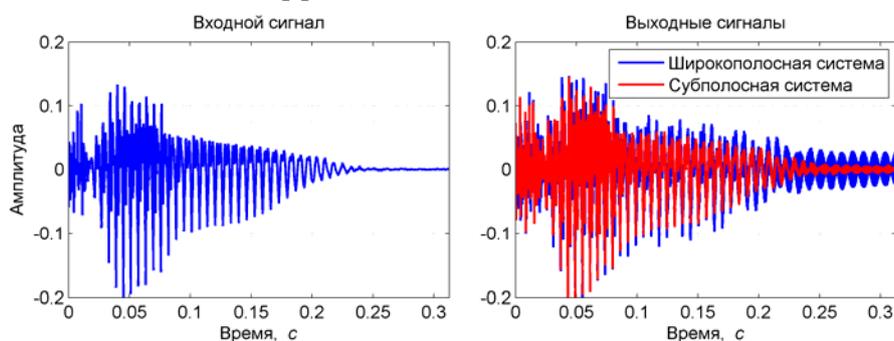


Рис. 5. Качественное сравнение широкополосной и субполосной систем подавления эффекта АОС

### 1.3. Неравнополосный косинусно-модулированный банк фильтров

В системе подавления эффекта АОС используется неравнополосный банк фильтров, ширина полос которого адаптирована к разрешающей способности слухового восприятия. Исходя из того, что слуховая система человека более чувствительна к изменениям сигнала в области

нижних частот, банк фильтров должен обладать следующим свойством: ширина полос пропускания должна возрастать с увеличением частоты.

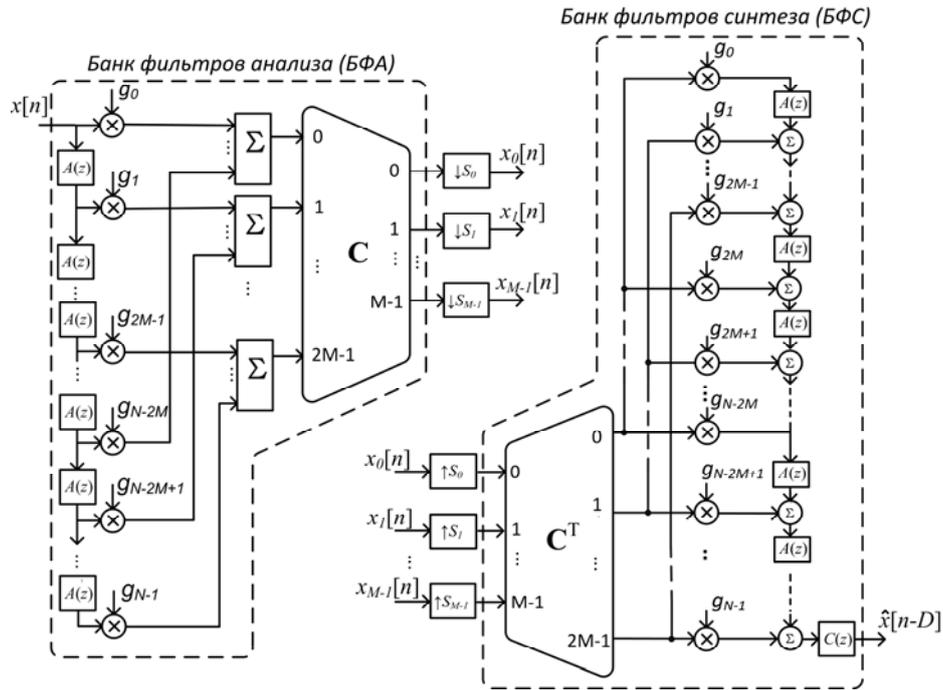


Рис. 6. Полифазная реализация неравнополосного КМБФ

В предложенной системе используется неравнополосный КМБФ [6], поскольку он обладает необходимой конфигурацией полос (рис. 7, а). Основой данного банка служит равнополосный КМБФ, импульсные характеристики фильтров анализа ( $h_k[n]$ ) и синтеза ( $f_k[n]$ ) которого получаются за счет косинусной модуляции фильтра-прототипа:

$$h_k[n] = 2h[n] \cos\left(\frac{\pi(k+1/2)}{M} \left(n - \frac{N-1}{2}\right) + (-1)^k \frac{\pi}{4}\right);$$

$$f_k[n] = 2h[n] \cos\left(\frac{\pi(k+1/2)}{M} \left(n - \frac{N-1}{2}\right) - (-1)^k \frac{\pi}{4}\right),$$

где  $N$  – порядок фильтра-прототипа;  $k = 0, \dots, M-1$  – номер канала;  $n = 0, \dots, N-1$ ;  $h[n]$  – коэффициенты фильтра-прототипа. Важной особенностью КМБФ является то, что для него разработаны эффективные схемы реализации, основанные на полифазном представлении фильтра-прототипа [7]. Для получения неравнополосной версии КМБФ необходимо выполнить замену всех элементов задержки в его полифазной структуре на фазовые звенья:

$$z^{-1} \rightarrow A(z). \quad (1)$$

В настоящей работе применялись фазовые звенья первого порядка с действительными коэффициентами ( $|\alpha| \leq 1$ ):

$$A(z) = \frac{\alpha + z^{-1}}{1 + \alpha z^{-1}}, \quad A(e^{j\omega}) = e^{j\varphi(\omega)}, \quad \varphi(\omega) = -\omega + 2 \arctg\left(\frac{\alpha \sin \omega}{\alpha \cos \omega - 1}\right).$$

Замена (1) приводит к отображению оси частот  $\omega \mapsto \varphi(\omega)$ , что позволяет получить неравнополосный банк фильтров. На рис. 6 показана полифазная структура неравнополосного КМБФ. Через  $C$  обозначен блок косинусной модуляции, который может быть реализован с использованием дискретного косинусного преобразования четвертого типа, для которого разработан ряд быстрых алгоритмов [8].

Для использования в субполосной системе подавления эффекта АОС (см. рис. 4) был рассчитан 8-канальный передискретизированный неравнополосный КМБФ по методу [9]. Для того чтобы система анализ/синтез неравнополосного КМБФ имела близкую к линейной фазо-частотную характеристику, ее выход должен быть пропущен через фильтр-корректор (на рис. 6 данный фильтр обозначен через  $C(z)$ ). Методика расчета корректора описана в работе [10]. На рис. 7, а, б изображены амплитудно-частотные характеристики (АЧХ) полученного банка фильтров, а его групповая задержка – на рис. 7, в. Коэффициенты децимации/интерполяции неравнополосного КМБФ имеют значения  $S_k = \{19, 9, 6, 4, 2, 1, 1, 1\}$ .

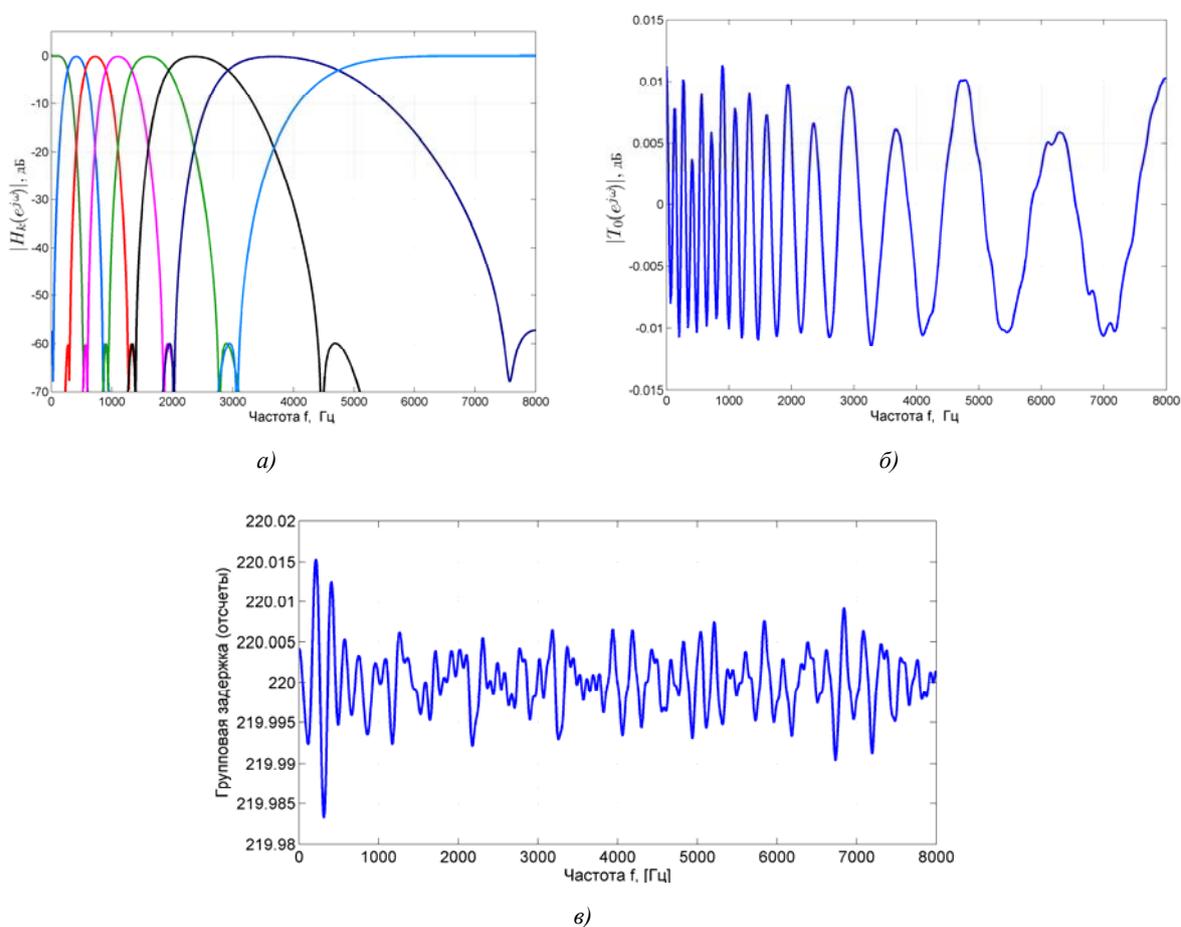


Рис. 7. Характеристики неравнополосного КМБФ: а) АЧХ банка фильтров анализа; б) АЧХ системы анализа/синтеза банка фильтров; в) групповая задержка системы анализа/синтеза банка фильтров

## 2. Оценка эффективности подавления сигнала АОС

### 2.1. Модель канала акустической обратной связи

Главный параметр, который влияет на акустическую обратную связь, – расстояние между динамиком и микрофоном слухового аппарата. Чем это расстояние меньше, тем больше будет эффект, вызываемый обратной связью. По этой причине слуховые аппараты категорий ITE (in-the-ear) и ITC (in-the-canal), у которых данное расстояние минимально, больше всего подвержены влиянию обратной связи. Эмпирические исследования демонстрируют, что частотные

характеристики каналов АОС различных слуховых аппаратов имеют много общих черт. В работе [11] показано, что канал АОС может быть смоделирован передаточной функцией

$$H_{FP}(s) = Ke^{-sT} \frac{(s - z_1)(s - z_2)(s - z_2^*)}{(s - p_1)(s - p_1^*)(s - p_2)(s - p_2^*)}, \quad (2)$$

где  $K$  – коэффициент усиления;  $T$  – задержка распространения сигнала в ушном канале между динамиком и микрофоном;  $z_1, z_2, p_1, p_2$  – параметры, определяющие канал АОС;  $s = \sigma + j\omega$  – комплексная переменная.

Для моделирования канала АОС слухового аппарата (типа ITE) использовались следующие параметры модели (2) [2]:  $K = 1$ ;  $T = 3,8$  мс;  $z_1 = 1500000$ ;  $z_2 = -40000 + 31400j$ ;  $p_1 = -5000 + 22600j$ ;  $p_2 = -500 + 37700j$ . На рис. 8 изображены частотные характеристики моделируемого канала АОС.

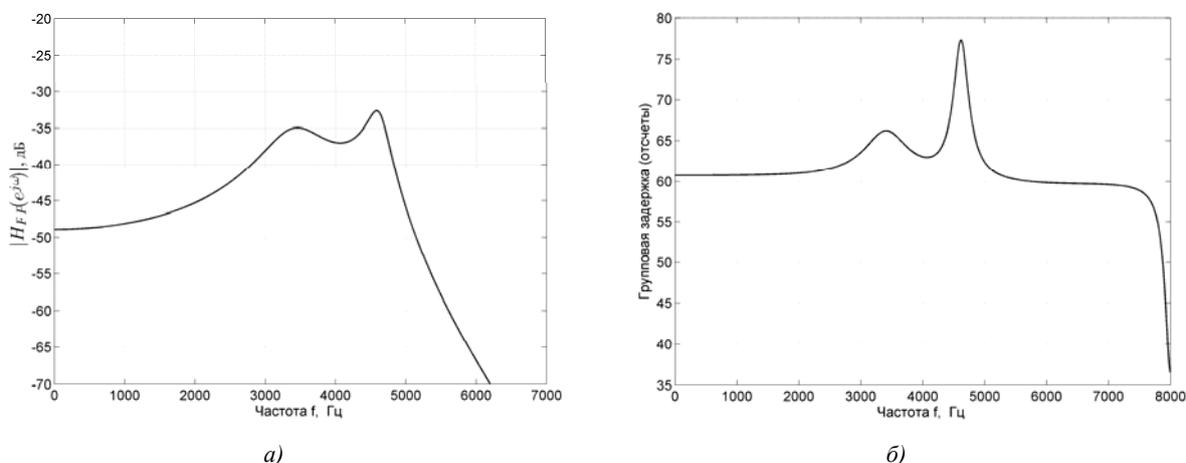


Рис. 8. Характеристики канала АОС: а) АЧХ; б) групповая задержка

## 2.2. Применяемые оценки

Для того чтобы оценить производительность системы подавления АОС, воспользуемся объективным параметром SegSNR (segmented signal-to-noise ratio – сегментированное отношение сигнал-шум), который определяется как среднее значение отношения сигнал-шум каждого выходного блока сигнала слухового аппарата:

$$SegSNR = \frac{1}{B} \sum_{m=1}^B 10 \log_{10} \left( \frac{P_{signal}^{(m)}}{P_{noise}^{(m)}} \right),$$

где  $B$  – число фреймов сигнала;  $P_{signal}^{(m)}$  – мощность сигнала на  $m$ -м фрейме;  $P_{noise}^{(m)}$  – мощность шума на  $m$ -м фрейме. В экспериментах размер фрейма выбирался равным 30 мс. Полезный сигнал получен как выход слухового аппарата в условиях отсутствия АОС. Шум оценивался как разность между полезным сигналом и выходом слухового аппарата с работающей системой подавления эффекта АОС.

Для перцептуальной оценки качества выходного сигнала использовалась оценка PESQ (perceptual evaluation of speech quality) [12]. PESQ является объективным критерием качества канала связи, использующим разницу между спектрами громкости опорного и тестового сигналов для вычисления перцептуального качества тестового сигнала. Оценка PESQ колеблется в пределах от  $-0,5$  до  $4,5$ . Наивысшая оценка показывает, что тестовый сигнал не содержит слышимых искажений и практически идентичен опорному. Оценки 4, 3 и 2 можно интерпретиро-

вать как качество «хорошее», «немного раздражающее», «раздражающее». Оценка между  $-0,5$  и  $1$  свидетельствует о наличии неприемлемо большого количества шума и искажений в тестовом сигнале [4]. В качестве опорного сигнала в данном случае выступает выходной сигнал слухового аппарата в условиях отсутствия АОС, а в качестве тестового – сигнал реального СА, в котором присутствует АОС и активна система подавления.

Измерение разборчивости производилось по методу СИ (speech intelligibility index), который основан на вычислении индекса артикуляции. В соответствии с методикой [13] весь частотный диапазон речевого сигнала разделяется на 18 полос, в пределах которых определяется отношение сигнала к шуму. Сигнал делится на полосы такой ширины, чтобы вклад каждой полосы в результирующую разборчивость был одинаковый (полосы одинаковой разборчивости). В каждой полосе отношение сигнал-шум рассчитывается, определенным образом взвешивается и используется для вычисления коэффициента СИ, который может изменяться в пределах от 0 до 1. Значения индекса принято интерпретировать следующим образом [13]:

- ниже  $0,3$  – плохая разборчивость;
- от  $0,3$  до  $0,5$  – удовлетворительная разборчивость;
- от  $0,5$  до  $0,7$  – хорошая разборчивость;
- выше  $0,7$  – очень хорошая разборчивость.

Для моделирования прямого канала, т. е. обработки сигнала в СА, выбрана усредненная АЧХ (рис. 9), которая компенсирует типичное повреждение слуховой системы. Наибольшие потери происходят в районе  $1,5$  кГц, т. е. в том диапазоне частот, где речь имеет наибольшую информативность. Усиление выходного сигнала в СА регулируется при помощи коэффициента  $g_c$ .

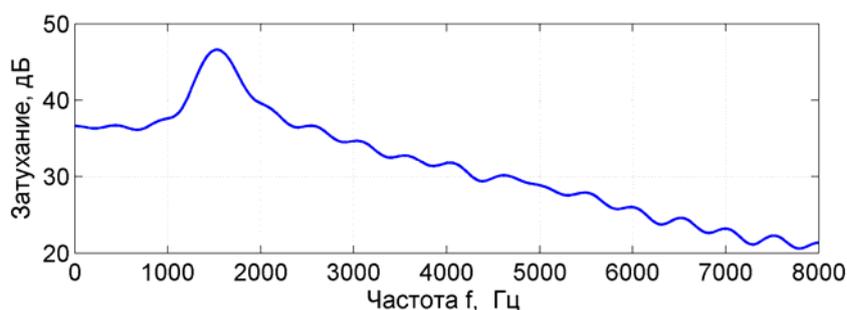


Рис. 9. Модель частотной характеристики слухового аппарата ( $g_c = 46,5$  дБ)

### 2.3. Результаты экспериментов

Оценка производительности системы подавления эффекта АОС производилась на речевом сигнале длительностью  $4,5$  с с частотой дискретизации  $16$  кГц. Сравнивались системы со следующими конфигурациями: без подавления АОС, широкополосная система подавления эффекта АОС (см. рис. 2), субполосная система подавления эффекта АОС (см. рис. 4). Эксперименты проводились для различных порядков адаптивных фильтров ( $L = 64, 128, 256$  и  $512$ ). Порядок субполосных адаптивных фильтров  $L_k$  в зависимости от  $L$  рассчитывался по следующему правилу:

$$L_k = \left\lfloor \frac{L}{S_k} \right\rfloor,$$

где  $S_k$  – коэффициент децимации/интерполяции в  $k$ -м канале, а оператор  $\lfloor a \rfloor$  обозначает наибольшее целое число, меньшее  $a$ . Работа систем проверялась для различных значений коэффициента усиления  $g_c$ . В частности, рассматривались случаи  $g_c = \{45,5; 46,5; 47,5; 48,5; 49,5\}$  дБ.

В табл. 1–3 приведены основные результаты экспериментов (отметка НС в таблицах означает, что система нестабильна).

Таблица 1  
Значения показателя SegSNR при различных коэффициентах усиления  $g_c$  и длине адаптивного фильтра  $L$

Коэффициент усиления $g_c$ , дБ	Без подавления АОС	Широкополосная система подавления эффекта АОС				Субполосная система подавления эффекта АОС			
		64	128	256	512	64	128	256	512
45,5	4,064	4,500	4,440	4,445	4,394	5,059	5,027	4,966	4,712
46,5	0,340	2,087	2,049	2,099	2,032	3,306	3,306	3,240	2,893
47,5	НС	-1,468	-1,611	-1,387	-1,667	0,922	0,862	0,641	-0,232
48,5	–	НС	НС	НС	НС	-2,497	-2,865	-4,038	НС
49,5	–	–	–	–	–	НС	НС	НС	–

Таблица 2  
Значения PESQ при различных коэффициентах усиления  $g_c$  и длине адаптивного фильтра  $L$

Коэффициент усиления $g_c$ , дБ	Без подавления АОС	Широкополосная система подавления эффекта АОС				Субполосная система подавления эффекта АОС			
		64	128	256	512	64	128	256	512
45,5	3,227	3,269	3,269	3,270	3,265	3,301	3,295	3,290	3,284
46,5	2,897	3,006	3,005	3,013	3,004	3,110	3,098	3,087	3,049
47,5	НС	2,623	2,623	2,663	2,668	2,868	2,860	2,844	2,780
48,5	–	НС	НС	НС	НС	2,133	2,161	2,263	НС
49,5	–	–	–	–	–	НС	НС	НС	–

Таблица 3  
Значения показателя SII при различных коэффициентах усиления  $g_c$  и длине адаптивного фильтра  $L$

Коэффициент усиления $g_c$ , дБ	Без подавления АОС	Широкополосная система подавления эффекта АОС				Субполосная система подавления эффекта АОС			
		64	128	256	512	64	128	256	512
45,5	0,549	0,550	0,551	0,552	0,551	0,555	0,555	0,554	0,553
46,5	0,498	0,517	0,518	0,518	0,517	0,527	0,527	0,525	0,523
47,5	НС	0,456	0,455	0,458	0,452	0,486	0,485	0,481	0,469
48,5	–	НС	НС	НС	НС	0,395	0,390	0,372	НС
49,5	–	–	–	–	–	НС	НС	НС	–

Результаты показывают, что применение субполосной техники адаптивной фильтрации позволяет более эффективно бороться с эффектом АОС. По отношению сигнал-шум (см. табл. 1) субполосная система при больших коэффициентах усиления дает выигрыш порядка 1–2,5 дБ. Преимущества субполосной системы подавления выявляют как показатели перцептуального качества речи (PESQ), так и показатели индекса разборчивости речи (SII). Необходимо отметить и тот факт, что адаптивная система работает тем лучше, чем ближе порядок адаптивного фильтра к реальной длине канала АОС.

На рис. 10 показаны результаты работы системы на основе широкополосного подавления эффекта АОС и системы на основе субполосного подавления (для значений параметров  $g_c = 47,5$  дБ,  $L = 64$ ). Из приведенных спектрограмм видно, что скорость сходимости субполосной системы значительно выше, за счет чего уменьшается общий эффект от воздействия АОС. Предложенная система характеризуется повышенным порогом устойчивости и позволяет использовать более высокие коэффициенты усиления для обработки сигнала в прямом канале.

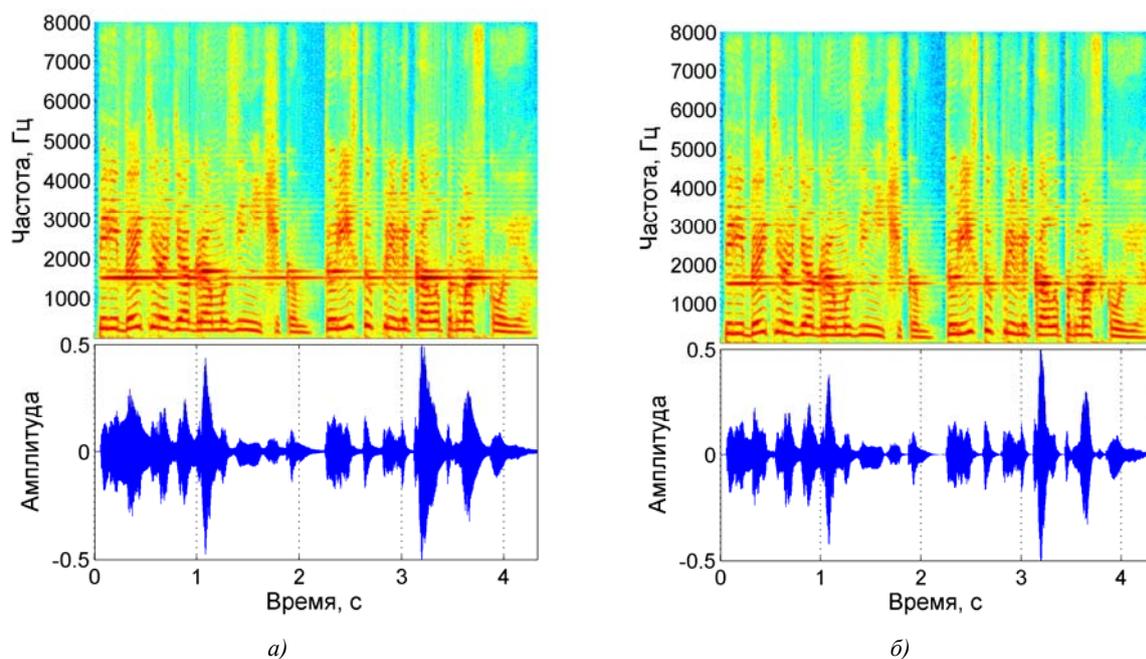


Рис. 10. Спектрограмма сигнала, обработанного системой подавления АОС:  
 а) широкополосная система; б) субполосная система

### Заключение

Предложенная субполосная система подавления акустической обратной связи для слуховых аппаратов, основанная на использовании передискретизированного неравнополосного КМБФ, не вносит дополнительной задержки в прямой путь распространения сигнала в слуховом аппарате. За счет разбиения сигнала на субполосы происходит более быстрая адаптация и более точная оценка компенсирующего сигнала обратной связи. Полученные значения таких объективных показателей, как отношение сигнал-шум, perceptуальное качество канала связи, а также индекс разборчивости показывают, что предложенный подход позволяет значительно улучшить качество выходного речевого сигнала в слуховом аппарате.

### Список литературы

1. Kates, J.M. Feedback cancellation in hearing aids: results from a computer simulation / J.M. Kates // *IEEE Transactions on Signal Processing*. – 1991. – Vol. 39, № 3. – P. 553–562.
2. Acoustic feedback reduction based on filtered-X LMS and normalized filtered-X LMS algorithms in digital hearing aids based on WOLA filterbank / R. Vincen-Bueno [et al.] // *IEEE Intern. Symp. on Intelligent Signal Proc.*, Alcalá, Spain, 3–5 October, 2007. – Alcalá, 2007. – P. 1–6.
3. Bauml, R.W. Uniform polyphase filter banks for use in hearing aids: design and constraints / R.W. Bauml, W. Sorgel // *Proc. of the European signal processing conf.*, Switzerland, Lausanne, August, 2008. – Lausanne, 2008.
4. Pandey, A. Low-delay signal processing for digital hearing aids / A. Pandey, V.J. Mathews // *IEEE Transactions on audio, speech, and language processing*. – 2011. – Vol. 19, № 4. – P. 699–710.
5. Petraglia, M.R. Nonuniform subband adaptive filtering with critical sampling / M.R. Petraglia, P.B. Batalheiro // *IEEE Transactions on signal processing*. – 2008. – Vol. 56, № 2. – P. 565–575.
6. Parfieniuk, M. Tunable nonuniform filter bank mixing cosine modulation with perceptual frequency warping by allpass transformation / M. Parfieniuk, A. Petrovsky // *Automatic control and computer sciences*. – 2004. – Vol. 38, № 4. – P. 44–52.
7. Koilpillai, D. Cosine-modulated FIR filter banks satisfying perfect reconstruction / D. Koilpillai, P.P. Vaidyanathan // *IEEE Transaction on Signal Processing*. – 1992. – Vol. 40, № 4. – P. 770–783.

8. Вашкевич, М.И. Применение полиномиальных алгебр и теории Галуа для синтеза быстрых алгоритмов дискретных косинусных преобразований / М.И. Вашкевич, А.А. Петровский // Цифровая обработка сигналов. – 2011. – № 3. – С. 2–10.
9. Вашкевич, М.И. Проектирование передискретизированного неравнополосного косинусно-модулированного банка фильтров / М.И. Вашкевич, А.А. Петровский // Информатика. – 2011. – № 2 (30). – С. 21–39.
10. Парфенюк, М. Неравнополосный банк фильтров с фазовым преобразованием и объединением субполос для обработки речевых сигналов / М. Парфенюк, М.И. Вашкевич, А.А. Петровский // Речевые технологии. – 2009. – № 4. – С. 53–69.
11. Hellgren, J. System identification of feedback in hearing aids / J. Hellgren, T. Lunner, S. Arlinger // Acoustic Society of America. – 1999. – Vol. 105, № 5. – P. 3481–3496.
12. ITU-T Rec. P 862. Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow band telephone networks and speech coders [Electronic resource]. – 2001. – Mode of access : <http://www.itu.int/rec/T-REC-P.862-200102-I/en>. – Date of access : 11.05.2012.
13. Kamm, C.A. Speech recognition and the Articulation Index for normal and hearing-impaired listeners / C.A. Kamm, D.D. Dirks, T.S. Bell // Acoustic Society of America. – 1985. – Vol. 77. – P. 281–288.

Поступила 17.04.12

*Белорусский государственный университет  
информатики и радиоэлектроники,  
Минск, П. Бровки, 6  
e-mail: vashkevich@bsuir.by,  
palex@bsuir.by*

**M.I. Vashkevich, E.S. Azarov, A.A. Petrovsky**

### **ACOUSTIC FEEDBACK CANCELLATION IN HEARING AID USING NONUNIFORM FILTER BANK**

A method of acoustic feedback prevention for hearing aid has been proposed. The method is based on the subband adaptive filtering. The main distinguishing feature of the method is that no delay is added into the forward path of the hearing aid. Due to the non-uniform subband signal decomposition the method provides higher convergence rate and higher suppression level than the full band adaptive filtering.

УДК 004.934.5

Л.И. Цирульник, Д.А. Покладок

## ГРАММАТИЧЕСКИЙ СЛОВАРЬ И ПРАВИЛА ОПРЕДЕЛЕНИЯ СЛОВЕСНОГО УДАРЕНИЯ ДЛЯ СИНТЕЗА РЕЧИ ПО ТЕКСТУ НА МОБИЛЬНЫХ УСТРОЙСТВАХ

*Описываются исследования, направленные на выявление закономерностей расстановки ударений в словах русского языка. Предлагается алгоритм расстановки словесных ударений в произвольных текстах. Тестирование полученных результатов показывает, что с использованием разработанных правил точность расстановки словесных ударений уменьшается всего на 1,4 % при уменьшении объема грамматического словаря в восемь раз. Такое уменьшение объема позволяет использовать словарь в системе синтеза речи для мобильных устройств, характеризующихся малым объемом памяти.*

### Введение

Системы синтеза речи по тексту к настоящему моменту достигли высокого качества как по критериям разборчивости и естественности синтезируемого голоса, так и по техническим характеристикам, что способствует их широкому применению в центрах обработки вызовов, при управлении сложными объектами, для создания аудиокниг и т. д. Широкое распространение может получить использование систем синтеза речи на мобильных устройствах, таких как карманные персональные компьютеры, смартфоны, мобильные телефоны. Примерами приложений могут служить озвучивание SMS-сообщений [1] и указаний автомобильной навигационной системы [2], чтение писем электронной почты [3].

Большинство современных систем синтеза речи для мобильных устройств, например Acapela TTS for Windows Mobile [4], Nuance TALKS [5], Mobile Speak [6] и др., работают на смартфонах под управлением операционных систем Windows Mobile или Symbian. Однако смартфоны, на которых возможна работа этих систем синтеза речи, составляют, по различным подсчетам, только 10–15 % рынка мобильных телефонов [7, 8].

Основная масса мобильных аппаратов, используемых в настоящее время, характеризуется низким быстродействием и относительно небольшим объемом памяти. В то же время современные системы синтеза речи требуют большого объема памяти для хранения лингвистических и акустических ресурсов, что не позволяет напрямую «перенести» существующие системы на мобильные платформы. При этом одним из основных информационных ресурсов системы синтеза речи является грамматический словарь, который используется для определения словесного ударения и лексико-грамматических характеристик слов входного текста. Объем электронного грамматического словаря достигает 50 МБ [9].

Целью исследований, описанных в настоящей статье, является выявление закономерностей расстановки словесного ударения с целью уменьшения объема словаря. Подобные исследования проводились и ранее. Например, в работе [10] приводятся эвристические правила определения словесного ударения, основанные на морфологических признаках слов. Однако приводимые правила выполняются далеко не всегда, их использование требует корректировки и уточнения. В работе [11] описан вероятностный метод определения позиции словесного ударения, учитывающий стандартные префиксы и словоизменительные окончания русского языка. Этот метод требует нахождения возможных грамматических характеристик и части речи слова для определения в нем позиции ударения. Он показывает более точные результаты, чем предыдущий, однако не может быть применен на мобильных устройствах, поскольку необходимость нахождения морфологических и грамматических характеристик слов потребует либо большого объема памяти, либо больших вычислительных ресурсов, что практически нивелирует преимущества автоматической расстановки позиции словесного ударения. В настоящей статье описан метод определения словесного ударения, основанный на эвристических и статистических правилах.

## 1. Расстановка ударений в словаре

Исследования проводились в два этапа. На первом этапе проверялись и корректировались эвристические правила расстановки словесных ударений, на втором – выявлялись статистические правила расстановки ударений.

Набор эвристически выявленных закономерностей включает 47 правил (содержащих 171 подправило), которые формулируются, например, так: *«Если слово заканчивается на «-льон-» (плюс окончание), то ударение падает на этот слог»* или *«В словах, содержащих суффиксы «-оват-», «-еват-», ударение падает на вторую гласную суффикса»*.

В качестве материала для исследований был взят грамматический словарь А.А. Зализняка [12]. Словарь содержит около 100 000 слов и отражает (с помощью специальной системы условных обозначений) современное словоизменение, т. е. склонение существительных, прилагательных, местоимений, числительных и спряжение глаголов. Для исследования была использована электронная версия словаря [9], в котором для каждого слова указана позиция ударения. Словарь был сформирован путем построения парадигм всех слов и последующего удаления дубликатов. С целью исключения неоднозначной интерпретации из словаря были удалены омографы, а также клитики и частично-ударные слова. Объем полученного словаря составил 2 070 816 уникальных слов.

Эвристические правила были запрограммированы, затем для каждого правила было выявлено количество слов в словаре, удовлетворяющих ему, а также количество слов, для которых данное правило не выполняется. Если правило выполнялось менее чем в 80 % случаев, оно исключалось из дальнейшего рассмотрения. В результате было оставлено 41 правило, содержащее 125 подправил.

Далее были выявлены и удалены правила, полностью и почти полностью поглощаемые другими. Например, исследования показали, что правилу *«Если слово заканчивается на «-ованн-» плюс окончание, то ударение падает на первую гласную подстроки»* удовлетворяют 10 812 слов, из которых 10 786 удовлетворяют правилу *«Если слово заканчивается на подстроку «-иров-» плюс окончание и перед подстрокой в слове стоит по меньшей мере один слог, то ударение падает на первую гласную подстроки»*. Второму из правил удовлетворяет 46 451 слово. Таким образом, первое правило почти полностью поглощается вторым, но не наоборот. В результате удаления поглощаемых правил осталось 36 правил, содержащих 98 подправил.

В ходе исследований было выявлено, что некоторые слова удовлетворяют условиям двух правил, но при этом позиция ударения в них соответствует одному из правил, но не соответствует другому. Например, слово *«биологи»* удовлетворяет условиям следующих двух правил, но только первое из них выполняется: *«Если слово заканчивается на подстроку «-олог» плюс окончание, то ударение падает на первую гласную подстроки»* и *«В словах, заканчивающихся на подстроку «-логи-» плюс окончание, ударение падает на первую гласную подстроки»*. Для того чтобы результирующим правилам удовлетворяло максимально возможное число слов, для каждого правила был определен приоритет. При этом из двух правил менее приоритетным считалось то, большее количество исключений из которого удовлетворяет другому правилу. Если же исключения из правила не удовлетворяют ни одному другому правилу, то оно имеет наивысший приоритет.

После такого рода уточнения и корректировки эвристических правил была вычислена статистика выполнения этих правил на материале словаря. В итоге было установлено, что из 2 070 816 слов эвристическим правилам удовлетворяют 786 179 слов, или 38 % объема словаря. Данные слова были удалены из словаря, в результате чего его объем составил 1 284 637 слов.

На следующем этапе исследований выявлялись, как уже было сказано выше, статистические характеристики позиций ударения в словах. Для этого все слова были разбиты на классы в зависимости от количества слогов в них, т. е. образовался класс двусложных, трехсложных и т. д. слов (максимальное количество слогов в словах равнялось 11). Далее для каждого класса вычислялось количество слов, в которых ударение падает на первый слог, второй слог и т. д. вплоть до числа слогов в данном классе. Результаты вычислений представлены в таблице, где каждая строка характеризует определенный класс (т. е. слова с определенным количеством слогов), каждый столбец – количество слов из этого класса, в которых ударение падает на первый,

второй и т. д. слог. Например, общее количество трехсложных слов в словаре равно 235 525. Из них, как видно из третьей строки таблицы, количество слов с ударением на первый слог – 58 371, с ударением на второй слог – 126 305 и с ударением на третий слог – 50 849.

Статистика распределения позиций ударения в зависимости от количества слогов в слове

Кол-во слогов слова	Количество слов, в которых ударение падает на слог с номером										
	1	2	3	4	5	6	7	8	9	10	11
2	34272	25158	–	–	–	–	–	–	–	–	–
3	58371	126305	50849	–	–	–	–	–	–	–	–
4	46585	141078	165881	14241	–	–	–	–	–	–	–
5	22939	80725	156218	44445	2675	–	–	–	–	–	–
6	4576	22363	86140	44435	7541	422	–	–	–	–	–
7	415	2062	18383	22862	8121	1140	34	–	–	–	–
8	20	71	896	4109	4441	1165	111	0	–	–	–
9	8	0	0	171	923	613	110	1	0	–	–
10	0	0	0	15	17	91	52	5	0	0	–
11	0	0	0	0	0	0	0	1	0	0	0

На основании полученных результатов было сформулировано следующее правило: ударным слогом в слове из *n* слогов принимается тот, на который чаще всего падало ударение в исследуемых словах. Например, в трехсложных словах, согласно этому правилу, ударным принимается второй слог, в четырехсложных словах – третий слог и т. д. Можно утверждать, что в общем случае ударение падает на серединный или следующий за ним слог слова.

Все слова, ударения в которых соответствуют сформулированному правилу, были удалены из словаря. Общее количество удаленных слов равнялось 531 867, или 41,4 %.

В результате выявления и использования эвристических и статистических правил объем словаря был уменьшен на 64 % и составил 752 770 слов.

## 2. Расстановка ударений и частоты встречаемости слов в корпусе текстов

Результаты, полученные на основе словаря, могут оказаться не совсем объективными, поскольку не все слова, встречающиеся в текстах на естественном языке, присутствуют в словаре. Для получения более объективных результатов, а также для проверки точности эвристических правил необходимо провести исследования на корпусе текстов, причем в корпусе должны быть указаны словесные ударения.

Кроме того, корпус текстов можно использовать для дальнейшего уменьшения содержания словаря, а именно для удаления из него редко встречающихся (или вообще неупотребимых) слов. К таким словам относятся, в частности, словоформы, образование которых затруднительно (некоторые краткие прилагательные, например «деревенск», «геройск», «величайш»; некоторые существительные в родительном падеже множественного числа, например «мечт», «брюзг», «башок»; некоторые страдательные причастия и т. п.), а также устаревшие слова (например «уповать», «дабы» и т. п.). Корпус текстов, используемый для удаления из словаря неупотребимых слов, должен содержать максимальное количество словоформ, но не обязательно должен иметь, в отличие от первого корпуса, указатели словесных ударений.

Для решения второй задачи использовался Национальный корпус русского языка (НКРЯ) [13], который характеризуется представительностью и содержит художественные, публицистические, учебные, научные, деловые, разговорные, диалектные и другие тексты общим объемом более 190 млн словоупотреблений. Для решения первой задачи использовался подкорпус НКРЯ со снятой омонимией (и с указанными позициями словесных ударений) объемом около 6 млн словоупотреблений.

Проверка точности эвристических правил осуществлялась путем выполнения запросов к подкорпусу НКРЯ и обработки полученных наборов слов. Результат проверки представлен на

рис. 1, где по оси абсцисс показаны номера правил и подправил, расположенные по приоритету, по оси ординат – количество слов, позиция ударения в которых соответствует данному правилу.

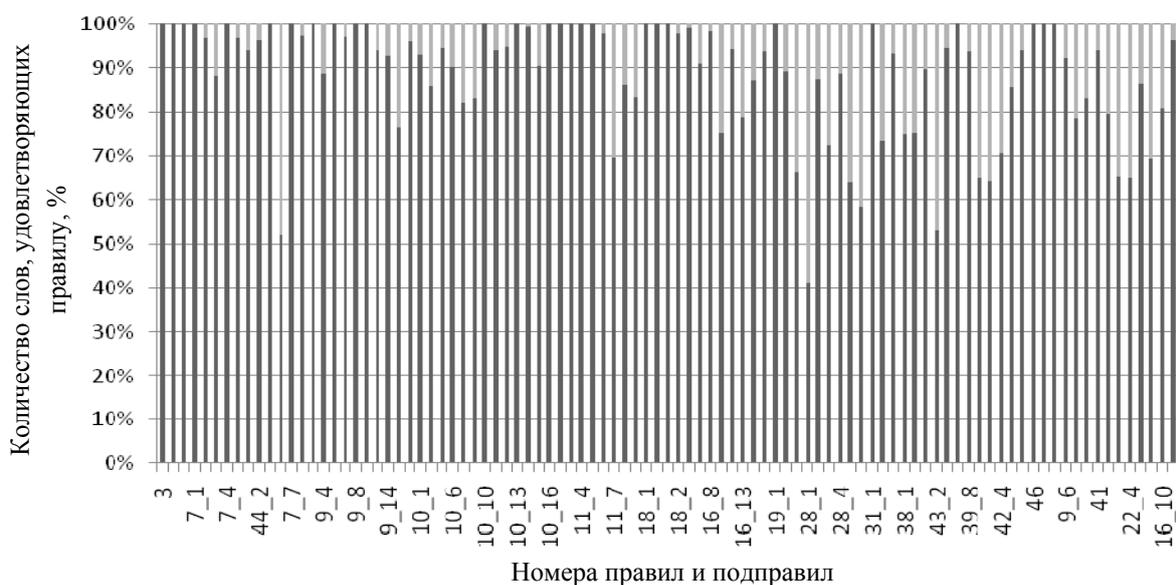


Рис. 1. Статистические характеристики выполнения эвристических правил в корпусе текстов

Как видно из рис. 1, для 20 правил из 98 количество слов, удовлетворяющих этим правилам, не превышает 80 %. Условия данных правил были уточнены, после чего были заново вычислены статистические показатели выполнения всех правил. Для трех правил процент выполнения не превысил 80, и эти правила были удалены.

При проверке точности эвристических правил в НКРЯ было найдено 5 295 слов (более 11 300 словоупотреблений с учетом частоты встречаемости слов), отсутствующих в словаре. Из них 940 слов (2 193 словоупотребления) явились исключениями из правил, и в них были неправильно определены позиции ударений. Однако в процентном соотношении количество ошибок составило 1,8 % от всех проверяемых слов (или 1 % с учетом частоты встречаемости этих слов). Таким образом, точность эвристических правил с учетом частоты встречаемости слов достигает 99 %.

Для определения неупотребимых слов была вычислена частота встречаемости слов словаря в НКРЯ (рис. 2).



Рис. 2. Результаты вычисления частоты встречаемости слов словаря в НКРЯ

Вычисления показали, что из 752 770 слов словаря 486 627 слов ни разу не встретились в НКРЯ. Следовательно, они могут быть удалены из словаря. Результирующий объем словаря составил 266 143 слова, или 12 % от исходного объема.

### 3. Алгоритм определения словесного ударения

На основании результатов исследований был разработан алгоритм определения позиции ударения в произвольном слове. Он состоит из трех этапов: поиск по словарю, поиск по эвристическим правилам, поиск по статистическим правилам. Если на каком-то этапе позиция ударения определена, поиск прекращается.

*Входные данные алгоритма:* произвольное слово, в котором необходимо определить позицию ударения.

Ресурсы алгоритма:

- грамматический словарь с указанием словесных ударений;
- набор эвристических и статистических правил определения позиции ударения.

*Выходные данные:* слово с указанной позицией ударения.

*Структура алгоритма:*

Шаг 1. Осуществляется поиск входного слова в словаре. Если слово найдено, в нем устанавливается ударение согласно отмеченному в словаре и происходит переход к шагу 5.

Шаг 2. Для каждого  $i$  от 1 до  $n$ , где  $n$  – количество эвристических правил,

{

*если* слово удовлетворяет условию  $i$ -го правила,  
*то* ударение устанавливается согласно  $i$ -му правилу и  
происходит переход к шагу 5.

}

Шаг 3. Вычисляется количество слогов в слове  $j$ .

Шаг 4. Ударение устанавливается согласно статистическому правилу для слов, имеющих  $j$  слогов.

Шаг 5. Конец алгоритма.

### 4. Тестирование точности алгоритма определения позиции ударения

Тестирование проводилось на части НКРЯ, предоставляемой для свободного использования. Эта часть является случайной выборкой предложений из корпуса со снятой омонимией (с указанными позициями ударений) объемом 180 тысяч словоупотреблений.

Из текстов подкорпуса были предварительно удалены омографы, цифры, символы, не являющиеся буквами русского алфавита, и ошибочные слова (без указанных позиций ударений).

Тестировалась корректность алгоритма определения позиций ударения с использованием исходного словаря (объемом более 2 млн словоформ) и сокращенного словаря (объемом немногим более 266 тыс. словоформ).

Результаты тестирования для слов и словоупотреблений в подкорпусе представлены на рис. 2 и 3 соответственно. По оси абсцисс на рисунках цифрами 1, 2, 3 показаны этапы выполнения алгоритма: 1 – поиск в словаре, 2 – определение позиции ударения в соответствии с эвристическими правилами, 3 – определение позиции ударения в соответствии со статистическими правилами. Для каждого этапа показано процентное соотношение слов (на рис. 2) или словоупотреблений (на рис. 3), для которых на этом этапе была правильно либо ошибочно определена позиция ударения. Доли правильно определенных ударений показаны светло-серым цветом, доли ошибочно определенных – черным. Например, на рис. 2, *a* показано, что на этапе 1 (поиск в словаре) были правильно определены позиции ударения для 93,4 % слов подкорпуса. На этапе 2 (применение эвристических правил) были правильно определены ударения в 1,8 % слов подкорпуса и ошибочно – в 0,2 % слов. На этапе 3 (применение статистических правил) из оставшихся 4,6 % слов для половины (2,3 %) позиции ударения были определены правильно и для половины – ошибочно.

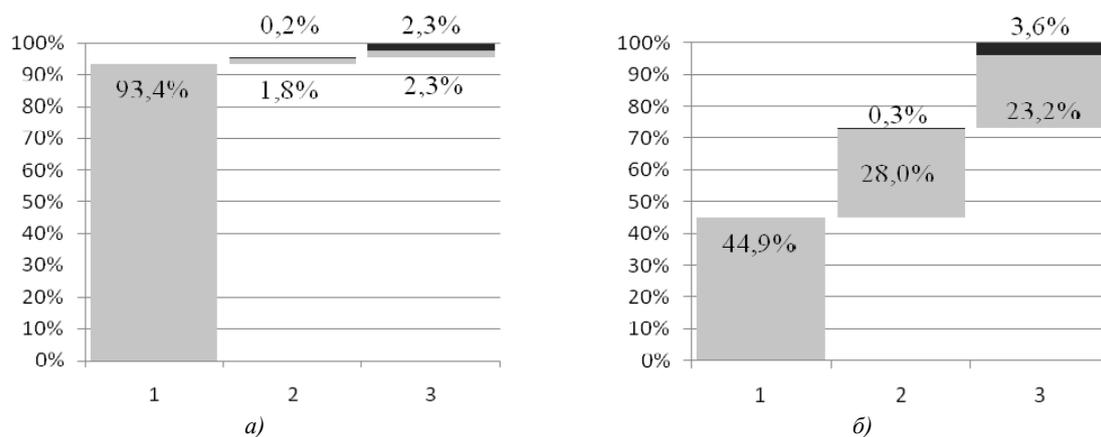


Рис. 2. Точность определения позиции ударения для слов подкорпуса с использованием: а) полного словаря; б) сокращенного

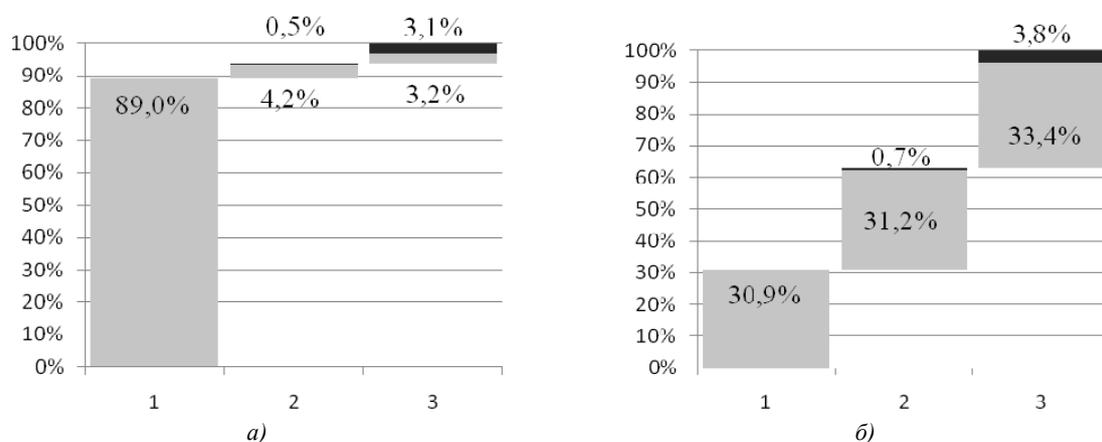


Рис. 3. Точность определения позиции ударения для словоупотреблений подкорпуса с использованием: а) полного словаря; б) сокращенного

Из рисунков видно, что в полном словаре можно найти более чем в два раза больше слов (и почти в три раза больше словоупотреблений), чем в сокращенном. Эвристические правила могут быть применены к 2 % слов (или 4,7 % словоупотреблений), оставшихся после поиска в полном словаре, в то время как среди слов, оставшихся после поиска в сокращенном словаре, 28,3 % (31,9 % словоупотреблений) удовлетворяют условиям эвристических правил. Процент правильно определенных ударений с использованием эвристических правил довольно высок как в случае использования полного, так и в случае использования сокращенного словарей. Рисунки также показывают, что применение статистических правил дает удовлетворительные результаты: в 23,2 % слов (33,4 % словоупотреблений) ударения определены правильно и только в 3,6 % слов (3,8 % словоупотреблений) – ошибочно при использовании на предыдущих этапах сокращенного словаря.

Рисунки также наглядно демонстрируют, что точность алгоритма определения позиции ударения с использованием сокращенного словаря составляет 95,4 % (или, с учетом частоты встречаемости слов, 96,1 %), что меньше точности того же алгоритма с использованием полного словаря всего на 0,9 % (или, с учетом частоты встречаемости слов, на 1,4 %).

##### 5. Особенности структуры электронного словаря и программной реализации процедуры поиска по словарю

При разработке формата хранения словаря ставились две задачи: уменьшение объема словаря и ускорение процедуры поиска по словарю. Поиск осуществляется с помощью хэш-таблиц, что обеспечивает вычислительную сложность процедуры поиска порядка  $O(1)$  [14].

Для уменьшения объема электронного словаря сохранялись не сами слова, а соответствующие им хэш-коды. Для каждого слова словаря был сгенерирован 26-битовый хэш-код и 6-битовый код варианта позиции ударения. В результате под каждое слово было выделено 4 байта. Данный список был отсортирован в порядке убывания и разделен на 256 файлов. Именами этих файлов являются старшие байты хэш-кодов слов в шестнадцатеричном представлении. Таким образом, хранение старшего байта осуществляется в имени файла, а остальные 3 байта расположены в одноименном файле в порядке убывания для последующего быстрого поиска.

В результате программная версия словаря, содержащего 266 143 слова, составила 789 кБ.

### **Заключение**

В результате исследований были уточнены эвристические правила определения позиции ударения в словах русского языка, а также сформулированы статистические правила. Использование этих правил позволило сократить объем грамматического словаря с более 2 млн слов до чуть более 260 тыс. слов, или в восемь раз.

Предложен алгоритм определения словесного ударения, использующий эвристические и статистические правила расстановки ударений, а также грамматический словарь (полный либо сокращенный). Тестирование работы данного алгоритма на текстах НКРЯ показало, что при использовании сокращенного словаря точность определения позиции ударения уменьшается всего на 1,4 %.

Для хранения сокращенного словаря в памяти использовались хэш-коды слов и коды варианта позиции ударения, что позволило уменьшить физический объем словаря до 790 кБ.

Словарь такого объема может использоваться в системах синтеза речи для мобильных устройств, характеризующихся малым объемом памяти.

Авторы благодарят разработчиков Национального корпуса русского языка, позволившего провести данные исследования, и персонально Леонида Лейбовича Иомдина и Алексея Игоревича Зобнина за помощь в вычислении частоты встречаемости слов в НКРЯ.

### **Список литературы**

1. SMS2Voice. Сервис голосовых сообщений [Электронный ресурс]. – 2002. – Режим доступа : <http://voice.s-soft.org>. – Дата доступа : 08.12.11.
2. What Does Garmin GPS System Text to Speech Mean? // eHOW [Electronic recourse]. – 2010. – Mode of access : [http://www.ehow.com/info\\_8550345\\_garmin-system-text-speech-mean.html](http://www.ehow.com/info_8550345_garmin-system-text-speech-mean.html). – Date of access : 08.12.11.
3. VoiceOver // Apple [Электронный ресурс]. – 2011. – Режим доступа: <http://www.apple.com/ru/accessibility/voiceover/>. – Дата доступа : 08.12.11.
4. Acapela TTS for Windows Mobile // Acapela [Electronic recourse]. – 2005. – Mode of access : <http://www.acapela-group.com/acapela-tts-for-windows-mobile-2-2-speech-solutions-tts.html>. – Date of access : 08.12.11.
5. Nuance TALKS // Nuance [Electronic recourse]. – 2004. – Mode of access : <http://www.nuance.com/talks/>. – Date of access : 08.12.11.
6. Mobile Speak [Electronic recourse] // Code Factory. – 2004. – Mode of access : <http://www.codefactory.es/en/products.asp?id=316>. – Date of access : 08.12.11.
7. Gartner Says Worldwide Mobile Phone Sales Grew 17 Per Cent in First Quarter 2010. Press Release // Gartner [Electronic recourse]. – 2010. – Mode of access : <http://www.gartner.com/it/page.jsp?id=1372013>. – Date of access : 08.12.11.
8. Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011; Smartphone Sales Increased 42 Percent. Press Release // Gartner [Electronic recourse]. – 2011. – Mode of access : <http://www.gartner.com/it/page.jsp?id=1848514>. – Date of access : 08.12.11.
9. Цирульник, Л.И. Алгоритмы создания и пополнения грамматического словаря русского языка для синтеза речи по тексту / Л.И. Цирульник, В.В. Веремей // Информатика. – 2012. – № 1 (31). – С. 27–38.

10. Алгоритм поиска ударений [Электронный ресурс]. – 2007. – Режим доступа : [http://proteus2001.narod.ru/802/data/00.htm#accent search](http://proteus2001.narod.ru/802/data/00.htm#accent%20search). – Дата доступа : 08.12.11.
11. Автоматическое определение места ударения в незнакомых словах в системе синтеза речи / О.Г. Хомицевич [и др.] // Материалы XXXVI Междунар. филологической конф. – СПб., 2008. – С. 175–183.
12. Зализняк, А.А. Грамматический словарь русского языка: словоизменение / А.А. Зализняк. – 2-е изд., стереотип. – М. : Рус.яз., 1980. – 880 с.
13. Национальный корпус русского языка [Электронный ресурс]. – 2000. – Режим доступа : <http://www.ruscorpora.ru/>. – Дата доступа : 08.12.11.
14. Кнут, Д.Э. Искусство программирования / Д.Э. Кнут. – 2-е изд. – М. : Вильямс, 2011. – Т. 3 : Сортировка и поиск. – 824 с.

Поступила 08.12.2011

*Объединенный институт проблем  
информатики НАН Беларуси,  
Минск, Сурганова, 6  
e-mail: liliya.tsirulnik@gmail.com,  
dima.pokladok@gmail.com*

**L.I. Tsirulnik, D.A. Pokladok**

#### **GRAMMATICAL DICTIONARY AND STRESS PLACEMENT RULES FOR MOBILE DEVICE TTS-SYNTHESIS**

Studies of stress placement regularities for Russian language are described. The research shows that the size of the dictionary can be significantly reduced. An algorithm of word stress placement is suggested. Testing of the results shows that the size of the dictionary can be reduced eight times, while the word stress placement accuracy decreases just 1.4 %. Reducing dictionary size allows using it in mobile devices, which are characterized by a limited memory.

## ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

УДК 621.38

П.Н. Бибило, А.Л. Соловьев

**ФУНКЦИОНАЛЬНЫЕ VHDL-МОДЕЛИ ЭЛЕМЕНТОВ FPGA  
СЕМЕЙСТВА SPARTAN 3 ДЛЯ КОНВЕРТАЦИИ ПРОЕКТОВ  
ЦИФРОВЫХ СИСТЕМ В ЗАКАЗНЫЕ СБИС**

*На основе анализа структуры микросхемы FPGA xc3s1000-4ft256 и экспериментального изучения функций элементов предлагаются синтезируемые VHDL-модели тех логических элементов, которые входят в конфигурируемые логические блоки FPGA семейства Spartan 3. Необходимость в таких моделях возникает при конвертации реализованных на FPGA проектов цифровых систем в проекты, пригодные для синтеза в библиотеках проектирования заказных СБИС.*

**Введение**

Широко распространенной практикой проектирования цифровых систем является этап их предварительной реализации на FPGA. Такая реализация позволяет быстро оценить сложность и быстродействие проекта и получить экспериментальный образец действующей аппаратуры. Однако для специальных применений требуется использование только отечественной элементной базы. Перевод проекта, реализованного на FPGA, в описание, пригодное для реализации на отечественной элементной базе, является важной научно-технической задачей. Если все спецификации исходного, т. е. реализованного на FPGA, проекта цифровой системы заданы на языке VHDL, то реализация такого проекта на заказной СБИС значительно облегчается [1]. Проблема усложняется, если FPGA-проект должен быть переведен в заказную СБИС из формата топологического редактора (FPGA Editor) системы WebPack ISE (Integrated Software Environment – САПР фирмы Xilinx) [2] автоматизированного проектирования программируемых логических интегральных схем фирмы Xilinx. В данном редакторе после автоматического размещения элементов и трассировки соединений (этап Place & Route) проект может быть подвергнут «ручной» модификации. Внутренними форматами представления проектов на этапе топологического проектирования в системе ISE являются форматы NCD (Native Description Language) и текстовый формат XDL (Xilinx Design Language). В литературе рассматривались задачи использования внутреннего формата XDL описания FPGA-проектов для реализации в составе программируемых микросхем специальных подсхем – макросов. Например, в [3] такая задача рассматривалась для микросхем семейств Virtex II и Virtex II Pro.

Основными блоками FPGA xc3s1000-4ft256 семейства Spartan 3 являются блоки ввода-вывода, конфигурируемые логические блоки (Configurable Logical Block – CLB), блочная память (Block RAM – BRAM) и 18-разрядные умножители (Multiplier) [4]. Процессы перепроектирования макроэлементов ОЗУ и умножителей для их реализации в составе заказных СБИС представляют собой отдельные задачи. Далее рассматривается перепроектирование той части проекта, которая реализуется в виде совокупности сетей CLB. Конвертацию таких проектов FPGA предлагается выполнять, решая две основные задачи конвертации.

**1. Задачи конвертации проектов FPGA**

Первой задачей конвертации является задача получения структурных описаний проектов, реализованных на FPGA. Чтобы структурные описания были синтезируемыми, необходимо иметь синтезируемые функциональные (поведенческие) описания всех элементов, которые используются в полученных структурных описаниях. Таковыми являются элементы, составляющие конфигурируемые логические блоки. Будем называть эти элементы базисными элементами CLB, а совокупность базисных элементов CLB обозначать  $S$ .

Второй задачей конвертации, которая и рассматривается в данной статье, является *задача разработки синтезируемых функциональных (поведенческих) VHDL-описаний* элементов, составляющих множество  $S$ .

Заметим, что интерфейсы VHDL-описаний элементов множества  $S$  имеются в фирменной документации на FPGA семейства Spartan 3, доступной по сети Интернет [5]. В служебных библиотеках системы проектирования ISE имеются соответствующие VHDL-описания, которые, однако, являются несинтезируемыми и пригодны лишь для целей моделирования проекта после выполнения этапа размещения и трассировки проекта. В этих описаниях, пригодных для моделирования, большое число параметров языка VHDL используется для задания временных характеристик элементов FPGA. Однако при синтезе схем по VHDL-описаниям временные зависимости в исходных описаниях либо игнорируются, либо заменяются другими, определяемыми временными задержками элементов целевой библиотеки синтеза. При конвертации FPGA-проектов важно правильно формировать функции проектов, не принимая во внимание задержки сигналов, поэтому функциональные VHDL-описания элементов из  $S$  могут быть составлены без указания конкретных временных задержек, т. е. временные задержки элементов из  $S$  полагаются далее равными нулю.

Рассматриваемая в данной статье задача имеет большое практическое (техническое) значение, так как ее решение, а именно получение синтезируемых функциональных описаний для элементов из  $S$ , позволяет выполнять синтез исходного проекта в другом базисе логических элементов, входящих в ту или иную целевую библиотеку проектирования заказной СБИС. Для целей синтеза схем заказных СБИС применялся промышленный синтезатор LeonardoSpectrum, который был настроен на пользовательскую библиотеку проектирования, описанную в [1, с. 342]. Множества синтезируемых и несинтезируемых конструкций языка VHDL для этого синтезатора изучались в [6].

## 2. Конфигурируемый логический блок

Основной структурной единицей FPGA является блок CLB. Он состоит из двух секций левой руки (SLICEM, где M – от слова Memory) и двух секций правой руки (SLICEL, где L – от слова Logic), входы и выходы которых подсоединены к переключательной матрице соединений (Switch Matrix). Через эту матрицу блоки CLB соединены с линиями глобальных соединений. На рис. 1 изображение секции SLICEM получим тогда, когда штриховые линии будут заменены непрерывными. Если же на рис. 1 удалить те элементы и те их соединения, которые изображены штриховыми линиями, то получим изображение секции SLICEL.

Рассмотрим сначала более простую секцию SLICEL (рис. 1 без штриховых линий). Она состоит из двух LUT (Look-Up Table), двух программируемых элементов памяти (FFX, FFY), нескольких логических элементов (И (AND), «сумма по модулю два» (XOR)) и мультиплексоров с одним управляющим входом (MUX2). Связи структурных элементов CLB являются *программируемыми*, их можно оставлять в схеме либо удалять из нее с помощью так называемых программируемых соединений, что позволяет реализовать в одной секции SLICEL разнообразные логические схемы, выполняющие различные функции. Пример программируемого соединения (точки коммутации проводников) можно увидеть в правом верхнем углу на рис. 1 перед выходным полюсом YB. Другие программируемые соединения показаны на входах элементов треугольниками. Программируемый элемент LUT является универсальным логическим элементом и может реализовать любую булеву функцию не более чем от четырех переменных. Элементы памяти могут функционировать как FF (Flip-Flop – триггер, синхронизируемый фронтом) либо как Latch (триггер с потенциальным управлением (защелка)). Таким образом, в секции SLICEL можно реализовать схему с памятью – конечный автомат.

Структура секции SLICEM (рис. 1 со штриховыми линиями) усложнена по сравнению с секцией SLICEL. Усложнение касается реализации LUT. Данные программируемые элементы могут быть настроены на реализацию логических функций – аналогично, как и в секции SLICEL. Дополнительно они могут выполнять функции блока памяти RAM либо функции сдвигового регистра. Два LUT (16-разрядный сдвиговый регистр) могут быть соединены в один 32-разрядный сдвиговый регистр, три – в 48-разрядный и т. д. Для этого в состав SLICEM вводится вспомогательный элемент WSGEN, осуществляющий общую синхронизацию сдвигов, и другие вспомогательные элементы.



- блока памяти (RAM);
- сдвигового регистра заданной длины (не более 16 бит).

### 3.1. Функциональная модель программируемого элемента LUT, используемого как комбинационный элемент

В структурном описании FPGA-проекта, заданном в формате XDL системы проектирования WebPack ISE 13.1 [2], логические функции программируемого элемента LUT, используемого как комбинационный элемент, описываются с помощью скобочных логических выражений, содержащих только четыре оператора: «@» – сумму по модулю 2, «~» – отрицание, «\*» – конъюнкцию и «+» – дизъюнкцию. Так, например, выражение

$$D = (\sim A1 * (A3 @ A4)) + (A1 * A2 * A4)$$

задает логическую функцию

$$D = (\overline{A1} \& (A3 \oplus A4)) \vee (A1 \& A2 \& A4),$$

которая на VHDL представляется оператором <= назначения сигнала

$$D \leq (\text{not } A1 \text{ and } (A3 \text{ xor } A4)) \text{ or } (A1 \text{ and } A2 \text{ and } A4).$$

В VHDL логические операторы имеют следующие обозначения: not (отрицание), and (конъюнкция), or (дизъюнкция), xor (сумма по модулю 2).

Другой формат структурного описания FPGA-проекта, получаемый после выполнения этапа Place & Route проектирования FPGA в системе ISE, использует задание функции, реализуемой элементом LUT, в виде четверки чисел. Такое задание соответствует представлению 16-разрядного булева вектора – столбца значений булевой функции, зависящей от четырех переменных. Например, в последнем столбце табл. 1 записан набор 6CA0 из четырех чисел ( $6_{16} = 6_{10}$ ,  $C_{16} = 12_{10}$ ,  $A_{16} = 10_{10}$ ,  $0_{16} = 0_{10}$ ). Данный набор кодирует столбец значений булевой функции так, как показано в табл. 1. На языке VHDL табличное представление булевой функции записывается очевидным образом – составляется совершенная ДНФ функции, которая записывается с помощью операторов VHDL (листинг 1).

Таблица 1

ADR0	ADR1	ADR2	ADR3	D	Код (6CA0)
0	0	0	0	0	0
0	0	0	1	0	
0	0	1	0	0	
0	0	1	1	0	
0	1	0	0	0	A
0	1	0	1	1	
0	1	1	0	0	
0	1	1	1	1	
1	0	0	0	0	C
1	0	0	1	0	
1	0	1	0	1	
1	0	1	1	1	
1	1	0	0	0	6
1	1	0	1	1	
1	1	1	0	1	
1	1	1	1	0	

**Листинг 1.** Функциональная модель программируемого элемента LUT, реализующего логическую функцию D (табл. 1)

```
library IEEE;
use ieee.std_logic_1164.all;
entity LUT4_MM is -- "6CA0"
port (ADR0, ADR1, ADR2, ADR3: in std_logic;
      O : out std_logic);
end;
architecture beh_LUT4_MM is
begin
O <= (ADR0 and (not ADR1) and ADR2 and (not ADR3)) or
      (ADR0 and ADR1 and ADR2 and (not ADR3)) or
      ((not ADR0) and ADR1 and (not ADR2) and ADR3) or
      (ADR0 and ADR1 and (not ADR2) and ADR3) or
      (ADR0 and (not ADR1) and ADR2 and ADR3) or
      ((not ADR0) and ADR1 and ADR2 and ADR3);
end;
```

Если в структурном описании проекта элемент LUT используется как блок памяти (RAM), то функциональной моделью RAM является логическая функция, реализуемая в этом блоке памяти. Функциональная модель блока памяти, реализующего логическую функцию, не отличается от рассмотренной функциональной модели комбинационного элемента, реализующего ту же функцию.

### 3.2. VHDL-модель программируемого элемента LUT, используемого как сдвиговый регистр

Особенностью использования LUT как сдвигового регистра является возможность варьирования длиной регистра – длина регистра адресуется и может быть от 1 до 16 бит. Напомним, что сдвиговые регистры реализуются в LUT, которые входят в состав только секций SLICEM. Структурная схема сдвигового регистра SRLC16E в виде цепочки триггеров FF и мультиплексора MUX с четырьмя управляющими входами изображена на рис. 2 [7], функциональная синтезируемая VHDL-модель – в листинге 2. Набор <A3, A2, A1, A0> значений сигналов интерпретируется как адрес (число *i* в двоичной системе счисления), задающий длину регистра. Задание значения *i* адреса позволяет направить на выход мультиплексора MUX сигнал с выхода *i*-го триггера FF. Длина регистра задается значением параметра INIT в VHDL-модели сдвигового регистра SRLC16E (листинг 2).

**Листинг 2.** Функциональная модель LUT, реализующего сдвиговый регистр SRLC16E

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity SRLC16E is
  Generic(INIT:std_logic_vector(15 downto 0):=X"0000");
  Port (
    D : in std_logic;
    CE : in std_logic;
    CLK : in std_logic;
    A0 : in std_logic;
    A1 : in std_logic;
    A2 : in std_logic;
    A3 : in std_logic;
    Q : out std_logic;
    Q15 : out std_logic);
end SRLC16E;
architecture Behavioral of SRLC16E is
signal A : std_logic_vector (3 downto 0);
signal sig : std_logic_vector (15 downto 0) :=INIT;
begin
A <= (A3, A2, A1, A0);
process(CLK)
begin
```

```

if (CLK'event and CLK='1') then
  if (CE='1')then
    sig<=sig(14 downto 0) & D;
  end if;
end if;
end process;
Q15<=sig(15);
Q<=
sig(0) when A=X"0" ELSE
sig(1) when A=X"1" ELSE
sig(2) when A=X"2" ELSE
sig(3) when A=X"3" ELSE
sig(4) when A=X"4" ELSE
sig(5) when A=X"5" ELSE
sig(6) when A=X"6" ELSE
sig(7) when A=X"7" ELSE
sig(8) when A=X"8" ELSE
sig(9) when A=X"9" ELSE
sig(10) when A=X"A" ELSE
sig(11) when A=X"B" ELSE
sig(12) when A=X"C" ELSE
sig(13) when A=X"D" ELSE
sig(14) when A=X"E" ELSE
sig(15) when A=X"F" ;
end Behavioral;

```

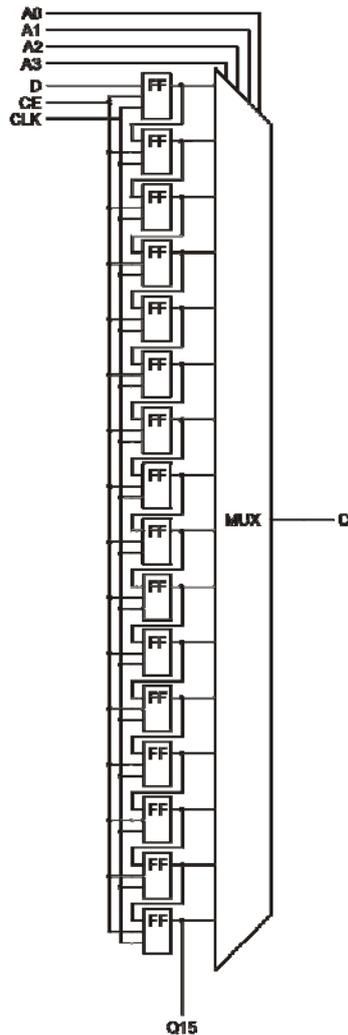


Рис. 2. Сдвиговый регистр адресуемой длины

#### 4. VHDL-модели программируемых элементов памяти

Реализация каждого настраиваемого элемента памяти FFX, FFY (см. рис. 1) в логических блоках SLICEL, SLICEM снабжается четырьмя параметрами конфигурации:

- типом элемента памяти (FF, Latch);
- типом сброса: синхронным (SYNC) либо асинхронным (ASYNC);
- установкой начального значения, значение INIT0 свидетельствует о начальной установке элемента памяти в нуль, INIT1 – в единицу;
- сбросом в единицу (SRHIGH) либо в нуль (SRLOW).

Кроме того, различные реализации элементов памяти кроме обязательных входов D (данные) и CK (в триггерах типа FF – синхросигнал, в триггерах типа Latch – сигнал разрешения) могут иметь дополнительные входы: CE – разрешение, SR – сброс (при значении SRHIGH триггер устанавливается в единицу, при SRLOW – в нуль), REV – установка в единицу (если REV используется, то сброс SR всегда SRLOW). Далее через Q будет обозначаться выход элемента памяти. Функциональная модель триггера FF с полным набором входных сигналов представлена в листинге 3.

*Листинг 3.* Функциональная модель триггера FF

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity FF is
Port
(D,CE,CK,SR,REV : in std_logic;
    Q : out std_logic);
end;
architecture Beh of FF is
signal temp : std_logic:='0'; --INIT0
begin
process (SR,REV,CK)
begin
    if (SR='1') then -- ASYNC
        temp <='0'; -- SRLOW
    elsif (REV='1') then temp <='1';
    elsif (CK'event and CK'last_value = '0' and CK='1') then
        if (CE='1') then temp<=D;
        end if;
    end if;
end process;
Q<=temp;
end Beh;
```

Функциональная модель триггера, представленная в листинге 4, реализуется в FPGA триггером типа Latch, имеющим асинхронный сброс (ASYNCR), с начальной установкой в нуль (INIT0), сбросом в нуль (SRLOW) и полным набором входных сигналов. Заметим, что при синтезе схемы в библиотеке элементов, заданной пользователем, начальные значения сигналов игнорируются, а при синтезе схемы FPGA начальные значения сигналов поддерживаются, и это нужно принимать во внимание при замене схемы FPGA заказной СБИС.

*Листинг 4.* Функциональная модель триггера типа Latch

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity Latch is
Port
(D,CE,CK,SR,REV : in std_logic;
    Q : out std_logic);
end;
architecture Beh of Latch is
signal temp : std_logic:='0';
```

```

begin
process (SR,REV,CE,CK)
begin
if SR='1' then temp<='0';
  elsif (REV='1') then temp<='1';
  elsif (CE='1') then
    if (CK='1') then temp<=D;
    end if;
  end if;
end if;
end process;
Q<=temp;
end Beh;

```

Если же в структурном описании появляются триггеры, у которых некоторые входы отсутствуют (входы D и CK триггеров являются обязательными), то используется второй подход к формированию VHDL-модели триггера. Он состоит в «уменьшении» числа входов функциональной модели триггера и исключении соответствующих операторов из VHDL-программы. Например, в табл. 2 представлены параметры, а в листинге 5 – «урезанная» модель триггера FF\_CE\_I0\_A\_L, у которого отсутствуют входы SR, REV.

Таблица 2

Входные сигналы	Имеются (+), отсутствуют (-)	Настройки	Да (+), нет (-)
D	+	INIT0	+
CK	+	INIT1	-
CE	+	ASYNC	+
SR	-	SYNC	-
REV	-	SRLOW	+
		SRHIGH	-

Листинг 4. Функциональная модель триггера FF\_CE\_I0\_A\_L

```

Entity FF_CE_I0_A_L is
  Port (D : in std_logic;
        CK : in std_logic;
        CE : in std_logic;
        Q : out std_logic );
end FF_CE_I0_A_L;
architecture Behavioral of FF_CE_I0_A_L is
  signal temp : std_logic:='0';    --INIT0
begin
  process (CK)
  begin
    if (CK'event and CK'last_value = '0' and CK='1') then
      if (CE='1') then temp<=D;
      end if;
    end if;
  end process;
  Q<=temp;

```

Заметим, что имя FF\_CE\_I0\_A\_L триггера формируется по определенным правилам, т. е. в зависимости от типа триггера, используемых входов и значений настроек. Кроме того, при конвертации FPGA-проектов требуются модели триггеров с дополнительным входным сигналом, который осуществляет начальную установку триггера в нуль либо в единицу.

С учетом того что схемные реализации элементов памяти могут различаться не только значениями параметров конфигурации, но и различным числом входов, получается довольно значительное число функциональных моделей триггеров, которые требуются для получения синтезируемых описаний проектов, реализованных в микросхемах FPGA.

## 5. Верификация моделей элементов CLB

В табл. 3 приведен список основных элементов CLB, требующихся для конвертации рассматриваемого класса проектов. Все построенные функциональные VHDL-модели элементов CLB являются моделями с нулевыми задержками: при моделировании выходные реакции любого из элементов появляются в тот же момент времени, когда изменяется хотя бы один входной сигнал этого элемента.

Таблица 3

Имя элемента	Тип элемента		Функция
	Комбинационный элемент	Элемент памяти	
XOR2	+		Сумма по модулю 2
AND2	+		Конъюнкция
MUX2	+		Мультиплексор
INV	+		Инвертор
BUF, OBUF	+		Передача сигнала
MULT18X18	+		Умножитель
FF		+	Триггер
LATCHE		+	Триггер-защелка
SRLC16E		+	Сдвиговый регистр
RAMD16		+	Блок памяти

Функциональные модели элементов CLB проверены при конвертации 30 FPGA-проектов в синтезируемые VHDL-описания. Исходные описания реализованных на FPGA проектов представляли собой либо алгоритмические VHDL-описания, либо логические схемы в графическом редакторе системы ISE, либо созданные в системе State-CAD [1] графические описания конечных автоматов. Проверка правильности VHDL-моделей элементов CLB осуществлялась на основе сравнения результатов моделирования исходных FPGA-проектов и конвертированных VHDL-описаний проектов. Для тех проектов, которые были представлены в виде VHDL-описаний, функциональная эквивалентность исходного и конвертированного проектов была подтверждена в системе FormalPro формальной верификации фирмы Mentor Graphics [8]. Проверка свойства синтезируемости для конвертированных VHDL-проектов осуществлялась путем синтеза логических схем в целевых библиотеках проектирования заказных СБИС. В качестве основной целевой библиотеки была использована библиотека логических элементов, приведенная в [1, с. 342]. Для всех проектов было достигнуто одинаковое поведение (по результатам окончания тактов моделирования) исходного проекта и конвертированного. Таким образом, конвертация обеспечивает функциональную эквивалентность исходного FPGA-проекта и конвертированного проекта. Однако величины длины такта (тактовая частота) для исходного и реализованного на заказной СБИС конвертированного проектов различаются. Это обусловлено технологической базой: FPGA семейства Spartan 3 реализуются по своей технологии (90 нм SRAM КМОП), а реализация заказной СБИС может быть выполнена по другой технологии, например по отечественной КМОП-технологии и более «крупным» технологическим нормам.

## Заключение

Предложенные VHDL-модели элементов конфигурируемых логических блоков микросхем FPGA семейства Spartan 3 обеспечивают поддержку автоматизированного перепроектирования FPGA на заказные СБИС. Получение синтезируемых VHDL-описаний позволяет применять промышленные системы проектирования на базе VHDL – проводить моделирование, сравнивать поведение исходного и конвертированного проектов, синтезировать логические схемы, размещаемые в составе заказных СБИС.

**Список литературы**

1. Бибилло, П.Н. Системы проектирования интегральных схем на основе языка VHDL. StateCAD, ModelSim, LeonardoSpectrum / П.Н. Бибилло. – М. : СОЛОН-Пресс, 2005. – 384 с.
2. Зотов, Ю.В. Проектирование цифровых устройств на основе ПЛИС фирмы XILINX в САПР WebPack ISE / Ю.В. Зотов. – М. : Горячая линия – Телеком, 2003. – 624 с.
3. An XDL-based busmacro generator for customizable communication interfaces for dynamically and partially reconfigurable systems / C. Claus [et al.] // Field Programmable Logic and Applications (FPL2010) : 20th Intern. Conf. – Milano, 2010.
4. Кузелин, О.М. Современные семейства ПЛИС фирмы Xilinx : справочное пособие / О.М. Кузелин, Д.А. Кнышев, Ю.В. Зотов. – М. : Горячая линия – Телеком, 2004. – 440 с.
5. Spartan-3 Libraries Guide for HDL Designs [Electronic resource]. – Mode of access : [http://www.xilinx.com/itp/xilinx10/books/docs/spartan3\\_hdl/spartan3\\_hdl.pdf](http://www.xilinx.com/itp/xilinx10/books/docs/spartan3_hdl/spartan3_hdl.pdf). – Date of access : 26.03.2012.
6. Бибилло, П.Н. О несинтезируемых конструкциях языка VHDL / П.Н. Бибилло // Современная электроника. – 2008. – № 5. – С. 68–71.
7. Using Look-Up Tables as Shift Registers (SRL16) in Spartan-3 Generation FPGAs [Electronic resource]. – Mode of access : [http://www.xilinx.com/support/documentation/application\\_notes/xapp465.pdf](http://www.xilinx.com/support/documentation/application_notes/xapp465.pdf). – Date of access : 26.03.2012.
8. Лохов, А. Обзор средств функциональной верификации компании Mentor Graphics / А. Лохов // Современная электроника. – 2005. – № 5. – С. 50–54.

Поступила 27.01.12

*Объединенный институт  
проблем информатики НАН Беларуси,  
Минск, Сурганова, 6  
e-mail: bibilo@newman.bas-net.by*

**P.N. Bibilo, A.L. Soloviev**

**FUNCTIONAL VHDL-MODELS OF FPGA-ELEMENTS  
OF SPARTAN 3 FAMILY FOR CONVERSION OF PROJECTS  
OF DIGITAL SYSTEMS INTO ASIC**

Based on the structure of FPGA xc3s1000-4ft256 and experimental study of element functions, the synthesized VHDL models of FPGA elements of Spartan 3 family are proposed for a conversion of digital systems projects into ASIC. The need for such models appears when converting projects of digital systems implemented in FPGA into the projects suitable for a synthesis of ASIC design.

УДК 519.714

Д.Я. Новиков, Л.Д. Черемисинова

## ИМПЛИКАТИВНЫЙ МЕТОД АНАЛИЗА ЧАСТИЧНЫХ БУЛЕВЫХ ФУНКЦИЙ

*Рассматривается задача определения, находятся ли системы частичных булевых функций в отношении эквивалентности или реализации. Предлагается метод решения задачи путем сведения ее к проверке выполнимости конъюнктивной нормальной формы, в основе формирования которой лежит использование отношения импликации.*

### Введение

Проектирование цифровых СБИС представляет собой многоэтапный процесс оптимизации и преобразований проектных решений, начиная с исходного описания на одном из входных языков проектирования и заканчивая схемой в целевом технологическом базисе. Проектные решения могут быть получены как автоматически (с использованием программных средств синтеза и оптимизации), так и полуавтоматически (путем корректировки проектного решения человеком). В любом случае во избежание распространения ошибки, допущенной на одном из ранних этапов проектирования, до стадии изготовления схемы необходимо верифицировать [1, 2] полученные решения – проверять, находятся ли они в отношении эквивалентности или реализации.

В настоящей работе рассматривается задача проверки отношений эквивалентности и реализуемости между системами частичных булевых функций, которая является одной из базовых задач верификации логических описаний с функциональной неопределенностью [3]. Предлагаемый метод решения задачи анализа булевых функций на эквивалентность и реализуемость основан на сведении ее к проверке выполнимости конъюнктивной нормальной формы (КНФ). Анализируемая КНФ строится в виде конъюнкции двух КНФ: разрешения для одной системы частичных булевых функций и запрета – для другой. Первая КНФ задает допустимые комбинации значений всех переменных для одной из анализируемых систем, вторая – запрещенные комбинации для другой. Формирование этих КНФ основано на использовании отношения импликации между наборами значений входных переменных и соответствующими им значениями функций системы.

### 1. Основные определения

*Частичная булева функция* (ЧБФ)  $f(\mathbf{x})$ , где  $\mathbf{x} = (x_1, x_2, \dots, x_n)$  – вектор значений аргументов функции, задается множествами  $U_f^0$ ,  $U_f^1$  и  $U_f^{ds}$  интервалов (или элементов) булева пространства  $E^n$ , на которых она принимает соответственно нулевое, единичное и неопределенное значения, при этом  $U_f^1 \cup U_f^0 \cup U_f^{ds} = E^n$ . Интервал ранга  $k$  фиксирует значения  $k$  переменных и покрывает  $2^{n-k}$  элементов булева пространства, его можно представить также в виде конъюнкции  $k$  литералов (под литералом понимается булева переменная или ее отрицание). В общем случае элементарная конъюнкция  $k_j$  и соответствующий ей интервал покрывает (или поглощает) элементарную конъюнкцию  $k_i$  (и соответствующий ей интервал), если множество литералов из  $k_j$  является подмножеством литералов из  $k_i$ .

Будем задавать систему ЧБФ  $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_m(\mathbf{x}))$  множеством многовыходных интервалов  $(\mathbf{u}^f, \mathbf{t}^f)$ , представляемых парами троичных векторов длины  $n$  и  $m$ . Входная часть  $\mathbf{u}^f$  является интервалом на множестве  $x_i \in \mathbf{x}$ , выходная часть  $\mathbf{t}^f$  – троичным вектором значений функций на интервале  $\mathbf{u}^f$ . Векторы  $\mathbf{u}^f$  и  $\mathbf{t}^f$  могут представляться также как конъюнкции литералов некоторых  $x_i \in \mathbf{x}$  и  $f_i \in \mathbf{f}(\mathbf{x})$ . В частном случае входная часть  $\mathbf{u}^f$  может представлять собой булев вектор  $\mathbf{v}^f$ , тогда многовыходной интервал  $(\mathbf{u}^f, \mathbf{t}^f)$  называется многовыходным набором  $(\mathbf{v}^f, \mathbf{t}^f)$ .

Для каждой функции  $f_j \in \mathbf{f}(\mathbf{x})$  справедливо: если  $j$ -я компонента вектора  $\mathbf{t}^f$  есть 1 или 0, то все элементы интервала  $\mathbf{u}^f$  принадлежат множеству  $U_{f_j}^1$  единичных значений функции  $f_j$  или

соответственно множеству  $U_f^0$  нулевых значений функции  $f_j$ ; если же значение  $j$ -й компоненты вектора  $\mathbf{t}^f$  не определено, то либо значение функции  $f_j$  не определено на всем интервале  $\mathbf{u}^f$ , либо  $f_j$  принимает разные значения на разных наборах из интервала  $\mathbf{u}^f$ .

Далее рассматриваются только непротиворечивые системы ЧБФ. Система булевых функций, заданная множеством многовыходных интервалов  $(\mathbf{u}_i^f, \mathbf{t}_i^f)$ , непротиворечива, если для всех пар  $(\mathbf{u}_i^f, \mathbf{t}_i^f)$  и  $(\mathbf{u}_j^f, \mathbf{t}_j^f)$ , таких, что  $\mathbf{u}_i^f \wedge \mathbf{u}_j^f \neq 0$ , имеет место  $\mathbf{t}_i^f \wedge \mathbf{t}_j^f \neq 0$ .

Система ЧБФ  $\mathbf{f}(\mathbf{x})$ , заданная множеством  $S^f$  многовыходных интервалов, в матричном виде может быть представлена парой трюичных матриц  $\mathbf{U}^f$  и  $\mathbf{T}^f$ , задающих своими строками входные и выходные части многовыходных интервалов из  $S^f$ . Например, если  $S^f = \{(x_1 x_2 \bar{x}_3, y_1 \bar{y}_3), (\bar{x}_1 x_3 x_5 \bar{x}_6, \bar{y}_1 y_3), (\bar{x}_1 \bar{x}_2 x_4 \bar{x}_5, \bar{y}_1 \bar{y}_2), (x_2 \bar{x}_3 x_4 x_6, y_1 \bar{y}_3), (x_1 \bar{x}_2 \bar{x}_5, \bar{y}_1 y_3), (x_1 \bar{x}_2 \bar{x}_4 x_6, y_2), (x_2 x_3 x_4 x_6, y_1 y_2)\}$ , система  $\mathbf{f}(\mathbf{x})$  представляется как

$$\mathbf{U}^f = \begin{matrix} & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \\ \begin{matrix} 1 \\ 0 \\ 0 \\ -1 \\ 1 \\ 1 \\ -1 \end{matrix} & \begin{matrix} 1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{matrix} & \begin{matrix} 0 \\ -1 \\ 0 \\ 1 \\ -1 \\ -1 \\ 0 \end{matrix} & \begin{matrix} - \\ - \\ - \\ 1 \\ - \\ 0 \\ 0 \end{matrix} & \begin{matrix} - \\ - \\ 0 \\ - \\ - \\ - \\ - \end{matrix} & \begin{matrix} - \\ 0 \\ 1 \\ - \\ 0 \\ - \\ 0 \end{matrix} & \begin{matrix} - \\ 0 \\ - \\ 1 \\ - \\ - \\ - \end{matrix} \end{matrix}; \quad \mathbf{T}^f = \begin{matrix} & y_1 & y_2 & y_3 \\ \begin{matrix} 1 \\ - \\ 0 \\ 1 \\ 0 \\ - \\ 1 \end{matrix} & \begin{matrix} - \\ 0 \\ 0 \\ - \\ - \\ 1 \\ 1 \end{matrix} & \begin{matrix} 0 \\ 1 \\ - \\ 0 \\ - \\ - \\ 1 \end{matrix} & \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{matrix} \end{matrix}. \quad (1)$$

## 2. Отношения между частичными булевыми функциями

Булева функция  $g(\mathbf{x})$  (полностью определенная или частичная) реализует частичную булеву функцию  $f(\mathbf{x})$  (обозначается как  $f(\mathbf{x}) \leq g(\mathbf{x})$ ) тогда и только тогда, когда

$$U_f^1 \subseteq U_g^1, U_f^0 \subseteq U_g^0. \quad (2)$$

Система булевых функций  $\mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), g_2(\mathbf{x}), \dots, g_m(\mathbf{x}))$  реализует систему ЧБФ  $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_m(\mathbf{x}))$ , если для всех  $i$ , где  $1 \leq i \leq m$ , имеет место  $f_i(\mathbf{x}) \leq g_i(\mathbf{x})$ .

Выполнение условия (2) гарантирует, что на области определения системы ЧБФ  $\mathbf{f}(\mathbf{x})$  (менее определенной) система  $\mathbf{g}(\mathbf{x})$  имеет ту же функциональность, что и  $\mathbf{f}(\mathbf{x})$ , т. е. для любого двоичного набора значений переменных  $\mathbf{x}$  вектор значений функций  $\mathbf{g}(\mathbf{x})$  должен поглощаться вектором значений функций  $\mathbf{f}(\mathbf{x})$ .

ЧБФ  $g(\mathbf{x})$  эквивалентна ЧБФ  $f(\mathbf{x})$  (обозначается как  $f(\mathbf{x}) \equiv g(\mathbf{x})$ ) тогда и только тогда, когда  $U_f^1 = U_g^1$  и  $U_f^0 = U_g^0$ .

Система ЧБФ  $\mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), g_2(\mathbf{x}), \dots, g_m(\mathbf{x}))$  эквивалентна системе ЧБФ  $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_m(\mathbf{x}))$ , если для всех  $i$ , где  $1 \leq i \leq m$ , имеет место  $f_i(\mathbf{x}) \equiv g_i(\mathbf{x})$ .

Условие эквивалентности булевых функций можно перефразировать также следующим образом: ЧБФ  $g(\mathbf{x})$  эквивалентна ЧБФ  $f(\mathbf{x})$  тогда и только тогда, когда  $f(\mathbf{x}) \leq g(\mathbf{x})$  и  $g(\mathbf{x}) \leq f(\mathbf{x})$ , что эквивалентно  $U_f^1 \subseteq U_g^1, U_f^0 \subseteq U_g^0, U_g^1 \subseteq U_f^1, U_g^0 \subseteq U_f^0$ .

Следовательно, для того чтобы проверить отношение эквивалентности, достаточно проверить отношение реализуемости в обе стороны.

Для того чтобы проверить, реализуется ли система ЧБФ  $\mathbf{f}(\mathbf{x})$  системой  $\mathbf{g}(\mathbf{x})$ , необходимо проверить выполнение условия (2) всех функций  $f_i \in \mathbf{f}(\mathbf{x}), g_i \in \mathbf{g}(\mathbf{x})$ . Процедуру проверки реализуемости можно упростить, рассматривая одновременно все функции системы ЧБФ. Для этого все многовыходные интервалы  $(\mathbf{u}^f, \mathbf{t}^f)$  системы  $\mathbf{f}(\mathbf{x})$  расщепляются на многовыходные наборы  $(\mathbf{v}^f, \mathbf{t}^f)$ , причем расщеплению подвергаются входные части многовыходных интервалов, а выходные их части не меняются. В результате расщепления получаем систему  $\mathbf{f}(\mathbf{x})$ , заданную многовыходными наборами  $(\mathbf{v}_i^f, \mathbf{t}_i^f)$ . Система ЧБФ  $\mathbf{f}(\mathbf{x})$  реализуется системой ЧБФ  $\mathbf{g}(\mathbf{x})$ , если для каждого многовыходного набора  $(\mathbf{v}_i^f, \mathbf{t}_i^f)$  выполняются следующие условия: для каждой определенной компоненты  $t_i^{fk}$  вектора  $\mathbf{t}_i^f$  найдется такой многовыходной интервал  $(\mathbf{u}_j^g, \mathbf{t}_j^g)$ , у которого соответствующая компонента  $t_j^{gk}$  выходной части  $\mathbf{t}_j^g$  равна  $t_i^{fk}$ , а входная часть  $\mathbf{u}_j^g$  поглощает входную часть  $\mathbf{v}_i^f$ .

Например, система ЧБФ  $f(x)$  (1) реализуется следующей системой  $g(x)$ :

$$\begin{array}{cccccc}
 x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & & y_1 & y_2 & y_3 \\
 1 & 1 & 0 & - & - & - & & 1 & - & 0 & 1 \\
 0 & - & - & - & 1 & 0 & & - & 0 & - & 2 \\
 - & 0 & - & 1 & 0 & - & & 0 & 0 & 1 & 3 \\
 \mathbf{U}^g = & 1 & 0 & - & - & 1 & ; & \mathbf{T}^g = & 1 & 1 & 0 & 4 \\
 1 & 0 & - & - & 0 & - & & 0 & - & - & 5 \\
 1 & 0 & - & 0 & - & - & & - & 1 & 1 & 6 \\
 - & - & 1 & - & 1 & 0 & & - & - & 1 & 7
 \end{array} \quad (3)$$

Ясно, что такой метод проверки реализуемости может быть применим на практике только в том случае, если входные части многовыходных интервалов системы ЧБФ  $f(x)$  заданы небольшими интервалами, когда число  $p$  неопределенных компонент соответствующих векторов невелико. Например, уже при  $p = 10$  каждый интервал расщепляется на  $2^{10} = 1024$  набора. Для случаев, когда  $p$  относительно велико, предлагается использовать описанный ниже метод на основе сведения к задаче проверки выполнимости КНФ.

### 3. КНФ разрешения системы частичных булевых функций

КНФ представляет полностью определенную булеву функцию, заданную в виде конъюнкции одного или более дизъюнктов, каждый из которых представляет собой дизъюнкцию одного или более литералов (соответствующих разным переменным или их инверсиям). Задача проверки выполнимости КНФ заключается в нахождении такого присваивания значений переменным из  $x$  (может быть, и не всем), которое обращает КНФ в единицу. Если такое присваивание удастся найти, то говорят, что КНФ выполнима и полученное присваивание называют выполняющим эту КНФ, иначе КНФ невыполнима. Невыполнимая КНФ представляет функцию, тождественно равную нулю.

Матричный вид КНФ, имеющей  $k$  дизъюнктов и  $n$  переменных, представляется троичной матрицей  $C$ , строки которой задают дизъюнкты, а столбцы соответствуют переменным. Элемент  $c_i^j$  на пересечении  $i$ -й строки и  $j$ -го столбца матрицы  $C$  принимает значение 1, 0 или «-» в зависимости от формы ( $x_j$  или  $\bar{x}_j$ ) переменной  $x_j$  или ее отсутствия в  $i$ -м дизъюнкте КНФ.

Исходя из определения ЧБФ, всякий ее многовыходной интервал  $s_i^g = (u_i^g, t_i^g)$  задает импликативное условие  $u_i^g \rightarrow t_i^g$ : при присваивании переменным  $x_j$  значений, обращающих  $u_i^g$  в 1, функции системы  $g(x)$  принимают значения, обращающие  $t_i^g$  в 1.

При верификации комбинационных схем, основанной на решении задачи проверки выполнимости КНФ, строится так называемая КНФ разрешения для каждой из сравниваемых комбинационных схем [2], которая задает все возможные допустимые комбинации сигналов на всех полюсах схемы. Аналогично можно построить КНФ разрешения системы ЧБФ  $g(x)$ , которая будет задавать все возможные допустимые комбинации ее переменных (аргументов и функций). Нетрудно доказать, что для непротиворечивой системы ЧБФ справедливо следующее утверждение.

Утверждение 1. КНФ разрешения системы ЧБФ  $g(x)$ , заданной множеством многовыходных интервалов  $s_i^g = (u_i^g, t_i^g)$ ,  $i = 1, 2, \dots, r$ , порождается формулой

$$C^g = (u_1^g \rightarrow t_1^g)(u_2^g \rightarrow t_2^g) \dots (u_r^g \rightarrow t_r^g).$$

КНФ  $C_i$  разрешения, порожаемая многовыходным интервалом  $s_i = (u_i, t_i)$  с  $u_i = x_1^i x_2^i \dots x_{ni}^i$  и  $t_i = y_1^i y_2^i \dots y_{mi}^i$ , может быть получена преобразованием  $u_i \rightarrow t_i$  в булев базис следующим образом:

$$\begin{aligned}
 C_i &= (u_i \rightarrow t_i) = \bar{u}_i \vee t_i = \bar{x}_1^i \vee \bar{x}_2^i \vee \dots \vee \bar{x}_{ni}^i \vee y_1^i y_2^i \dots y_{mi}^i = \\
 &= (\bar{x}_1^i \vee \bar{x}_2^i \vee \dots \vee \bar{x}_{ni}^i \vee y_1^i) (\bar{x}_1^i \vee \bar{x}_2^i \vee \dots \vee \bar{x}_{ni}^i \vee y_2^i) \dots (\bar{x}_1^i \vee \bar{x}_2^i \vee \dots \vee \bar{x}_{ni}^i \vee y_{mi}^i).
 \end{aligned}$$

КНФ разрешения  $C_i$  состоит из  $m_i$  дизъюнктов. Например, КНФ разрешения многовыходного интервала  $s_1 = (x_1 x_2 \bar{x}_3, y_1 \bar{y}_3)$  системы ЧБФ  $g(x)$  (3) состоит из двух дизъюнктов:

$C_1^g = (\bar{x}_1 \vee \bar{x}_2 \vee x_3 \vee y_1)(\bar{x}_1 \vee \bar{x}_2 \vee x_3 \vee \bar{y}_3)$ , а КНФ разрешения  $C^g$  всей системы ЧБФ  $g(x)$  (3) в матричной форме имеет следующий вид:

$$C^g = \begin{matrix} & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & y_1 & y_2 & y_3 \\ \begin{matrix} 0 & 0 & 1 & - & - & - & 1 & - & - & 1 \\ 0 & 0 & 1 & - & - & - & - & - & 0 & 2 \\ 1 & - & - & - & 0 & 1 & - & 0 & - & 3 \\ - & 1 & - & 0 & 1 & - & 0 & - & - & 4 \\ - & 1 & - & 0 & 1 & - & - & 0 & - & 5 \\ - & 1 & - & 0 & 1 & - & - & - & 1 & 6 \\ - & 0 & 1 & - & - & 0 & 1 & - & - & 7 \\ - & 0 & 1 & - & - & 0 & - & 1 & - & 8 \\ - & 0 & 1 & - & - & 0 & - & - & 0 & 9 \\ 0 & 1 & - & - & 1 & - & 0 & - & - & 10 \\ 0 & 1 & - & 1 & - & - & - & 1 & - & 11 \\ 0 & 1 & - & 1 & - & - & - & - & 1 & 12 \\ - & - & 0 & - & 0 & 1 & - & - & 1 & 13 \end{matrix} \end{matrix} . \quad (4)$$

Утверждение 2. Система булевых функций  $g(x)$  реализует систему ЧБФ  $f(x)$ , если любой набор значений переменных (аргументов и функций), выполняющий КНФ разрешения  $C^f$ , является выполняющим и для КНФ разрешения  $C^g$ .

Очевидно, что проверка отношения реализуемости на основе этого утверждения несколько не проще проверки на основе определения (2). Ниже предлагается формализованный метод проверки отношения реализуемости, основанный на следующем утверждении.

Утверждение 3. Система булевых функций  $g(x)$  реализует систему ЧБФ  $f(x)$ , если любой набор значений переменных (аргументов и функций), не выполняющий КНФ разрешения  $C^f$ , не является выполняющим и для КНФ разрешения  $C^g$ .

#### 4. КНФ запрета системы частичных булевых функций

КНФ  $P^f$  запрета исходной системы ЧБФ  $f(x)$ , представленной множеством  $S^f$  многовыходных интервалов, задает все возможные недопустимые комбинации ее переменных (аргументов и функций). Эта КНФ выполнима, если нарушается условие  $u_i^f \rightarrow t_i^f$ , заданное хотя бы одним многовыходным интервалом из  $S^f$ . Соответственно КНФ запрета  $P_i$  многовыходного интервала  $s_i = (u_i, t_i)$ , где  $u_i = x_1^i x_2^i \dots x_{n_i}^i$  и  $t_i = y_1^i y_2^i \dots y_{m_i}^i$ , может быть получена преобразованием  $\bar{s}_i$ , задающим условие его нарушения, в булев базис следующим образом:

$$P_i = \bar{s}_i = \neg(u_i \rightarrow t_i) = u_i \bar{t}_i = x_1^i x_2^i \dots x_{n_i}^i (\bar{y}_1^i \vee \bar{y}_2^i \vee \dots \vee \bar{y}_{m_i}^i).$$

КНФ запрета  $P_i$  состоит из  $n_i + 1$  дизъюнктов. Например, КНФ запрета многовыходного интервала  $s_2^f = (x_1 x_2, f_1 \bar{f}_2)$  системы ЧБФ  $f(x)$  (1) состоит из трех дизъюнктов:  $P_2^f = (x_1)(x_2)(\bar{f}_1 \vee f_2)$ .

Утверждение 4. Функция запрета  $F^f$  системы ЧБФ  $f(x)$ , заданной множеством многовыходных интервалов  $s_i^f = (u_i^f, t_i^f)$ ,  $i = 1, 2, \dots, r$ , имеет вид

$$F^f = P_1^f \vee P_2^f \vee \dots \vee P_r^f = (u_1^f \bar{t}_1^f) \vee (u_2^f \bar{t}_2^f) \vee \dots \vee (u_r^f \bar{t}_r^f). \quad (5)$$

Все возможные недопустимые комбинации переменных функций системы ЧБФ  $f(x)$  представляются выполняющими формулу (5) наборами. Чтобы применить SAT-решатель [4–6] (программу проверки выполнимости КНФ) для их нахождения, необходимо формулу (5) преобразовать к виду КНФ, теоретически это всегда возможно, однако это NP-трудная задача. В работе [7] приведен метод линейной сложности для построения КНФ запрета системы ЧБФ путем кодирования составляющих ее КНФ запрета  $P_i^f$  кодами единичной длины. Метод заключается во введении  $r$  (по числу многовыходных интервалов) булевых переменных  $w_i$ , предназначенных для кодирования условий  $P_i^f$  нарушения реализуемости многовыходных интервалов. Каждое условие  $P_i^f$  кодируется кодом длины  $1 - \bar{w}_i$  и добавляется дизъюнкт, реализующий функцию выбора [8], которая для единичного кодирования имеет вид  $(w_1 \vee w_2 \vee \dots \vee w_r)$ . Таким образом, функция запрета  $F^f = P_1^f \vee P_2^f \vee \dots \vee P_r^f$  преобразуется к выражению

$$F^f = (P_1^f \vee \bar{w}_1) (P_2^f \vee \bar{w}_1) \dots (P_r^f \vee \bar{w}_r) (w_1 \vee w_2 \vee \dots \vee w_r) =$$

$$= (\mathbf{u}_1^f \bar{\mathbf{t}}_1^f \vee \bar{w}_1) (\mathbf{u}_2^f \bar{\mathbf{t}}_2^f \vee \bar{w}_2) \dots (\mathbf{u}_r^f \bar{\mathbf{t}}_r^f \vee \bar{w}_r) (w_1 \vee w_2 \vee \dots \vee w_r). \quad (6)$$

Формула (6) выполнима тогда и только тогда, когда выполнима формула (5).

Приняв во внимание, что  $\mathbf{u}_i^f = x_1^i x_2^i \dots x_{ni}^i$  и  $\bar{\mathbf{t}}_i^f = y_1^i y_2^i \dots y_{mi}^i$  в  $(\mathbf{u}_i^f, \bar{\mathbf{t}}_i^f)$ , каждый из первых  $r$  дизъюнктивных членов (6) преобразуется к КНФ

$$\mathbf{u}_i^f \bar{\mathbf{t}}_i^f \vee \bar{w}_i = (x_1^i \vee \bar{w}_i)(x_2^i \vee \bar{w}_i) \dots (x_{ni}^i \vee \bar{w}_i)(\bar{y}_1^i \vee \bar{y}_2^i \vee \dots \vee \bar{y}_{mi}^i \vee \bar{w}_i).$$

Соответственно функция запрета  $F^f$  представляется в виде КНФ запрета  $P^f$ . Например, КНФ запрета системы ЧБФ  $f(\mathbf{x})$  (1) в матричной форме имеет следующий вид:

	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$	$y_1$	$y_2$	$y_3$	$w_1$	$w_2$	$w_3$	$w_4$	$w_5$	$w_6$	$w_7$	
	1	-	-	-	-	-	-	-	-	0	-	-	-	-	-	-	1
	-	1	-	-	-	-	-	-	-	0	-	-	-	-	-	-	2
	-	-	0	-	-	-	-	-	-	0	-	-	-	-	-	-	3
	-	-	-	-	-	-	0	-	1	0	-	-	-	-	-	-	4
	0	-	-	-	-	-	-	-	-	0	-	-	-	-	-	-	5
	-	-	1	-	-	-	-	-	-	0	-	-	-	-	-	-	6
	-	-	-	1	-	-	-	-	-	0	-	-	-	-	-	-	7
	-	-	-	-	0	-	-	-	-	0	-	-	-	-	-	-	8
	-	-	-	-	-	-	1	0	-	0	-	-	-	-	-	-	9
	0	-	-	-	-	-	-	-	-	0	-	-	-	-	-	-	10
	-	0	-	-	-	-	-	-	-	0	-	-	-	-	-	-	11
	-	-	-	1	-	-	-	-	-	0	-	-	-	-	-	-	12
	-	-	-	-	0	-	-	-	-	0	-	-	-	-	-	-	13
	-	-	-	-	-	1	1	-	-	0	-	-	-	-	-	-	14
	-	1	-	-	-	-	-	-	-	0	-	-	-	-	-	-	15
	-	-	0	-	-	-	-	-	-	0	-	-	-	-	-	-	16
$P^f =$	-	-	-	1	-	-	-	-	-	0	-	-	-	-	-	-	17
	-	-	-	-	1	-	-	-	-	0	-	-	-	-	-	-	18
	-	-	-	-	-	0	-	1	-	-	-	0	-	-	-	-	19
	1	-	-	-	-	-	-	-	-	-	-	0	-	-	-	-	20
	-	0	-	-	-	-	-	-	-	-	-	0	-	-	-	-	21
	-	-	-	0	-	-	-	-	-	-	-	0	-	-	-	-	22
	-	-	-	-	-	1	-	0	-	-	-	0	-	-	-	-	23
	1	-	-	-	-	-	-	-	-	-	-	0	-	-	-	-	24
	-	0	-	-	-	-	-	-	-	-	-	0	-	-	-	-	25
	-	-	-	0	-	-	-	-	-	-	-	0	-	-	-	-	26
	-	-	-	-	1	-	-	-	-	-	-	0	-	-	-	-	27
	-	-	-	-	-	0	-	-	-	-	-	0	-	-	-	-	28
	-	1	-	-	-	-	-	-	-	-	-	0	-	-	-	-	29
	-	-	0	-	-	-	-	-	-	-	-	0	-	-	-	-	30
	-	-	-	0	-	-	-	-	-	-	-	0	-	-	-	-	31
	-	-	-	-	1	-	-	-	-	-	-	0	-	-	-	-	32
	-	-	-	-	-	0	0	-	-	-	-	0	-	-	-	-	33
	-	-	-	-	-	-	-	1	1	1	1	1	1	1	1	1	34

**5. Сведение задачи проверки отношений между системами ЧБФ к проверке выполнимости КНФ**

Суть предлагаемого метода проверки, реализуется ли система ЧБФ  $f(\mathbf{x})$  системой  $g(\mathbf{x})$ , состоит в следующем. Строится КНФ запрета  $P^f$  системы ЧБФ  $f(\mathbf{x})$  и КНФ разрешения  $C^g$  системы  $g(\mathbf{x})$ , затем эти КНФ объединяются в одну  $C^g \wedge P^f$ .

Утверждение 5. Система булевых функций  $g(\mathbf{x})$  реализует систему ЧБФ  $f(\mathbf{x})$ , если и только если КНФ  $C^g \wedge P^f$  невыполнима.

КНФ  $C^g \wedge P^f$ , получаемая объединением КНФ разрешения (4) и запрета (7), невыполнима; следовательно, система ЧБФ  $g(\mathbf{x})$  реализует систему  $f(\mathbf{x})$ . Для того чтобы проверить, реализует ли система ЧБФ  $f(\mathbf{x})$  систему  $g(\mathbf{x})$ , необходимо аналогичным образом построить КНФ разрешения  $C^f$  для системы ЧБФ  $f(\mathbf{x})$  и КНФ запрета  $P^g$  для системы  $g(\mathbf{x})$ :



Для оценки быстродействия предлагаемого метода проверки реализуемости между системами ЧБФ был решен ряд примеров, представляющих пары систем ЧБФ, сгенерированных псевдослучайным образом [9]. Генерировались системы ЧБФ, заведомо находившиеся в отношении реализуемости, так как в этих случаях вопрос о реализуемости сводился к проверке невыполнимости (см. утверждение 6) КНФ, на что требовался максимум времени работы SAT-решателя.

Программа, реализующая метод, была написана на языке C++, в качестве SAT-решателя использовалась известная программа MiniSat [4]. Исследования проводились на ПЭВМ Pentium IV 2400 МГц. Число аргументов систем ЧБФ варьировалось от 20 до 40, функций – от 20 до 50, интервалов области определения системы – от 1000 до 18 000. Время работы программы изменялось от 0,5 с (для простых примеров) до 2000 с (для наиболее сложных примеров).

### Заключение

Предложенный в работе метод проверки отношений эквивалентности и реализуемости между системами частичных булевых функций сводит рассматриваемую задачу к задаче проверки выполнимости КНФ и основан на использовании отношения импликации для формирования КНФ разрешения одной из сравниваемых систем и КНФ запрета другой. Следует заметить, что метод может быть применим и в том случае, когда функции сравниваемых систем полностью определены, при условии, что области их определения заданы в явном виде многовыходными интервалами.

Нетрудно показать, что размер формируемой предложенным методом КНФ, для которой решается задача выполнимости, полиномиально зависит от размерностей сравниваемых систем функций. Основное же время при проверке отношений эквивалентности и реализуемости между системами ЧБФ приходится на установление выполнимости полученной КНФ.

### Список литературы

1. Kuehlmann, A. Combinational and Sequential Equivalence Checking / A. Kuehlmann, A.J. Cornelis van Eijk // *Logic synthesis and Verification* ; ed. S. Hassoun, T. Sasao, R.K. Brayton. – Kluwer Academic Publishers, 2002. – P. 343–372.
2. *Advanced Formal Verification* / R. Drechsler [et al.]. – Boston, Dordrecht, London : Kluwer Academic Publishers, 2005. – 249 p.
3. Новиков, Д.Я. Программный комплекс для верификации комбинационных устройств в процессе логического проектирования / Д.Я. Новиков, Л.Д. Черемисинова // Четвертый Белорусский космический конгресс : Материалы конгресса, Минск, 25–27 октября 2011 г. : в 2 т. / ОИПИ НАН Беларуси ; редкол.: А.В. Тузиков [и др.]. – Минск, 2011. – Т. 2. – С. 289–293.
4. Sorensson, N. MiniSat v1.13 A SAT Solver with Conflict-Clause Minimization / N. Sorensson, N. Een // *The International Conference on Theory and Applications of Satisfiability Testing (SAT 2005)* [Electronic resource]. – 2005. – Mode of access : [http://www.lri.fr/~simon/contest/results/descriptions/solvers/minisat\\_static.pdf](http://www.lri.fr/~simon/contest/results/descriptions/solvers/minisat_static.pdf). – Date of access : 16.02.2009.
5. Kunz, W. SAT and ATPG: Algorithms for Boolean Decision Problems / W. Kunz, J. Marques-Silva, S. Malik // *Logic synthesis and Verification* ; ed. S. Hassoun, T. Sasao and R.K. Brayton. – Kluwer Academic Publishers, 2002. – P. 309–341.
6. Goldberg, E. BerkMin: A fast and robust SAT-Solver / E. Goldberg, Y. Novikov // *European Design and Test Conference (DATE'02)* : Proc. of the conf., Paris, 4–8 March 2002 / Le Palais des Congres. – Paris, 2002. – P. 142–149.
7. Cheremisinova, L. SAT-Based Approach to Verification of Logical Descriptions with Functional Indeterminacy / L. Cheremisinova, D. Novikov // *8th Intern. Workshop on Boolean Problems* : Proc. of the Workshop, Freiberg, 18–19 September 2008 / Techn. univ. Bergakademie ; ed. by V. Steinbach. – Freiberg, 2008. – P. 59–66.
8. Черемисинова, Л.Д. Формальная верификация описаний с функциональной неопределенностью на основе проверки выполнимости конъюнктивной нормальной формы / Л.Д. Черемисинова, Д.Я. Новиков // *Автоматика и вычислительная техника*. – 2010. – № 1. – С. 5–16.

9. Закревский, А.Д. Генераторы псевдослучайных логико-комбинаторных объектов в C++ / А.Д. Закревский, Н.Р. Торопов // Логическое проектирование. – Минск : Ин-т техн. кибернетики НАН Беларуси, 1999. – Вып. 4. – С. 49–63.

Поступила 01.03.12

*Объединенный институт проблем  
информатики НАН Беларуси,  
Минск, Сурганова, 6  
e-mail: dima\_nov@mail.ru  
cld@newman.bas-net.by*

**D.Ya. Novikov, L.D. Cheremisinova**

**IMPLICATIVE METHOD  
OF PARTIALLY DEFINED BOOLEAN FUNCTIONS ANALYSIS**

We consider a problem of verifying whether systems of partially defined Boolean functions are in relation of equivalence or realization. A solution method is proposed. It reduces the considered problem to the verification of satisfiability of a conjunctive normal form (CNF), which is constructed using implication relation.

УДК 519.873:519.718.7

Л.А. Золоторевич

## ПОСТРОЕНИЕ ТЕСТОВ И ВЕРИФИКАЦИЯ ПОТОКОВЫХ МОДЕЛЕЙ ЦИФРОВЫХ УСТРОЙСТВ НА ЯЗЫКЕ VHDL

*Предлагается единый подход к верификации проектов и направленному построению тестов контроля СБИС, представленных в потоковом виде на уровне RTL на языке VHDL с использованием арифметических, логических операторов и оператора If. Задача построения тестов и верификации проектов решается на основе КНФ-выполнимости некоторой системы булевых функций.*

### Введение

Проблемы верификации проектов и построения тестов контроля сложнофункциональных электронных систем остаются наиболее наукоемкими во всем спектре проблем проектирования СБИС [1]. Современные интегральные схемы содержат порядка миллиарда транзисторов на кристалле, и разработка тестов для объектов такого размера на уровне структурного представления оказалась практически неразрешимой задачей. В то же время острая потребность в тестах имеет место на всех этапах жизненного цикла, начиная с верхних уровней процесса проектирования, так как формальные методы верификации развиты недостаточно и верификация проектов на практике решается на основе моделирования. Тесты контроля необходимы также на этапе производства для отбраковки готовых изделий и при эксплуатации для оценки работоспособности объектов.

Следует заметить, что быстро развивающаяся электронная отрасль выставляет все новые требования к задаче построения тестов. Если в конце прошлого века рассматривалась задача эффективного построения теста контроля на уровне заданной структуры объекта, то в настоящее время кроме данной задачи ведется поиск решения задачи построения тестов для систем, представленных в разных системах идентификации.

Подойти к решению задачи построения тестов для контроля СБИС оказалось возможным на основе идентификации объекта на начальных этапах проектирования, когда имеется некоторое поведенческое описание или описание объекта на уровне межрегистровых передач, которое содержит существенно меньшее число базовых примитивов, чем на структурном уровне. Вопросы разработки тестов контроля СБИС, когда отсутствуют сведения относительно структурной реализации объекта, решаются преимущественно на основе моделирования неисправностей и случайного построения и рассматриваются в работах [2–4]. В работе [5] предлагается методика построения функциональных неисправностей, аргументированно соответствующих неисправностям структурной реализации соответствующего механизма. Известны работы, рассматривающие задачи направленного построения тестов на верхних уровнях проектирования [3, 4].

Общий подход к иерархической генерации тестов СБИС на RTL-уровне [5], основанный на том, что каждая операция текстового кода, описывающего объект на уровне RTL, реализуется на структурном уровне некоторым набором аппаратных средств. Тест контроля этих средств может быть построен известными методами и средствами на основе структурного представления устройства. Тест вносится в описание объекта, устанавливаются ограничения на функционирование объекта. Задача построения теста контроля всего объекта сводится к решению системы арифметических уравнений с внесенными ограничениями.

В настоящей работе описывается механизм решения задачи направленной генерации тестов, основанный на построении системы арифметических уравнений и итерационном решении задачи КНФ-выполнимости некоторой системы булевых функций.

Проблема построения тестов непосредственно связана с проблемой верификации проектов сложнофункциональных цифровых систем. На сегодняшний день проблема верификации проектов на разных этапах проектирования решается в основном на основе моделирования объекта, так как применяемые методы формальной верификации ориентированы на решение

некоторых частных задач. Полнота верификации проектов обеспечивается полнотой применяемых при моделировании тестов. Предлагаемый в данной статье метод направлен как на решение задачи направленного построения тестов контроля, так и на верификацию проекта путем моделирования на заданном тесте. Метод исследован на примерах описаний с использованием логических операторов, операторов целочисленного сложения, умножения, назначения значений переменным и сигналам и оператора If.

## 1. Структура представления объекта на языке VHDL

С учетом сложившейся традиции цифровая система рассматривается на уровне RTL как две подсистемы: операционная, выполняющая преобразование данных в соответствии с заданными алгоритмами, и управляющая, реализующая управление операционной частью. Поэтому в качестве математической платформы для описания цифровой системы будем использовать описание в виде графов потоков данных и потока управления. Граф потока управления – это ориентированный граф с узлами, соответствующими операторам текстового кода, и ребрами, указывающими порядок выполнения операторов. Граф потока данных – это ориентированный граф с узлами, соответствующими выполняемым операциям, и ребрами, указывающими последовательность операций по преобразованию некоторой переменной или сигнала.

Граф потока управления и графы потоков данных разрабатываются при статическом анализе программного кода. Отображение описания объекта в виде графов потоков данных и потока управления применяется обычно при создании компиляторов. По существу, графы потоков данных и потока управления представляют собой внутреннюю структуру данных компилятора, в которой заложен результат анализа текста описания объекта. К сожалению, внутреннее представление объекта в существующих фирменных компиляторах VHDL недоступно для использования извне. Большая часть работ по построению графов потоков данных и потока управления базируется на применении моделей, построенных вручную. В работе [6] предложена реализация идеи формального построения графа применительно к C-спецификации. Ниже приводится структура внутреннего представления RTL-описания для реализации метода направленного построения теста.

Общая структура *DD* (decision diagrams) потока управления (рис. 1) есть ориентированный ациклический граф, в котором вершинами являются операторы языка VHDL. Ориентированные ребра соответствуют передаче управления от одной вершины к другой. Каждый оператор программного кода соответствует некоторому узлу *DD* потока управления. Узел имеет маркер, который представляется семеркой вида  $\langle A_{stm}, A_{IF}, A_{Else}, A_{next}, P_{out}, P_{act}, P_t \rangle$ .

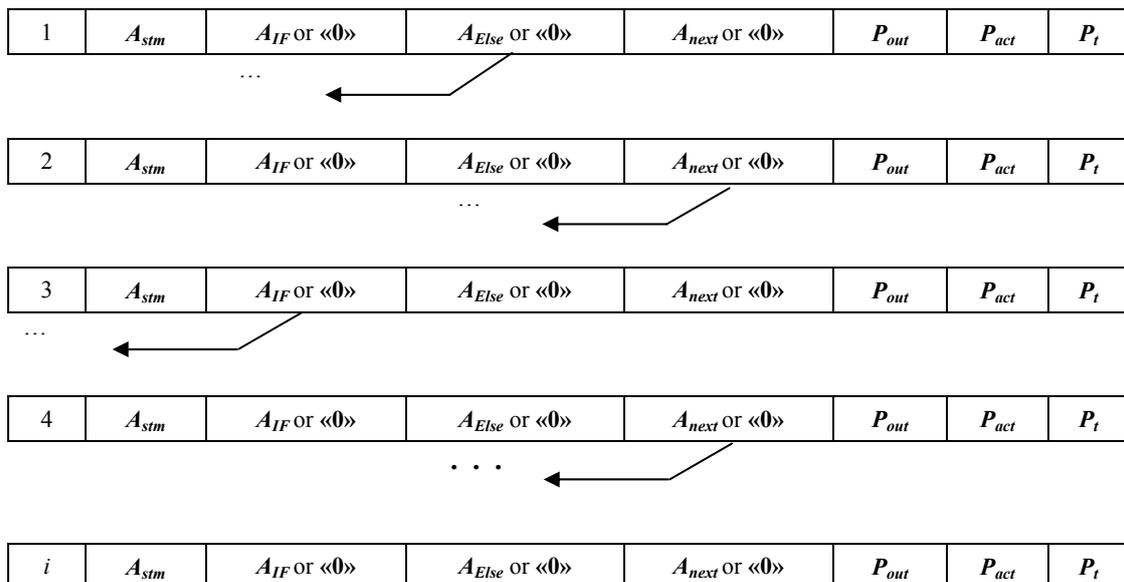


Рис. 1. Структура *DD* потока управления

Первое значение  $A_{stm}$  является адресом выполняемой логической или арифметической операции, последующие значения  $A_{If}$ ,  $A_{Else}$ ,  $A_{next}$  представляют собой ссылки на соответствующие узлы графа  $DD$  потока управления. Если узел 1 соответствует оператору  $If$  и условие ложно, то  $A_{If} = 0$ , а  $A_{Else}$  содержит указатель на узел  $DD$ , соответствующий следующему выполняемому оператору. Если узел соответствует не оператору  $If$ , а некоторому оператору с логической или арифметической операцией, то  $A_{If} = 0$ ,  $A_{Else} = 0$ , а  $A_{next}$  содержит указатель на некоторый узел  $DD$ , соответствующий следующему выполняемому оператору;  $P_{out}$  – признак наблюдаемости;  $P_{act}$  – признак активизации соответствующего оператора;  $P_t$  – время активизации.

На рис. 2 изображена структура  $DD$  потоков данных:  $Stm$  – адрес начала описания узла;  $ID$  – идентификатор левого операнда соответствующего оператора;  $A_{prvs}$  – ссылка на предшествующий оператор с аналогичным идентификатором левого операнда (или нуль);  $A_{next}$  – ссылка на последующий оператор с аналогичным идентификатором левого операнда (или нуль); Expression – вычисляемое арифметическое или логическое выражение.

$Stm$	$ID$	$A_{prvs}$ or «0»	$A_{next}$ or «0»	Expression
$Stm_1$				
$Stm_2$			$A_{next}$ or «0»	
...				
$Stm_n$				

Рис. 2. Структура  $DD$  потоков данных

На рис. 3 приведен некоторый набор операторов языка VHDL, который является модельным и не описывает реальный объект (после оператора 4, перед и после оператора 10 поставлены многоточия). Ниже рассмотрен пример построения  $DD$  потока управления (рис. 4) и потоков данных (рис. 5) для этого набора операторов.

```

entity Vonescnt is
  port (X1, X2: in unsigned (7downto 0), rst, clk: in bit;
        Z1, Z2: out unsigned (7downto 0);
end Vonescnt;
architecture Vonescnt_arch of Vonescnt is
  process (rst, clk)
  variable Rab1, Rab2, Rab3, Rab4, Rab5: unsigned (7downto 0);
  begin
    1   if (rst = '1') then
    2     RAB1:=0;
    3     RAB2:=0;
    4     Elsif (clk' event and clk=1) then
    5
    6     ...
    7     ...
    8     ...
    9     ...
    10    else
    ...
    end if;
  end process
end Vonescnt;

```

RAB1:=X1;  
 RAB2:=X2;  
 RAB3:= RAB1+ RAB2;  
 RAB4:=RAB5;  
 Z1<=RAB4;  
 Z2<=RAB3;

Рис. 3. Фрагмент VHDL-кода

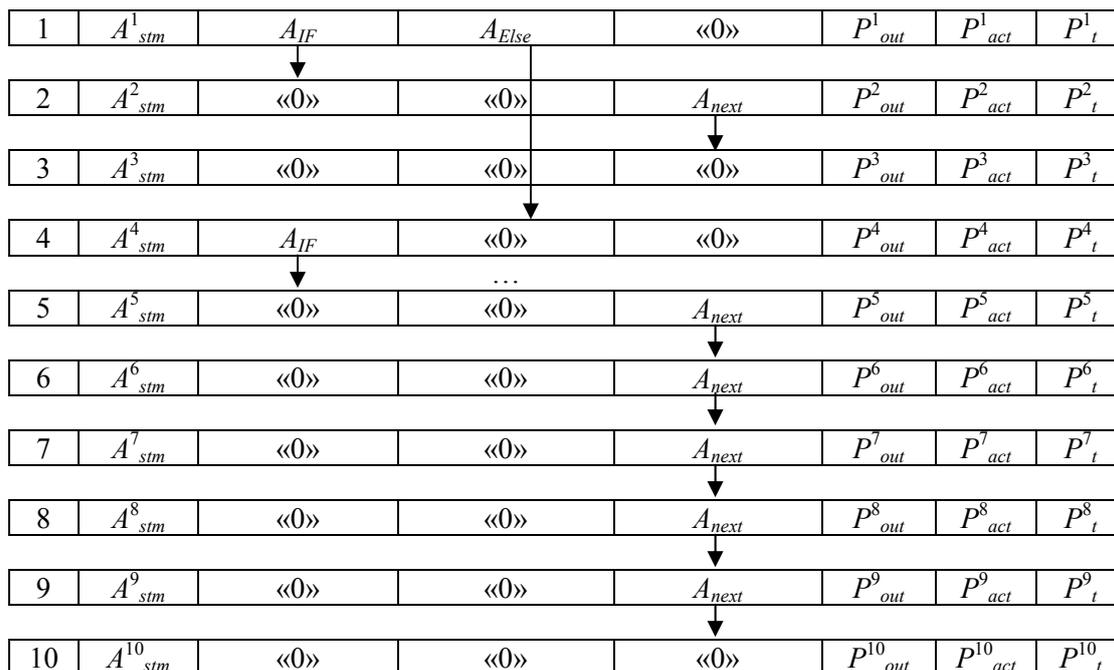


Рис. 4. Структура DD потока управления для примера на рис. 3

$Stm$	$DD$	$A_{prvs}$	$A_{next}$	Expression
$Stm_1$	$RST$	0	0	rst='1'
$Stm_2$	$RAB1$	0	$A^5_{stm}$	RAB1:=0;
$Stm_3$	$RAB2$	0	$A^6_{stm}$	RAB2:=0;
$Stm_4$		0	0	clk' event and clk=1
		...		
$Stm_5$	$RAB1$	$A^2_{stm}$	0	RAB1:=X1;
$Stm_6$	$RAB2$	$A^3_{stm}$	0	RAB2:=X2;
$Stm_7$	$RAB3$	0	0	RAB3:=RAB1+RAB2;
$Stm_8$	$RAB4$	0	0	RAB4:=RAB5;
$Stm_9$	$Z1$	0	0	Z1:=RAB4;
$Stm_{10}$	$Z2$	0	0	Z2:=RAB3;

Рис. 5. Структура DD потоков данных для примера на рис. 3

В теории тестового диагностирования известны методы направленного и случайного построения тестов. Если рассматривать возможность их реализации в указанной выше постановке, то следует отметить весьма существенную особенность. Если методы построения теста случайным образом можно попытаться реализовать на основе имеющихся фирменных компиляторов языка VHDL, то для направленного построения тестов необходимо знание внутреннего представления кода для построения структур графов, которое, к сожалению, не раскрывается разработчиками компиляторов и не может быть использовано при необходимости. Для построения графов необходим синтаксический анализатор программного кода, который используется при построении одного графа потока управления, и столько графов потоков данных, сколько переменных описывает полное состояние моделируемой системы.

Ответственным моментом является выбор моделей неисправностей рассматриваемых операторов кода описания объекта. В настоящее время нет доказательной базы для выбора некоторой определенной модели используемой неисправности, поэтому целесообразно работать в более широком диапазоне моделей, известных в литературе. В данной работе кроме известных моделей, таких как модель выпавшего оператора; замена условного перехода безусловным, одной операции некоторой другой; неисправности константного типа некоторой переменной, сигнала, некоторого

разряда переменной, будет использоваться также и модель функциональной неисправности, которая аргументированно соответствует неисправности константного типа реального объекта [5].

Научная гипотеза, используемая при разработке метода направленного построения тестов на RTL-уровне, формулируется так: для того чтобы найти входные данные, которые позволят определить по выходным данным наличие или отсутствие некоторой неисправности в системе, представленной в рамках любой системы идентификации, необходимо активизировать неисправность, т. е. заставить ее проявиться на выходе некоторого элемента системы, затем заставить ее проявиться хотя бы на одном из выходов системы, после чего необходимо вычислить все входные данные, которые позволят сохранить условия, полученные при решении данной задачи. Эта гипотеза сформулирована на основе идеи Рота, реализованной при построении тестов контроля объекта на структурном уровне.

## 2. Построение и решение системы арифметических уравнений

Общая идея метода направленного построения тестов контроля цифровых систем, описанных на уровне RTL на языке VHDL, заключается:

- в переходе от системы арифметических и логических уравнений, описывающих поведение объекта с некоторой внесенной неисправностью, к построению системы КНФ булевых функций разрешения;

- конъюнктивном объединении функций разрешения;

- решении задачи выполнимости результирующей КНФ разрешения объекта.

Предположим, что все входные переменные являются целочисленными размерностью  $n$  бит ( $\text{mod } 2^n$ ). Для генерации тестов необходимо:

- на основе VHDL-кода объекта составить систему арифметических уравнений, описывающих функционирование объекта;

- выполнить корректировку системы с учетом внесения неисправностей соответствующего оператора;

- поставить в соответствие каждой целочисленной переменной размерностью  $n$  бит ( $\text{mod } 2^n$ ) логический вектор длины  $n$ ;

- итеративно выполнить решение системы. Для получения  $i$ -го бита результата арифметические выражения транслируются в КНФ булевых функций разрешения.

Все полученные КНФ-функции разрешения объединяются по правилу И, решается задача КНФ-выполнимости полученной системы булевых функций. Если система выполнима, то получен  $i$ -й бит разрабатываемого теста контроля объекта с внесенной неисправностью. В противном случае тест не может быть построен, так как внесенные ограничения не могут быть удовлетворены. В таком случае для проверки рассматриваемой неисправности необходимо изменить систему управления объекта с целью повышения его управляемости и наблюдаемости.

**Пример:** рассмотрим фрагмент некоторого VHDL-кода (рис. 6). Здесь  $A, B, C, S$  – входные данные,  $L$  – переменная выхода. Положим, что переменные  $A, B, C$  являются целочисленными по модулю  $2^n$ , а  $S$  – однобитовая переменная. На рис. 7 показана система арифметических уравнений, которые описывают функционирование объекта, представленного VHDL-кодом (см. рис. 6). В объекте имеется мультиплексор, два сумматора, умножитель и схема сравнения. Построим тест, проверяющий правильность выполнения оператора целочисленного сложения  $G = B + C$ .

```

if S = '0' then
  D := A;
else D := B;
  G := B + C;
  E := D + C;
  F := E * G;
  L := D < G;
end if;

```

Рис. 6. Фрагмент VHDL-кода

$$\begin{aligned}
 D &= \bar{S} * A + S * B ; \\
 \boxed{G} &= B + C ; \\
 E &= D + C ; \\
 F &= E * G ; \\
 L &= D < G ;
 \end{aligned}$$

Рис. 7. Система арифметических уравнений ( $\text{mod } 2^n$ )

С данным оператором связаны аппаратные средства (некоторый механизм реализации), обеспечивающие сложение целых чисел. Положим, что известен тест (последовательность входных наборов) для контроля сумматора по модулю  $2^n$  и выбран один набор теста, который задает  $B = 15$ ,  $C = 1$ ,  $G' = 17$  по модулю  $2^n$ , где  $G'$  – неисправное значение переменной  $G$ . Для построения теста контроля рассматриваемого объекта система уравнений (рис. 7) должна быть скорректирована, как показано на рис. 8, чтобы обеспечить распространение неисправности к выходам объекта и определение входных переменных. Здесь операторы 1–5 описывают исправный исходный объект, операторы 6–8 – процесс внесения неисправности, операторы 9–13 задают ограничения, обеспечивающие распространение эффекта неисправности к выходу объекта.

№ п/п	Арифметические уравнения	№ п/п	Арифметические уравнения
1	$D = \bar{S} \times A + S \times B \pmod{2^n}$	8	$G' = 17 \pmod{2^n}$
2	$G = B + C \pmod{2^n}$	9	$G' \neq G \pmod{2^n}$
3	$E = D + C \pmod{2^n}$	10	$F' = E * G' \pmod{2^n}$
4	$F = E * G \pmod{2^n}$	11	$F \neq F' \pmod{2^n}$
5	$L = D < G \pmod{2^n}$	12	$L' = D < G' \pmod{2^n}$
6	$B = 15 \pmod{2^n}$	13	$L \neq L' \pmod{2^n}$
7	$C = 1 \pmod{2^n}$		

Рис. 8. Система арифметических уравнений ( $\text{mod } 2^n$ ) с внесенной неисправностью

Идея решения подобных систем арифметических уравнений основана на поразрядном подходе к вычислению значений и состоит в том, что каждой целочисленной переменной  $m \leq 2^n$  ставится в соответствие определенный логический вектор размерности  $n$ . Для описания алгоритма вычисления системы арифметических уравнений используем логическую функцию разрешения, которая задает соотношения между исправными логическими состояниями выводов физических элементов, реализующих определенную логическую функцию.

Функция  $F^f$ , называемая функцией разрешения для логической функции  $f$ , вводится в работе [7]. Функция  $F^f$  зависит не только от аргументов функции  $f$ , но и от самой  $f$  и принимает значение логической 1 при всех допустимых состояниях входных и выходной переменных. Функция  $F^f$  принимает значение 0 при всех недопустимых состояниях входных и выходной переменных. Для полноты изложения рассмотрим получение функции разрешения  $F^f$  для функции конъюнкции  $f = a * b$  (табл. 1).

Таблица 1  
Функции разрешения и запрета

$a$	$b$	$f$	$F^f$	$\overline{F^f}$
0	0	0	1	0
0	1	0	1	0
1	0	0	1	0
1	1	1	1	0
0	0	1	0	1
0	1	1	0	1
1	0	1	0	1
1	1	0	0	1

Получим СКНФ функции  $F^f$  по табл. 1, выбирая конституэнты 0. Для этого необходимо произвести следующие действия:

- 1) выбрать наборы аргументов, на которых функция обращается в нуль;
- 2) выписать дизъюнкции, соответствующие этим наборам, причем если аргумент  $x_i$  входит в набор как нуль, то в дизъюнцию он вписывается без изменения. Если же аргумент  $x_i$  входит в данный набор как единица, то в соответствующую дизъюнцию вписывается его отрицание;

3) все выписанные дизъюнкции соединить знаком конъюнкции:

$$F^f = (a \vee b \vee \bar{f})(a \vee \bar{b} \vee \bar{f})(\bar{a} \vee b \vee \bar{f})(\bar{a} \vee \bar{b} \vee f) = (a \vee \bar{f})(b \vee \bar{f})(\bar{a} \vee \bar{b} \vee f).$$

Функция  $\bar{F}^f$ , называемая функцией запрета, принимает значения, инверсные функции разрешения  $F^f$ . Выражение функции разрешения для операции поразрядного сложения  $f = a \oplus b$  выглядит так:  $(a \vee b \vee \bar{f})(a \vee \bar{b} \vee \bar{f})(\bar{a} \vee b \vee \bar{f})(\bar{a} \vee \bar{b} \vee f)$ .

Отметим, что описание объектов в виде КНФ функций разрешения и решение задачи SAT (выполнимости булевой функции) в последнее время достаточно широко используются в литературе при решении задач логического проектирования [8–12].

В табл. 2 приведены однобитовые операции, используемые в рассматриваемом на рис. 8 примере, и соответствующие функции разрешения в виде КНФ.

Таблица 2

Функции разрешения

Однобитовые арифметические и логические уравнения	КНФ функций разрешения
$f = b \vee c$	$(\bar{b} \vee f)(\bar{c} \vee f)(b \vee c \vee \bar{f})$
$f = b \times c$	$(b \vee \bar{f})(c \vee \bar{f})(\bar{b} \vee \bar{c} \vee f)$
$f = a \oplus b$	$(a \vee b \vee \bar{f})(a \vee \bar{b} \vee \bar{f})(\bar{a} \vee b \vee \bar{f})(\bar{a} \vee \bar{b} \vee f)$
$f = a \sim b$	$(a \vee b \vee f)(a \vee \bar{b} \vee \bar{f})(\bar{a} \vee b \vee \bar{f})(\bar{a} \vee \bar{b} \vee f)$
$f = \bar{s} \times a + s \times b$	$(s \vee a \vee \bar{f})(a \vee b \vee \bar{f})(\bar{s} \vee b \vee \bar{f}) \times$ $\times (s \vee \bar{a} \vee f)(\bar{s} \vee \bar{b} \vee f)(\bar{a} \vee \bar{b} \vee f)$
$f = a \geq b$	$(a \vee f)(\bar{a} \vee \bar{f})$
$f = b < c$	$(\bar{b} \vee \bar{f})(c \vee \bar{f})(b \vee \bar{c} \vee f)$
$f = 1$	$f$
$f = 0$	$\bar{f}$
$f \neq f'$	$(f \vee f' \vee \bar{f}'')(f \vee \bar{f}' \vee f'') \times$ $\times (\bar{f} \vee f' \vee f'')(\bar{f} \vee \bar{f}' \vee \bar{f}'')$

Решение системы уравнений, приведенной на рис. 8, выполняется итеративно. Чтобы одновременно решить равенства по модулю 2, объединяем все КНФ разрешения логических функций вместе, используя логическую функцию конъюнкции. Чтобы найти второй бит решения этого уравнения, необходимо найти рекурсивное уравнение по модулю 2 аналогично, используя результаты предыдущей итерации.

Для вычисления очередного бита результата формируется система функций разрешения, соответствующих каждой арифметической функции, затем решается задача выполнимости конъюнкции всех функций разрешения. К примеру, для уравнения  $E = D + C$  вначале вычисляется значение младшего бита результата  $E_0 = D_0 + C_0$ . После вычисления  $E_0$  его результат используется при формировании следующих битов результата. Для определения порядка вычисления битов высшего порядка из битов более низкого порядка необходимо рассмотреть различные формы уравнений. Для операции суммирования, начиная со 2-й итерации, очередной бит результата вычисляется следующим образом:  $E_i = D_i + C_i + P_i$ , где  $P_i$  – значение переноса из  $i + 1$ -го разряда.

В табл. 3 приведены функции разрешения для первого бита системы арифметических уравнений (см. рис. 8).

Таблица 3

Функции разрешения для вычисления первого бита результата ( $f_0^n$  – результат предыдущей итерации)

Арифметические уравнения	Эквивалентные функции разрешения
$D_0 = \overline{S_0} * A_0 + S_0 * B_0 \pmod{2}$	$(S_0 \vee A_0 \vee \overline{D_0})(A_0 \vee B_0 \vee \overline{D_0})(\overline{S_0} \vee B_0 \vee \overline{D_0}) \times$ $\times (S_0 \vee \overline{A_0} \vee D_0)(\overline{S_0} \vee \overline{B_0} \vee D_0)(\overline{A_0} \vee \overline{B_0} \vee D_0)$
$G_0 = B_0 + C_0 \pmod{2}$	$(C_0 \vee B_0 \vee \overline{G_0})(C_0 \vee \overline{B_0} \vee \overline{G_0}) \times$ $\times (\overline{C_0} \vee B_0 \vee G_0)(\overline{C_0} \vee \overline{B_0} \vee \overline{G_0})$
$E_0 = D_0 + C_0 \pmod{2}$	$(C_0 \vee D_0 \vee \overline{E_0})(C_0 \vee \overline{D_0} \vee \overline{E_0}) \times$ $\times (\overline{C_0} \vee D_0 \vee E_0)(\overline{C_0} \vee \overline{D_0} \vee \overline{E_0})$
$F_0 = E_0 * G_0 \pmod{2}$	$(E_0 \vee \overline{F_0})(G_0 \vee \overline{F_0})(\overline{E_0} \vee \overline{G_0} \vee F_0)$
$L_0 = D_0 < G_0 \pmod{2}$	$(\overline{D_0} \vee \overline{L_0})(G_0 \vee \overline{L_0})(D_0 \vee \overline{G_0} \vee L_0)$
$B_0 = 1 \pmod{2}$	$B_0$
$C_0 = 1 \pmod{2}$	$C_0$
$G'_0 = 1 \pmod{2}$	$G'_0$
$G'_0 \neq G_0 \pmod{2}$	$(G_0 \vee G'_0 \vee \overline{f_0^n})(G_0 \vee \overline{G'_0} \vee f_0^n) \times$ $\times (\overline{G_0} \vee G'_0 \vee f_0^n)(\overline{G_0} \vee \overline{G'_0} \vee \overline{f_0^n})$
$F'_0 = E_0 * G'_0 \pmod{2}$	$(E_0 \vee \overline{F'_0})(G_0 \vee \overline{F'_0})(\overline{E_0} \vee \overline{G'_0} \vee F'_0)$
$F_0 \neq F'_0 \pmod{2}$	$(F_0 \vee F'_0 \vee \overline{f_0^n})(F_0 \vee \overline{F'_0} \vee f_0^n) \times$ $\times (\overline{F_0} \vee F'_0 \vee f_0^n)(\overline{F_0} \vee \overline{F'_0} \vee \overline{f_0^n})$
$L'_0 = D_0 < G'_0 \pmod{2}$	$(\overline{D_0} \vee \overline{L'_0})(G'_0 \vee \overline{L'_0})(D_0 \vee \overline{G'_0} \vee L'_0)$
$L_0 \neq L'_0 \pmod{2}$	$(L_0 \vee L'_0 \vee \overline{f_0^n})(L_0 \vee \overline{L'_0} \vee f_0^n) \times$ $\times (\overline{L_0} \vee L'_0 \vee f_0^n)(\overline{L_0} \vee \overline{L'_0} \vee \overline{f_0^n})$

Для рекурсивного вычисления неравенства необходимо учитывать, что оно может быть определено только в старшем  $i$ -м бите ( $i$  от 0 до  $n - 1$ ). Все функции разрешения объединяются знаком конъюнкции, решается задача выполнимости полученной системы булевых функций:

$$\begin{aligned}
& ((S_0 \vee A_0 \vee \overline{D_0})(A_0 \vee B_0 \vee \overline{D_0})(\overline{S_0} \vee B_0 \vee \overline{D_0}) * (S_0 \vee \overline{A_0} \vee D_0)(\overline{S_0} \vee \overline{B_0} \vee D_0)(\overline{A_0} \vee \overline{B_0} \vee D_0)) \times \\
& \quad \times ((C_0 \vee B_0 \vee \overline{G_0})(C_0 \vee \overline{B_0} \vee \overline{G_0})(\overline{C_0} \vee B_0 \vee G_0) * (\overline{C_0} \vee \overline{B_0} \vee \overline{G_0})) \times \\
& \quad \times ((C_0 \vee D_0 \vee \overline{E_0})(C_0 \vee \overline{D_0} \vee \overline{E_0})(\overline{C_0} \vee D_0 \vee E_0) * (\overline{C_0} \vee \overline{D_0} \vee \overline{E_0})) \times \\
& \times ((E_0 \vee \overline{F_0})(G_0 \vee \overline{F_0})(\overline{E_0} \vee \overline{G_0} \vee F_0)) * ((\overline{D_0} \vee \overline{L_0})(G_0 \vee \overline{L_0})(D_0 \vee \overline{G_0} \vee L_0)) \times B_0 \times C_0 \times G'_0 \times \\
& \quad \times ((G_0 \vee G'_0 \vee \overline{f_0^n})(G_0 \vee \overline{G'_0} \vee f_0^n)(\overline{G_0} \vee G'_0 \vee f_0^n) * (\overline{G_0} \vee \overline{G'_0} \vee \overline{f_0^n})) \times \\
& \quad \times ((E_0 \vee \overline{F'_0})(G_0 \vee \overline{F'_0})(\overline{E_0} \vee \overline{G'_0} \vee F'_0)) \times \\
& \quad \times ((F_0 \vee F'_0 \vee \overline{f_0^n})(F_0 \vee \overline{F'_0} \vee f_0^n)(\overline{F_0} \vee F'_0 \vee f_0^n) * (\overline{F_0} \vee \overline{F'_0} \vee \overline{f_0^n})) \times \\
& \quad \times ((\overline{D_0} \vee \overline{L'_0})(G'_0 \vee \overline{L'_0})(D_0 \vee \overline{G'_0} \vee L'_0)) \times \\
& \quad \times ((L_0 \vee L'_0 \vee \overline{f_0^n})(L_0 \vee \overline{L'_0} \vee f_0^n)(\overline{L_0} \vee L'_0 \vee f_0^n) * (\overline{L_0} \vee \overline{L'_0} \vee \overline{f_0^n})) = 1.
\end{aligned}$$

Более детальное рассмотрение примера до момента получения конечного результата является громоздким, так как требует описания процедур, связанных с итерационным вычислением выполнимости функции разрешения. Рассмотрение примера в статье направлено на то, чтобы показать основные этапы построения теста. Дадим интерпретацию ожидаемого результата.

Ярусно-параллельная форма показанного на рис. 6 VHDL-кода включает мультиплексор, два сумматора, умножитель и схему сравнения (рис. 9).

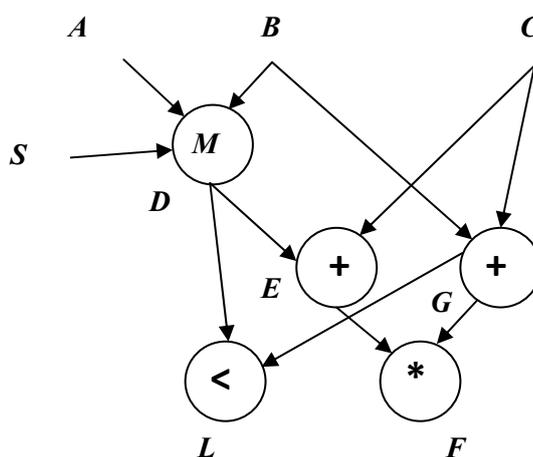


Рис. 9. Ярусно-параллельная форма представления кода

Рассматривается неисправность, которая приводит к появлению ошибочного результата  $G' = B + C = 17$  ( $B = 15, C = 1$ ) вместо  $G = B + C = 16$  (см. рис 8). Очевидно, что тестом должны быть такие значения входных переменных для рассматриваемого кода, т. е.  $A, B, C$  и  $S$ , при которых будет выполняться условие  $L \neq L'$  или  $F \neq F'$ . При этом переменные  $B$  и  $C$  определены на этапе внесения неисправности. Поэтому будут вычислены значения переменных  $A$  и  $S$ . Для получения теста контроля объекта по выходу  $L$  необходимо определить такие значения параметров  $A$  и  $S$ , которые обеспечат выполнение условия  $L \neq L'$ .

Если  $S = 1$ , то для построения теста необходимо, чтобы выполнялись условия  $(D = 15) < (G = 16)$  и  $(D = 15) \geq (G' = 17)$  или  $(D = 15) \geq (G = 16)$  и  $(D = 15) < (G' = 17)$ , которые являются конфликтными. Это говорит о том, что неисправность по выходу  $L$  не определяется. Если  $S = 0$ , то для построения теста необходимо, чтобы выполнялись условия  $(D = A) < (G = 16)$  и  $(D = A) \geq (G' = 17)$  или  $(D = A) \geq (G = 16)$  и  $(D = A) < (G' = 17)$ . Решением будет  $A = 16$ . Тестом явится  $S = 0, A = 16, B = 15, C = 1$ .

### 3. Верификация проектов на RTL-уровне

Верификация проектов, как известно, занимает большую часть времени проектирования сложнофункциональной СБИС. На сегодняшний день основным методом практической верификации на всех этапах проектирования является моделирование. Такой подход, основанный на моделировании, требует наличия тестов. Для моделирования объекта, описанного на языке VHDL на уровне RTL, предлагается метод, который заключается в следующем:

1. Представить VHDL-код в виде системы арифметических уравнений.
2. Добавить в полученную систему уравнений означивание входных и выходных переменных.
3. Итеративно решить систему. Для этого перейти от системы арифметических уравнений к системе логических функций разрешения, учитывая особенность соответствующей итерации.

4. Получить КНФ-функцию разрешения и вычислить ее выполнимость. По результатам перейти к новому входно-выходному экземпляру теста или закончить моделирование и перейти к анализу результатов.

Рассмотрим моделирование объекта, описанного VHDL-кодом, который приведен на рис. 6. Предположим, что известен тест, одним из шаблонов которого является  $S = 0$ ,  $A = 34$ ,  $B = 45$ ,  $C = 7$ ,  $L = 1$ ,  $F = 2132$ . Система арифметических уравнений дополняется тестовыми значениями входных и выходных переменных и переводится в систему функций разрешения для проведения первого этапа вычислений, затем формируется КНФ функции разрешения конъюнктивным объединением полученных функций и решается задача выполнимости. В табл. 4 приведены система арифметических уравнений и система логических функций разрешения для выполнения первой итерации вычислений.

Таблица 4

Функции разрешения для получения первого бита результата верификации

Арифметические уравнения	Эквивалентные функции разрешения
$D_0 = \overline{S_0} * A_0 + S_0 * B_0 \pmod{2}$	$(S_0 \vee A_0 \vee \overline{D_0})(A_0 \vee B_0 \vee \overline{D_0})(\overline{S_0} \vee B_0 \vee \overline{D_0}) \times$ $\times (S_0 \vee \overline{A_0} \vee D_0)(\overline{S_0} \vee \overline{B_0} \vee D_0)(\overline{A_0} \vee \overline{B_0} \vee D_0)$
$G_0 = B_0 + C_0 \pmod{2}$	$(C_0 \vee B_0 \vee \overline{G_0})(C_0 \vee \overline{B_0} \vee \overline{G_0}) \times$ $\times (\overline{C_0} \vee B_0 \vee G_0)(\overline{C_0} \vee \overline{B_0} \vee \overline{G_0})$
$E_0 = D_0 + C_0 \pmod{2}$	$(C_0 \vee D_0 \vee \overline{E_0})(C_0 \vee \overline{D_0} \vee \overline{E_0}) \times$ $\times (\overline{C_0} \vee D_0 \vee E_0)(\overline{C_0} \vee \overline{D_0} \vee \overline{E_0})$
$F_0 = E_0 * G_0 \pmod{2}$	$(E_0 \vee \overline{F_0})(G_0 \vee \overline{F_0})(\overline{E_0} \vee \overline{G_0} \vee F_0)$
$L_0 = D_0 < G_0 \pmod{2}$	$(\overline{D_0} \vee \overline{L_0})(G_0 \vee \overline{L_0})(D_0 \vee \overline{G_0} \vee L_0)$
$A_0 = 0 \pmod{2}$	$\overline{A_0}$
$B_0 = 1 \pmod{2}$	$B_0$
$C_0 = 1 \pmod{2}$	$C_0$
$L_0 = 1 \pmod{2}$	$L_0$
$F_0 = 0 \pmod{2}$	$\overline{F_0}$

### Заключение

Предложенный подход к направленному построению тестов и верификации ориентирован на проекты цифровых объектов, представленных потоковой моделью на уровне RTL на языке VHDL с использованием арифметических, логических операторов и оператора if-then-else. При переходе от RTL-уровня к структурному представлению объекта, когда объект рассматривается как сеть из функциональных блоков комбинационного и последовательностного типов, для построения тестов применяются другие подходы.

### Список литературы

1. Electronic Design Automation: Synthesis, Verification, and Test / ed. L.-T. Wang, Y.-W. Chang, K.-T. Cheng. – Elsevier, 2009.
2. Gharebaghi, A.M. High-Level Test Generation from VHDL Behavioral Descriptions / A.M. Gharebaghi, Z. Navabi // Proc. of VHDL Intern. Users Forum Fall Workshop. – Orlando, Florida, 2000. – P. 123–126.
3. Murray, B.T. Hierarchical Test Generation Using Precomputed Tests for Modules / B.T. Murray, J.P. Hayes // Intern. Test Conf. – Washington, 1988. – P. 221–229.
4. Golubeva O. High-level test generation for hardware testing and software validation / O. Golubeva, M. Sonza Reorda, M. Violante // Workshop of High-Level Design Validation and Test. – San Francisco, California, 2003. – P. 143–148.
5. Zolotarevich, L.A. Development of tests for VLSI circuit testability at the upper design levels / L.A. Zolotarevich, A.V. Il'inkova // Automation and Remote Control. – 2010. – Vol. 71, iss. 9. – P. 1888–1898.

6. Vallerio, K.S. Task graph extraction for embedded system synthesis / K.S. Vallerio, N.K. Jha // Proc. of IEEE Conf. on VLSI Design. – Portland, Oregon, 2003.
7. Larrabee, T. Test pattern generation using Boolean satisfiability / T. Larrabee // IEEE Trans. Computer-Aided Design. – 1992. – Vol. 11, № 1. – P. 4–15.
8. Новиков, Д.Я. Верификация функциональных описаний с неопределенностью на основе парафазного представления булевых функций / Д.Я. Новиков, Л.Д. Черемисинова // Информатика. – 2010. – № 3. – С. 54–62.
9. Test Pattern Generation using Boolean Proof Engines / R. Drechsler [et al.]. – Springer, Dordrecht, Heidelberg, London, New York, 2009.
10. Automatic Constraint Based Test Generation for Behavioral HDL Models / S.K. Hari [et al.] // IEEE Trans. on VLSI systems. – 2008. – Vol. 16, № 4. – P. 408–421.
11. Alizadeh, B. High level test generation without ILP and SAT Solvers / B. Alizadeh, M. Fujita // Int. Workshop on High Level Design Validation and Testing (HLDVT07). – Irvin, Ca, 2007. – P. 298–304.
12. Koo, H.-M. Functional Test Generation Using Design and Property Decomposition Techniques / H.-M. Koo, P. Mishra // ACM Transactions on Embedded Computing Systems. – 2009. – Vol. 8, № 4. – Article 32. – P. 1–32.

Поступила 16.03.12

*Белорусский государственный университет,  
Минск, пр. Независимости, 4  
e-mail: zolotorevichLA@bsu.by*

**L.A. Zolotorevich**

### **CONSTRUCTION OF TESTS AND VERIFICATION OF DIGITAL DEVICES FLOW MODELS ON VHDL LANGUAGE**

A unified approach to verification of projects and directed construction of VLSI tests control presented at the RTL level in language VHDL is offered. The problem of tests construction and verifications of projects is solved via CNF-satisfiability of some system of Boolean functions.

## МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

УДК 519.21

А.Р. Еремина

**ИНВАРИАНТНОСТЬ СТАЦИОНАРНОГО РАСПРЕДЕЛЕНИЯ СОСТОЯНИЙ НЕОДНОРОДНЫХ СЕТЕЙ МАССОВОГО ОБСЛУЖИВАНИЯ С МНОГОРЕЖИМНЫМИ СТРАТЕГИЯМИ И ДИСЦИПЛИНОЙ DPS**

*Рассматриваются открытые и замкнутые сети массового обслуживания с многорежимными стратегиями, разнотипными заявками и дисциплиной обслуживания «дискриминаторное разделение процессора» (DPS). Устанавливается инвариантность стационарного распределения вероятностей состояний сети по отношению к функциональной форме распределений величин работ, требующихся для обслуживания заявок и переключения режимов работы приборов в узлах, при фиксированных математических ожиданиях.*

**Введение**

Системы и сети массового обслуживания уже давно применяются в качестве вероятностных моделей различных информационно-компьютерных систем и сетей [1, 2]. В последнее время они также нашли широкое применение в финансовом секторе, производстве, логистике, технике и т. д.

Для указанных отраслей большое значение имеет изучение сетей, в узлах которых обслуживающие приборы могут работать с различной производительностью, требовать ремонта или замены. Ю.В. Малинковский ввел в рассмотрение сети массового обслуживания с многорежимными стратегиями [3, 4]. В таких сетях однолинейные узлы могут работать в различных режимах, которые характеризуются разной производительностью обслуживающего прибора. При этом прибор не выходит из строя полностью – уменьшается его производительность. Подобные сети исследовались также в работе [5].

Вместе с распространением сетей в качестве моделей реальных объектов растет и актуальность их исследования. При этом важную роль играет проблема инвариантности стационарного распределения вероятностей состояний сетей по отношению к функциональному виду распределений величин работ, требующихся для обслуживания заявок и переключения режимов работы приборов в узлах. Это связано с тем, что в реальных сетях указанные распределения часто отличаются от экспоненциального. Поэтому большую практическую значимость имеет изучение таких сетей массового обслуживания, в которых заявки обслуживаются не только по экспоненциальному закону, а обслуживающие приборы в узлах могут полностью или частично выходить из строя, работать с меньшей (или большей) производительностью, требовать ремонта или замены.

Для вычислительных сетей также представляют интерес модели, описывающие эффект разделения средств сети между несколькими требованиями, работами и т. д. При так называемых дисциплинах «разделения процессора» все или некоторые группы заявок обслуживаются одновременно единственным прибором с переменной скоростью обслуживания, принимающей дробные значения и изменяющейся во времени в зависимости от состояния сети. К группе данных дисциплин относятся «обобщенное разделение процессора» (GPS – Generalized Processor-Sharing), «справедливое разделение процессора» (EPS – Egalitarian Processor-Sharing), «дискриминаторное разделение процессора» (DPS – Discriminatory Processor-Sharing) и др. Подробно они рассмотрены в [6]. Свойства инвариантности сетей с разделением процессора исследовались в работах Чанди, Ховарда и Тоуслея [7], Келли [8] и др.

В настоящей работе впервые исследуются открытые и замкнутые сети с многорежимными стратегиями и дисциплиной обслуживания DPS, для которых величины работ по обслуживанию заявок и переключению режимов приборов в узлах имеют произвольные законы распределения, а сами заявки являются разнотипными.

### 1. Постановка задачи

Рассматриваются открытые и замкнутые сети с многорежимными стратегиями обслуживания, состоящие из  $N$  узлов, в которых циркулируют заявки  $M$  типов. Принадлежность заявки определенному типу обуславливает количество работы, необходимое для обслуживания данной заявки прибором узла.

В случае открытой сети поступающий поток заявок – простейший с интенсивностью  $\lambda$ . Каждая заявка входного потока независимо от других заявок направляется в  $l$ -й узел и становится заявкой типа  $u$  с вероятностью  $p_{0(l,u)}$ ,  $l = \overline{1, N}$ ,  $u = \overline{1, M}$  ( $\sum_{l=1}^N \sum_{u=1}^M p_{0(l,u)} = 1$ ). В случае замкнутой сети в ней циркулирует  $\sum_{l=1}^N n(l) = \overline{N} < \infty$  заявок, где  $n(l)$  – общее число заявок в  $l$ -м узле.

Обслуживание заявок в узлах осуществляется в соответствии с дисциплиной обслуживания DPS. Поступившая в узел заявка сразу начинает обслуживаться (очередь в ее традиционном понимании отсутствует). В моменты поступления новых или ухода обслуженных заявок происходят скачки скорости обслуживания. При такой дисциплине каждая заявка имеет свою скорость выполнения работы по обслуживанию, которая пропорциональна числу заявок данного типа в рассматриваемом узле и обратно пропорциональна общему числу заявок в узле.

После обслуживания в  $l$ -м узле заявка типа  $u$  мгновенно и независимо от других заявок направляется в  $k$ -й узел и становится заявкой типа  $v$  с вероятностью  $p_{(l,u)(k,v)}$ , а с вероятностью

$p_{(l,u)0}$  покидает сеть ( $\sum_{k=1}^N \sum_{v=1}^M p_{(l,u)(k,v)} + p_{(l,u)0} = 1$  для открытой сети и  $\sum_{k=1}^N \sum_{v=1}^M p_{(l,u)(k,v)} = 1$  для замкнутой сети;  $l, k = \overline{1, N}$ ;  $u, v = \overline{1, M}$ ).

В каждом узле  $l$  находится единственный обслуживающий прибор, который может работать в  $r_l + 1$  режимах  $0, 1, \dots, r_l$ ,  $l = \overline{1, N}$ . В качестве основного режима работы полагается режим 0. Во время переключения прибора с одного режима работы на другой число заявок в узле не меняется. Переход возможен только на соседние режимы. Под соседними понимаются режимы, номера которых имеют разницу  $\pm 1$ .

Состояние сети в момент времени  $t$  будем характеризовать вектором  $x(t) = (x_1(t), x_2(t), \dots, x_N(t))$ , где  $x_l(t) = (\bar{x}_l(t), j_l(t)) = (x_{l1}(t), x_{l2}(t), \dots, x_{lM}(t), j_l(t))$  описывает состояние  $l$ -го узла в момент времени  $t$ . Здесь  $x_{lu}(t)$  – число заявок  $u$ -го типа в  $l$ -м узле в момент времени  $t$ ,  $j_l(t)$  – режим, в котором работает  $l$ -й узел в момент времени  $t$ . Тогда  $x(t)$  обладает не более чем счетным фазовым пространством состояний  $X = X_1 \times X_2 \times \dots \times X_N$ , где  $X_l = \{(\bar{x}_l, j_l) = (x_{l1}, x_{l2}, \dots, x_{lM}, j_l), x_{lu} = 0, 1, 2, \dots; l = \overline{1, N}, u = \overline{1, M}, j_l = \overline{0, r_l}\}$ , в случае открытой сети и конечным фазовым пространством состояний  $Y = (x \in X_1 \times X_2 \times \dots \times X_N : |x_1| + |x_2| + \dots + |x_N| = \overline{N} < \infty)$  в случае замкнутой сети. Здесь  $x_{lu}$  – число заявок типа  $u$  в  $l$ -м узле,  $|x_l| = \sum_{u=1}^M x_{lu}$  – общее число заявок в  $l$ -м узле.

Обозначим через  $(\bar{0}, 0)$  такое состояние  $l$ -го узла, когда в нем отсутствуют заявки и узел функционирует в режиме работы 0.

Количество работы по обслуживанию поступившей в  $l$ -й узел заявки  $u$ -го типа является случайной величиной с произвольной функцией распределения  $B_{lu}(\tilde{u})$  и математическим ожиданием  $\tau_{lu} < \infty$ .

Пусть в момент времени  $t$  состояние  $l$ -го узла есть вектор  $(\bar{x}_l, j_l)$ . Тогда работа по обслуживанию заявки типа  $u$  выполняется со скоростью  $\alpha_{lu}(x_{lu}, |x_l|, j_l) = \alpha_{lu}(x_{lu}, j_l) \frac{x_{lu}}{|x_l|}$ , если в узле находится всего  $|x_l|$  заявок,  $x_{lu}$  заявок типа  $u$  и узел работает в режиме  $j_l$

( $l = \overline{1, N}, u = \overline{1, M}, j_l = \overline{0, r_l}$ ). При этом полагается, что  $\alpha_{lu}(x_{lu}, |x_l|, j_l) > 0$ , если  $x_{lu} \neq 0$ ;  $\alpha_{lu}(x_{lu}, |x_l|, j_l) = 0$ , если  $x_{lu} = 0$  или  $|x_l| = 0$ .

Пусть  $v_l(\bar{x}_l, j_l)$  – скорость выполнения работы по переходу  $l$ -го узла из режима  $j_l$  в режим  $j_l + 1$  ( $j_l = \overline{0, r_l - 1}$ );  $\varphi_l(\bar{x}_l, j_l)$  – скорость выполнения работы по переходу  $l$ -го узла из режима  $j_l$  в режим  $j_l - 1$  ( $j_l = \overline{1, r_l}$ ). Полагается, что  $v_l(\bar{x}_l, r_l) = 0$ ,  $\varphi_l(\bar{x}_l, 0) = 0$ .

Количество работы, необходимое для перехода прибора  $l$ -го узла из основного (нулевого) режима в режим 1, является случайной величиной с произвольной функцией распределения  $\Phi_l(0, \tilde{u})$  и математическим ожиданием  $\eta_l(0) < \infty$ . При этом если в момент времени  $t$  состояние узла есть вектор  $(\bar{x}_l, 0)$ , то работа по осуществлению указанного перехода выполняется со скоростью  $v_l(\bar{x}_l, 0)$ .

Для состояния  $(\bar{x}_l, j_l)$ , у которого  $1 \leq j_l \leq r_l - 1$ , количество работы, необходимое для изменения режима (на  $j_l - 1$  или  $j_l + 1$ ), также является случайной величиной с произвольной функцией распределения  $\Phi_l(j_l, \tilde{u})$  и математическим ожиданием  $\eta_l(j_l) < \infty$ . Если в момент времени  $t$  состояние узла есть  $(\bar{x}_l, j_l)$ , то работа по изменению режима выполняется со скоростью  $v_l(\bar{x}_l, j_l) + \varphi_l(\bar{x}_l, j_l)$ , при этом с вероятностью  $\frac{v_l(\bar{x}_l, j_l)}{v_l(\bar{x}_l, j_l) + \varphi_l(\bar{x}_l, j_l)}$  прибор  $l$ -го узла переходит в режим  $j_l + 1$ , а с вероятностью  $\frac{\varphi_l(\bar{x}_l, j_l)}{v_l(\bar{x}_l, j_l) + \varphi_l(\bar{x}_l, j_l)}$  – в режим  $j_l - 1$ .

Аналогично количество работы, необходимое для перехода прибора  $l$ -го узла из режима  $r_l$  в  $r_l - 1$ , имеет произвольную функцию распределения  $\Phi_l(r_l, \tilde{u})$  и математическое ожидание  $\eta_l(r_l) < \infty$ . При этом если в момент времени  $t$  состояние узла –  $(\bar{x}_l, r_l)$ , то работа по осуществлению указанного перехода выполняется со скоростью  $\varphi_l(\bar{x}_l, r_l)$ .

Полагаем, что матрица маршрутизации  $(p_{(l,u)(k,v)})$ ,  $u, v = \overline{1, M}$ , неприводима (в случае открытой сети  $l, k = \overline{0, N}$ ,  $p_{(0,u)(0,v)} = 0$ , а в случае замкнутой сети  $l, k = \overline{1, N}$ ).

Тогда система уравнений трафика в случае открытой сети принимает вид

$$\varepsilon_{lu} = \lambda p_{0(l,u)} + \sum_{k=1}^N \sum_{v=1}^M \varepsilon_{kv} p_{(k,v)(l,u)}, \quad l = \overline{1, N}, \quad u = \overline{1, M}, \quad (1)$$

где  $\varepsilon_{lu}$  – средняя интенсивность поступления в  $l$ -й узел заявок типа  $u$ .

Данная система уравнений трафика имеет единственное положительное решение  $(\varepsilon_{lu}, l = \overline{1, N}, u = \overline{1, M})$ , что можно доказать, перенумеровав соответствующим образом элементы матрицы вероятностей переходов. В результате получим систему уравнений трафика сети Джексона, для которой доказано существование единственного положительного решения [9].

Аналогично в случае замкнутой сети система уравнений трафика

$$\varepsilon_{lu} = \sum_{k=1}^N \sum_{v=1}^M \varepsilon_{kv} p_{(k,v)(l,u)}, \quad l = \overline{1, N}, \quad u = \overline{1, M}, \quad (2)$$

имеет единственное с точностью до постоянного множителя положительное решение  $(\varepsilon_{lu}, l = \overline{1, N}, u = \overline{1, M})$ .

При данном описании очередь в традиционном ее понимании отсутствует. Однако заявки, поступающие в узел и уже находящиеся там на обслуживании, можно пронумеровать в зависимости от количества выполненной работы по их обслуживанию. Таким образом, состояние  $l$ -го узла в момент времени  $t$  может быть уточнено вектором  $x_l(t) = (\bar{x}_l(t), j_l(t)) = (x_{l1}(t), x_{l2}(t), \dots, x_{lM}(t), j_l(t)) = (x_{l11}(t), x_{l12}(t), \dots, x_{l1r_{l1}}(t); x_{l21}(t), x_{l22}(t), \dots, x_{l2r_{l2}}(t); \dots; x_{lM1}(t), x_{lM2}(t), \dots,$

$x_{lMx_{lM}}(t); j_l(t)$ , где  $x_{lu1}(t)$  – заявка типа  $u$ , поступившая в  $l$ -й узел первой среди находящихся там в момент времени  $t$  заявок типа  $u$ ;  $x_{lux_{lu}}(t)$  – заявка типа  $u$ , поступившая в  $l$ -й узел последней среди находящихся там в момент времени  $t$  заявок типа  $u$ ,  $u = \overline{1, M}$ .

Пусть  $\psi_{lmn}(t)$  – количество работы, которое осталось выполнить с момента  $t$  для завершения обслуживания заявки типа  $u$ , стоящей на позиции  $n$  в  $l$ -м узле,  $\psi_l(t) = (\psi_{l11}(t), \psi_{l12}(t), \dots, \psi_{l1x_{l1}}(t); \psi_{l21}(t), \psi_{l22}(t), \dots, \psi_{l2x_{l2}}(t); \dots; \psi_{lM1}(t), \psi_{lM2}(t), \dots, \psi_{lMx_{lM}}(t))$ ,  $l = \overline{1, N}$ .

Пусть  $\xi_{lj_l}(t)$  – количество работы, которое осталось выполнить с момента  $t$  для перехода прибора  $l$ -го узла из режима  $j_l$  в соседний режим,  $\xi(t) = (\xi_{1,j_1(t)}(t), \xi_{2,j_2(t)}(t), \dots, \xi_{N,j_N(t)}(t))$ .

Таким образом,

$$\frac{\partial \psi_{lmn}(t)}{\partial t} = -\alpha_{lu}(x_{lu}, j_l) \frac{x_{lu}}{|x_l|};$$

$$\frac{\partial \xi_{lj_l}(t)}{\partial t} = -\left( v_l(\bar{x}_l, j_l) I_{(j_l \neq n)} + \phi_l(\bar{x}_l, j_l) I_{(j_l \neq 0)} \right),$$

когда состояние  $l$ -го узла есть  $(\bar{x}_l, j_l)$ . Здесь  $I_A$  – индикатор события  $A$ , равный 1, если событие  $A$  происходит, и равный 0, если событие  $A$  не происходит.

В общем случае процесс  $x(t)$  не является марковским, поэтому рассмотрим кусочно-линейный марковский процесс  $\zeta(t) = (x(t), \psi(t), \xi(t))$ , полученный путем добавления к  $x(t)$  непрерывных компонент  $\psi(t) = (\psi_1(t), \psi_2(t), \dots, \psi_N(t))$  и  $\xi(t)$ .

Под  $P = \{P(x)\}$  будем понимать стационарное распределение вероятностей состояний процесса  $x(t)$ .

Введем в рассмотрение стационарные функции распределения вероятностей состояний кусочно-линейного процесса  $\zeta(t)$ :

$$F(x, y, z) = F(x, y_{111}, y_{112}, \dots, y_{11x_{l1}}; y_{121}, y_{122}, \dots, y_{12x_{l2}}; \dots; y_{lM1}, y_{lM2}, \dots, y_{lMx_{lM}};$$

$$y_{211}, y_{212}, \dots, y_{21x_{21}}; y_{221}, y_{222}, \dots, y_{22x_{22}}; \dots; y_{2M1}, y_{2M2}, \dots, y_{2Mx_{2M}}; \dots;$$

$$y_{N11}, y_{N12}, \dots, y_{N1x_{N1}}; y_{N21}, y_{N22}, \dots, y_{N2x_{N2}}; \dots; y_{NM1}, y_{NM2}, \dots, y_{NMx_{NM}}; z_{1,j_1}, \dots, z_{N,j_N}) =$$

$$= \lim_{t \rightarrow \infty} P\{x(t) = x; \psi_{l11}(t) < y_{l11}, \psi_{l12}(t) < y_{l12}, \dots, \psi_{l1x_{l1}}(t) < y_{l1x_{l1}}; \dots;$$

$$\psi_{lM1}(t) < y_{lM1}, \psi_{lM2}(t) < y_{lM2}, \dots, \psi_{lMx_{lM}}(t) < y_{lMx_{lM}}, l = \overline{1, N};$$

$$\xi_{1,j_1(t)}(t) < z_{1,j_1}, \dots, \xi_{N,j_N(t)}(t) < z_{N,j_N}\}.$$

Обозначим  $\vartheta_l(\bar{x}_l, j_l) = v_l(\bar{x}_l, j_l) I_{(j_l \neq n)} + \phi_l(\bar{x}_l, j_l) I_{(j_l \neq 0)}$ .

## 2. Основной результат для открытой сети

В работе [10] был рассмотрен случай, когда длительности обслуживания заявок имеют экспоненциальное распределение, т. е. для  $l$ -го узла  $B_{lu}(\tilde{u}) = 1 - \exp\{-\mu_{lu}(x_{lu}, |x_l|, j_l)\tilde{u}\}$  ( $\tilde{u} > 0$ ), где  $\mu_{lu}(x_{lu}, |x_l|, j_l)$  – интенсивность обслуживания заявок типа  $u$  в  $l$ -м узле. Тогда  $x(t)$  – марковский процесс с непрерывным временем. Установлено, что при выполнении условий

$$v_l(x_{l1}, x_{l2}, \dots, x_{lM}, j_l - 1) \mu_{lu}(x_{lu}, j_l) \phi_l(x_{l1}, \dots, x_{lu} - 1, \dots, x_{lM}, j_l) =$$

$$= v_l(x_{l1}, \dots, x_{lu} - 1, \dots, x_{lM}, j_l - 1) \mu_{lu}(x_{lu}, j_l - 1) \phi_l(x_{l1}, x_{l2}, \dots, x_{lM}, j_l), \quad (3)$$

$$x_{lu} \neq 0, u = \overline{1, M}, l = \overline{1, N}, j_l = \overline{1, n_l};$$

$$\sum_{x \in X} q(x) \prod_{l=1}^N |x_l|! \prod_{u=1}^M \frac{\varepsilon_{lu}^{x_{lu}}}{x_{lu}!} \prod_{w=1}^{x_{lu}} \frac{1}{\mu_{lu}(w, j_l)} \prod_{k=1}^{j_l} \frac{v_l(\bar{0}, k-1)}{\phi_l(\bar{0}, k)} < \infty, \quad (4)$$

$$q(x) = \lambda + \sum_{l=1}^N (v_l(\bar{x}_l, j_l) + \phi_l(\bar{x}_l, j_l)) + \sum_{l=1}^N \sum_{u=1}^M \mu_{lu}(x_{lu}, j_l) \frac{x_{lu}}{|x_l|},$$

марковский процесс  $x(t)$  эргодичен, а его стационарное распределение имеет мультипликативную форму

$$P(x) = p_1(x_1) p_2(x_2) \times \dots \times p_N(x_N),$$

где  $p_l(\bar{x}_l, j_l) = |x_l|! \prod_{u=1}^M \frac{\varepsilon_{lu}^{x_{lu}}}{x_{lu}!} \prod_{w=1}^{x_{lu}} \frac{1}{\mu_{lu}(w, j_l)} \prod_{k=1}^{j_l} \frac{v_l(\bar{0}, k-1)}{\phi_l(\bar{0}, k)} p_l(\bar{0}, 0)$ ,  $\varepsilon_{lu}$  находятся из (1), а

$$p_l(\bar{0}, 0) = \left( \sum_{x_l \in X_l} |x_l|! \prod_{u=1}^M \frac{\varepsilon_{lu}^{x_{lu}}}{x_{lu}!} \prod_{w=1}^{x_{lu}} \frac{1}{\mu_{lu}(w, j_l)} \prod_{k=1}^{j_l} \frac{v_l(\bar{0}, k-1)}{\phi_l(\bar{0}, k)} \right)^{-1}.$$

Для описанных выше открытых сетей в случае, когда количество работы по обслуживанию поступившей в узел заявки имеет произвольную функцию распределения  $B_{lu}(\tilde{u})$ , а количество работы по переключению режима работы прибора в узле – произвольную функцию распределения  $\Phi_l(k, \tilde{u})$ , справедлива следующая теорема.

**Теорема 1.** Если выполнены условия

$$\begin{aligned} v_l(x_{l1}, x_{l2}, \dots, x_{lM}, j_l - 1) \alpha_{lu}(x_{lu}, j_l) \phi_l(x_{l1}, \dots, x_{lu} - 1, \dots, x_{lM}, j_l) = \\ = v_l(x_{l1}, \dots, x_{lu} - 1, \dots, x_{lM}, j_l - 1) \alpha_{lu}(x_{lu}, j_l - 1) \phi_l(x_{l1}, x_{l2}, \dots, x_{lM}, j_l), \end{aligned} \quad (5)$$

$$x_{lu} \neq 0, \quad u = \overline{1, M}, \quad l = \overline{1, N}, \quad j_l = \overline{1, r_l};$$

$$\sum_{x \in X} q(x) \prod_{l=1}^N |x_l|! \prod_{u=1}^M \frac{\varepsilon_{lu}^{x_{lu}}}{x_{lu}!} \prod_{w=1}^{x_{lu}} \frac{\tau_{lu}}{\alpha_{lu}(w, j_l)} \prod_{k=1}^{j_l} \frac{v_l(\bar{0}, k-1)}{\phi_l(\bar{0}, k)} < \infty, \quad (6)$$

где  $(\varepsilon_{lu}, l = \overline{1, N}, u = \overline{1, M})$  – решение системы уравнений трафика (1);

$$q(x) = \lambda + \sum_{l=1}^N \left( \eta_l^{-1}(j_l) v_l(\bar{x}_l, j_l) + \eta_l^{-1}(j_l) \phi_l(\bar{x}_l, j_l) + \sum_{u=1}^M \frac{\alpha_{lu}(x_{lu}, j_l)}{\tau_{lu}} \frac{x_{lu}}{|x_l|} \right) - \text{интенсивность выхода из}$$

состояния  $x$ , то процесс  $\zeta(t)$  эргодичен, при этом стационарные функции распределения вероятностей состояний  $F(x, y, z)$  определяются по формулам

$$\begin{aligned} F(x, y, z) = p_1(x_1) p_2(x_2) \times \dots \times p_N(x_N) \times \\ \times \prod_{l=1}^N \prod_{u=1}^M \prod_{w=1}^{x_{lu}} \tau_{lu}^{-1} \int_0^{y_{luw}} (1 - B_{lu}(\tilde{u})) d\tilde{u} \prod_{l=1}^N \eta_l^{-1}(j_l) \int_0^{z_{l,j_l}} (1 - \Phi_l(j_l, \tilde{u})) d\tilde{u}, \quad x \in X, \end{aligned} \quad (7)$$

где

$$p_l(\bar{x}_l, j_l) = \frac{\eta_l(j_l)}{\eta_l(0)} |x_l|! \prod_{u=1}^M \frac{\varepsilon_{lu}^{x_{lu}}}{x_{lu}!} \prod_{w=1}^{x_{lu}} \frac{\tau_{lu}}{\alpha_{lu}(w, j_l)} \prod_{k=1}^{j_l} \frac{v_l(\bar{0}, k-1)}{\phi_l(\bar{0}, k)} p_l(\bar{0}, 0); \quad (8)$$

$$p_l(\bar{0}, 0) = \eta_l(0) \left[ \sum_{x_l \in X_l} |x_l|! \eta_l(j_l) \prod_{u=1}^M \frac{\varepsilon_{lu}^{x_{lu}}}{x_{lu}!} \prod_{w=1}^{x_{lu}} \frac{\tau_{lu}}{\alpha_{lu}(w, j_l)} \prod_{k=1}^{j_l} \frac{v_l(\bar{0}, k-1)}{\phi_l(\bar{0}, k)} \right]^{-1}, \quad (9)$$

$$x \in X, l = \overline{1, N}.$$

**Доказательство.** Пусть выполнены условия (3), (4), т. е. в случае, когда  $x(t)$  – марковский процесс, существует стационарное эргодическое распределение  $x(t)$ . Тогда, по-видимому, и в

общем случае при выполнении условий (5), (6) существует стационарное эргодическое распределение процесса  $\zeta(t)$ , так как  $\zeta(t)$  получается из  $x(t)$  добавлением непрерывных компонент, а  $\mu_{lu}^{-1}(x_{lu}, |x_l, j_l) = \tau_{lu} \alpha_{lu}^{-1}(x_{lu}, |x_l, j_l)$ . Строгое доказательство этого факта может быть проведено, если учесть, что процесс  $\zeta(t)$  является регенерирующим. Функционирование сети схематично можно представить как чередование периодов, когда сеть находится в состоянии 0 (в каждом узле сети нет заявок и прибор работает в нулевом режиме), и периодов занятости сети (в противном случае). Далее доказательство сводится к применению предельной теоремы Смита для регенерирующих процессов [11, с. 41], при этом учитывается, что среднее время обслуживания заявки равно среднему времени обслуживания заявки в марковском случае, а среднее время переключения режима прибора равно среднему времени переключения режима прибора в марковском случае.

Для упрощения процедуры доказательства введем в рассмотрение некоторые вспомогательные обозначения и операторы:

$$\begin{aligned} x \pm e_{lu} &= (x_1, \dots, (x_{l1}, \dots, x_{lu} \pm 1, \dots, x_{lM}, j_l), \dots, x_N); \\ x + e_{lu} - e_{kv} &= (x_1, \dots, (x_{l1}, \dots, x_{lu} + 1, \dots, x_{lM}, j_l), \dots, (x_{k1}, \dots, x_{kv} - 1, \dots, x_{kM}, j_k), \dots, x_N); \\ x \pm e'_l &= (x_1, \dots, (x_{l1}, \dots, x_{lM}, j_l \pm 1), \dots, x_N); \\ T_{(l,u,n)}^+(y, z_{lun}) &= (\tilde{y}_1, \tilde{y}_2, \dots, \tilde{y}_N), \\ \tilde{y}_k &= (y_{k11}, y_{k12}, \dots, y_{k1x_{k1}}; y_{k21}, y_{k22}, \dots, y_{k2x_{k2}}; \dots; y_{kM1}, y_{kM2}, \dots, y_{kMx_{kM}}), k \neq l, \\ \tilde{y}_l &= (y_{l11}, \dots, y_{l1x_{l1}}; y_{l21}, \dots, y_{l2x_{l2}}; \dots; y_{lu1}, \dots, y_{l,u,n-1}, t_{lun}, y_{lun}, \dots, y_{l,ux_{lu}}; \dots; y_{lM1}, \dots, y_{lMx_{lM}}), \\ n &= \overline{1, x_{lu} + 1}; \\ T_{(l,u,n)}^-(y) &= (\tilde{y}_1, \tilde{y}_2, \dots, \tilde{y}_N), \\ \tilde{y}_k &= (y_{k11}, y_{k12}, \dots, y_{k1x_{k1}}; y_{k21}, y_{k22}, \dots, y_{k2x_{k2}}; \dots; y_{kM1}, y_{kM2}, \dots, y_{kMx_{kM}}), k \neq l, \\ \tilde{y}_l &= (y_{l11}, \dots, y_{l1x_{l1}}; y_{l21}, \dots, y_{l2x_{l2}}; \dots; y_{lu1}, \dots, y_{l,u,n-1}, y_{l,u,n+1}, \dots, y_{l,ux_{lu}}; \dots; y_{lM1}, \dots, y_{lMx_{lM}}), \\ n &= \overline{1, x_{lu}}. \end{aligned}$$

Для  $F(x, y, z)$  справедлива следующая система дифференциально-разностных уравнений:

$$\begin{aligned} \lambda F(x, y, z) &= \sum_{l=1}^N \sum_{u=1}^M \sum_{n=1}^{x_{lu}} \alpha_{lu}(x_{lu}, j_l) \frac{x_{lu}}{|x_l|} \left( \frac{\partial F(x, y, z)}{\partial y_{lun}} - \left( \frac{\partial F(x, y, z)}{\partial y_{lun}} \right)_{y_{lun}=0} \right) + \\ &+ \sum_{l=1}^N \mathfrak{G}_l(\bar{x}_l, j_l) \left( \frac{\partial F(x, y, z)}{\partial z_{l,j_l}} - \left( \frac{\partial F(x, y, z)}{\partial z_{l,j_l}} \right)_{z_{l,j_l}=0} \right) + \lambda \sum_{l=1}^N \sum_{u=1}^M \sum_{n=1}^{x_{lu}} p_{0(l,u)} B_{lu}(y_{lun}) F(x - e_{lu}, T_{(l,u,n)}^-(y), z) + \\ &+ \sum_{l=1}^N \sum_{u=1}^M \sum_{n=1}^{x_{lu}+1} \alpha_{lu}(x_{lu} + 1, j_l) \frac{x_{lu} + 1}{|x_l| + 1} p_{(l,u)0} \left( \frac{\partial F(x + e_{lu}, T_{(l,u,n)}^+(y, t_{lun}), z)}{\partial t_{lun}} \right)_{t_{lun}=0} + \\ &+ \sum_{l=1}^N \sum_{s=1, s \neq l}^N \sum_{v=1}^M \sum_{m=1}^{x_{sv}+1} \sum_{n=1}^{x_{lv}} \alpha_{sv}(x_{sv} + 1, j_s) \frac{x_{sv} + 1}{|x_s| + 1} p_{(s,u)(l,v)} B_{lv}(y_{lvm}) \left( \frac{\partial F(x + e_{su} - e_{lv}, T_{(s,u,m)}^+(T_{(l,v,n)}^-(y, t_{lvm})), z)}{\partial t_{sum}} \right)_{t_{lm}=0} + \\ &+ \sum_{l=1}^N \sum_{u,v=1}^M \sum_{m=1}^{x_{lu}} \sum_{n=1}^{x_{lv}} \alpha_{lu}(x_{lu} + 1, j_l) \frac{x_{lu} + 1}{|x_l|} p_{(l,u)(l,v)} B_{lv}(y_{lvm}) \left( \frac{\partial F(x + e_{lu} - e_{lv}, T_{(l,u,m)}^+(T_{(l,v,n)}^-(y, t_{lvm})), z)}{\partial t_{lum}} \right)_{t_{lm}=0} + \\ &+ \sum_{l=1}^N \left( v_l(\bar{x}_l, j_l - 1) \Phi_l(j_l, z_{l,j_l}) \left( \frac{\partial F(x - e'_l, y, z)}{\partial z_{l,j_l-1}} \right)_{z_{l,j_l-1}=0} + \phi_l(\bar{x}_l, j_l + 1) \Phi_l(j_l, z_{l,j_l}) \left( \frac{\partial F(x + e'_l, y, z)}{\partial z_{l,j_l+1}} \right)_{z_{l,j_l+1}=0} \right), \end{aligned} \quad (10)$$

$$x \in X.$$

В полученных уравнениях предполагается, что если аргумент функции  $F(x, y, z)$  не принадлежит фазовому пространству, т. е.  $x \notin X$ , то  $F(x, y, z) = 0$ .

Разобьем эту систему уравнений на уравнения локального баланса следующим образом:

$$\lambda F(x, y, z) = \sum_{l=1}^N \sum_{u=1}^M \sum_{n=1}^{x_{lu}+1} \alpha_{lu}(x_{lu}+1, j_l) \frac{x_{lu}+1}{|x_l|+1} p_{(l,u)0} \left( \frac{\partial F(x + e_{lu}, T_{(l,u,n)}^+(y, t_{lun}), z)}{\partial t_{lun}} \right)_{t_{lun}=0}; \quad (11)$$

$$\begin{aligned} & \sum_{l=1}^N \sum_{u=1}^M \sum_{n=1}^{x_{lu}} \alpha_{lu}(x_{lu}, j_l) \frac{x_{lu}}{|x_l|} \left( \left( \frac{\partial F(x, y, z)}{\partial y_{lun}} \right)_{y_{lun}=0} - \frac{\partial F(x, y, z)}{\partial y_{lun}} \right) = \\ & = \lambda p_{0(l,u)} B_{lu}(y_{lun}) F(x - e_{lu}, T_{(l,u,n)}^-(y), z) + \\ & + \sum_{s=1, s \neq l}^N \sum_{u=1}^M \sum_{m=1}^{x_{su}+1} \alpha_{su}(x_{su}+1, j_s) \frac{x_{su}+1}{|x_s|+1} p_{(s,u)(l,v)} B_{lv}(y_{lvn}) \left( \frac{\partial F(x + e_{su} - e_{lv}, T_{(s,u,m)}^+(T_{(l,v,n)}^-(y, t_{lvn})), z)}{\partial t_{sum}} \right)_{t_{lvn}=0} + \end{aligned} \quad (12)$$

$$\begin{aligned} & + \sum_{u=1}^M \sum_{m=1}^{x_{lu}} \alpha_{lu}(x_{lu}+1, j_l) \frac{x_{lu}+1}{|x_l|} p_{(l,u)(l,v)} B_{lv}(y_{lvn}) \left( \frac{\partial F(x + e_{lu} - e_{lv}, T_{(l,u,m)}^+(T_{(l,v,n)}^-(y, t_{lvn})), z)}{\partial t_{lum}} \right)_{t_{lvn}=0}; \\ & \mathfrak{D}_l(\bar{x}_l, j_l) \left( \left( \frac{\partial F(x, y, z)}{\partial z_{l,j_l}} \right)_{z_{l,j_l}=0} - \frac{\partial F(x, y, z)}{\partial z_{l,j_l}} \right) = \\ & = v_l(\bar{x}_l, j_l - 1) \Phi_l(j_l, z_{l,j_l}) \left( \frac{\partial F(x - e'_l, y, z)}{\partial z_{l,j_l-1}} \right)_{z_{l,j_l-1}=0} + \varphi_l(\bar{x}_l, j_l + 1) \Phi_l(j_l, z_{l,j_l}) \left( \frac{\partial F(x + e'_l, y, z)}{\partial z_{l,j_l+1}} \right)_{z_{l,j_l+1}=0}, \quad (13) \\ & l = \overline{1, N}, x \in X. \end{aligned}$$

Функции распределения вероятностей  $F(x, y, z)$ , определенные формулами (7)–(9), являются решением уравнений (11)–(13) и, значит, уравнений (10).

Действительно, подставим (7) в (11) и разделим обе части полученного соотношения на  $F(x, y, z)$ . Результатом будет являться следствие уравнения трафика  $\lambda = \sum_{l=1}^N \sum_{u=1}^M \varepsilon_{lu} p_{(l,u)0}$ .

Подставим (7) в (12), приведем подобные слагаемые и разделим обе части полученного соотношения на  $B_{lu}(y_{lun}) F(x - e_{lu}, T_{(l,u,n)}^-(y), z)$ . Результатом будет являться уравнение трафика (1).

Наконец, подставим (7) в (13). Учитывая (5), получим тождество.

Теорема 1 доказана.

**Следствие 1.** Если выполняются соотношения (5), (6), то процесс  $x(t)$  эргодичен, а его стационарное распределение  $P = \{P(x), x \in X\}$  не зависит от вида функций распределения  $B_l(\tilde{u})$ ,  $\Phi_l(k, \tilde{u})$  и имеет вид

$$P(x) = p_1(x_1) p_2(x_2) \times \dots \times p_N(x_N),$$

где  $p_l(x_l)$  определяются по формулам (8), (9).

### 3. Основной результат для замкнутой сети

Для замкнутой сети справедлива следующая теорема.

**Теорема 2.** Процесс  $\zeta(t)$  эргодичен. Если выполняются соотношения

$$\begin{aligned} & v_l(x_{l_1}, x_{l_2}, \dots, x_{l_M}, j_l - 1) \alpha_{lu}(x_{lu}, j_l) \Phi_l(x_{l_1}, \dots, x_{l_u} - 1, \dots, x_{l_M}, j_l) = \\ & = v_l(x_{l_1}, \dots, x_{l_u} - 1, \dots, x_{l_M}, j_l - 1) \alpha_{lu}(x_{lu}, j_l - 1) \Phi_l(x_{l_1}, x_{l_2}, \dots, x_{l_M}, j_l), \\ & x_{lu} \neq 0, u = \overline{1, M}, l = \overline{1, N}, j_l = \overline{1, r_l}, \sum_{l=1}^N |x_l| = \overline{N}, \end{aligned} \quad (14)$$

то стационарные функции распределения вероятностей состояний  $F(x, y, z)$  определяются по формулам

$$F(x, y, z) = C(N, M, \bar{N}) p_1(x_1) p_2(x_2) \times \dots \times p_N(x_N) \times \\ \times \prod_{l=1}^N \prod_{u=1}^M \prod_{w=1}^{x_{lu}} \tau_{lu}^{-1} \int_0^{y_{luw}} (1 - B_{lu}(\tilde{u})) d\tilde{u} \prod_{l=1}^N \eta_l^{-1}(j_l) \int_0^{z_{l,j_l}} (1 - \Phi_l(j_l, \tilde{u})) d\tilde{u}, x \in Y,$$

где

$$p_l(\bar{x}_l, j_l) = \eta_l(j_l) |x_l|! \prod_{u=1}^M \frac{\varepsilon_{lu}^{x_{lu}}}{x_{lu}!} \prod_{w=1}^{x_{lu}} \frac{\tau_{lu}}{\alpha_{lu}(w, j_l)} \prod_{k=1}^{j_l} \frac{v_l(\bar{0}, k-1)}{\phi_l(\bar{0}, k)}, \quad (15)$$

$\varepsilon_{lu}$  находятся из (2) и

$$C(N, M, \bar{N})^{-1} = \sum_{\substack{|x_1|, |x_2|, \dots, |x_N| \geq 0, \\ |x_1| + |x_2| + \dots + |x_N| = \bar{N}}} \sum_{j_1=0}^{r_1} \sum_{j_2=0}^{r_2} \dots \sum_{j_N=0}^{r_N} \prod_{l=1}^N \eta_l(j_l) |x_l|! \prod_{u=1}^M \frac{\varepsilon_{lu}^{x_{lu}}}{x_{lu}!} \prod_{w=1}^{x_{lu}} \frac{\tau_{lu}}{\alpha_{lu}(w, j_l)} \prod_{k=1}^{j_l} \frac{v_l(\bar{0}, k-1)}{\phi_l(\bar{0}, k)}, \quad (16)$$

$$l = \bar{1}, \bar{N}, \quad x \in Y.$$

Доказательство теоремы 2 аналогично доказательству теоремы 1.

Из теоремы 2 вытекает следующее утверждение.

**Следствие 2.** Процесс  $x(t)$  эргодичен. Если выполняются соотношения (14), то его стационарное распределение  $P = \{P(x), x \in Y\}$  не зависит от функционального вида распределений  $B_l(\tilde{u})$ ,  $\Phi_l(k, \tilde{u})$  и имеет вид

$$P(x) = C(N, M, \bar{N}) p_1(x_1) p_2(x_2) \times \dots \times p_N(x_N),$$

где  $p_l(x_l)$  определяются по формулам (15), а  $C(N, M, \bar{N})$  – по формуле (16).

### Заключение

Исследованы открытые и замкнутые сети массового обслуживания, в которых циркулируют заявки нескольких типов. В случае открытой сети входной поток заявок – простейший. В каждом из узлов находится единственный прибор, который может работать в нескольких режимах. Дисциплина обслуживания заявок прибором – «дискриминаторное разделение процесса» (DPS). Количество работы по обслуживанию заявки и количество работы по переключению режима прибора в узле являются случайными величинами с произвольными функциями распределения.

Установлены условия нечувствительности стационарного распределения вероятностей состояний указанных сетей к виду законов распределения величин работ, требующихся на обслуживание заявок и переключение режимов приборов в узлах, если фиксированы первые моменты этих законов. Также определено, что стационарное распределение сети имеет форму произведения, где каждый множитель есть стационарное распределение изолированного узла, помещенного в фиктивную окружающую среду с пуассоновским входящим потоком. Этот результат был получен с помощью метода расширения фазового пространства, когда процесс, описывающий поведение сети (вообще говоря, немарковский), дополняют непрерывными компонентами.

Автор выражает благодарность своему научному руководителю, профессору Ю.В. Малинковскому, за полезные обсуждения и помощь в анализе полученных результатов.

### Список литературы

1. Вишневский, В.М. Теоретические основы проектирования компьютерных сетей / В.М. Вишневский. – М. : Техносфера, 2003. – 506 с.

2. Матальцкий, М.А. Теория массового обслуживания и ее применения / М.А. Матальцкий, О.М. Тихоненко, А.В. Паньков. – Гродно : ГрГУ, 2008. – 771 с.
3. Малинковский, Ю.В. Замкнутые информационные сети с многорежимными стратегиями обслуживания / Ю.В. Малинковский, А.Ю. Нуеман // Материалы I Междунар. конф. «Информационные системы и технологии» (IST'2002). – Минск, 2002. – Ч. 1. – С. 324–328.
4. Малинковский, Ю.В. Мультипликативность стационарного распределения в открытых сетях с многорежимными стратегиями обслуживания / Ю.В. Малинковский, А.Ю. Нуеман // Вестні НАН Беларусі. Сер. фіз.-мат. навук. – 2001. – № 3. – С. 129–134.
5. Нуеман, А.Ю. Открытые сети с многорежимными стратегиями обслуживания и отрицательными заявками / А.Ю. Нуеман // Вестник ТГУ. – 2002. – № 1. – С. 90–93.
6. Яшков, С.Ф. Математические вопросы теории систем обслуживания с разделением процессора / С.Ф. Яшков // Итоги науки и техники. Сер. Теор. вероятн. Матем. статистика. Теор. кибернетика. – М. : ВИНТИ, 1990. – № 29. – С. 3–82.
7. Chandy, K.M. Product form and local balance in queueing networks / K.M. Chandy, I.H. Howard, D.F. Towsley // J-ACM. – 1977. – № 2. – P. 250–263.
8. Kelly, F.P. Networks of queues / F.P. Kelly // Adv. Appl. Probability. – 1976. – № 2. – P. 416–432.
9. Jackson, J.R. Jobshop-like Queueing Systems / J.R. Jackson // Manag. Sci. – 1963. – Vol. 10, № 1. – P. 131–142.
10. Летунович, Ю.Е. Стационарное распределение вероятностей состояний сетей с многорежимными стратегиями обслуживания и несколькими типами заявок : автореф. дис. ... к-та физ.-мат. наук: 01.01.05 / Ю.Е. Летунович. – Минск : БГУ, 2011. – 22 с.
11. Ивницкий, В.А. Теория сетей массового обслуживания / В.А. Ивницкий. – М. : Физматлит, 2004. – 772 с.

Поступила 15.02.12

*Гродненский государственный  
университет им. Я. Купалы,  
Гродно, ул. Ожешко, 22  
e-mail: a.eremina@grsu.by*

**A.R. Eryomina**

**INVARIANCE OF STATIONARY STATE DISTRIBUTION  
OF HETEROGENEOUS QUEUEING NETWORKS  
WITH MULTIMODE STRATEGIES AND DPS DISPATCHING RULE**

Open and closed queueing networks with multimode strategies, polytypic demands and Discriminatory Processor-Sharing dispatching rule are considered. It is established that the stationary distribution is invariant with regard to functional form of distribution of work quantities, which are required for servicing and switching, provided that the expectation values are fixed.

УДК 519.8

В.А. Емеличев, В.В. Коротков

## АНАЛИЗ УСТОЙЧИВОСТИ ПАРЕТО-ОПТИМАЛЬНОГО ПОРТФЕЛЯ ПРОЕКТОВ БИКРИТЕРИАЛЬНОЙ ИНВЕСТИЦИОННОЙ ЗАДАЧИ С КРИТЕРИЯМИ ВАЛЬДА И СЭВИДЖА

*Находятся нижняя и верхняя достижимые оценки радиуса устойчивости парето-оптимального портфеля двухкритериальной инвестиционной булевой задачи с максиминным критерием эффективности (доходности) и минимаксным критерием риска упущенной выгоды.*

### Введение

В работе [1] были получены оценки радиуса устойчивости лексикографически оптимального портфеля многокритериальной инвестиционной булевой задачи Марковица с упорядоченными по важности минимаксными критериями рисков Сэвиджа при условии, что в пространствах портфелей, рисков и состояний рынка задана одна и та же октаэдральная норма  $l_1$ . В настоящей работе устанавливаются нижняя и верхняя достижимые (неулучшаемые) оценки радиуса устойчивости парето-оптимального портфеля бикритериальной инвестиционной задачи с максиминным критерием Вальда для доходности и минимаксным критерием Сэвиджа для риска упущенной выгоды. При этом анализ устойчивости портфеля к возмущениям исходных данных (оценок эффективности и меры риска) проводится в предположении, что в пространстве портфелей задана октаэдральная норма  $l_1$ , а в пространстве состояний рынка – чебышевская норма  $l_\infty$ . Применение таких норм существенно видоизменяет технику доказательства оценок радиуса устойчивости по сравнению с аналогичной техникой, использованной в [1]. Кроме того, паретовский принцип оптимальности по сравнению с лексикографическим требует учета двух критериев, а не одного первого, как в [1]. И, наконец, разнородность критериев (MAXMIN и MINMAX) создает дополнительную громоздкость в изложении доказательств теорем в настоящей статье.

Отметим, что анализу устойчивости в области теории расписаний и разработке методов решения оптимизационных задач с неопределенными параметрами посвящен ряд работ белорусских авторов [2, 3].

### 1. Постановка задачи и основные определения

Рассмотрим бикритериальный вариант булевой задачи управления инвестициями, основанный на портфельной теории Марковица [4]. Для этого введем ряд обозначений. Пусть  $N_n = \{1, 2, \dots, n\}$  – альтернативные инвестиционные проекты (активы);  $N_m$  – возможные (прогнозные) состояния (ситуации) рынка. Отсюда следует, что  $N_m$  – множество вариантов сценариев развития. Пусть  $x_j = 1$ , если проект  $j \in N_n$  реализуется, и  $x_j = 0$  в противном случае. Инвестиционным портфелем назовем булев вектор  $x = (x_1, x_2, \dots, x_n)^T$ . Через  $X \subset \{0, 1\}^n$  будем обозначать множество всех возможных инвестиционных портфелей, т. е. тех портфелей, реализация которых не превосходит начального капитала инвестора.

Для каждого состояния рынка  $i \in N_m$  инвестиционный портфель  $x \in X$  будем оценивать двумя показателями (аддитивными функциями):  $\sum_{j \in N_n} e_{ij} x_j$  и  $\sum_{j \in N_n} r_{ij} x_j$ . Здесь  $e_{ij}$  – ожидаемая оценка эффективности (чистый доход) проекта  $j \in N_n$  в случае, когда рынок находится в состоянии  $i \in N_m$ ;  $r_{ij}$  – мера риска, которому подвергается инвестор, выбирая проект  $j \in N_n$  при  $i$ -м состоянии рынка. Отметим, что существует несколько подходов при оценке эффективности и риска инвестиционных проектов (см., например, [5–8]). Таким образом,

исходными данными задачи являются две матрицы – матрица эффективности  $E = [e_{ij}] \in \mathbf{R}^{m \times n}$  и матрица рисков  $R = [r_{ij}] \in \mathbf{R}^{m \times n}$ .

Желание инвестора выбрать наиболее прибыльный портфель всегда вступает в противоречие с желанием обеспечить вложения с наименьшим риском. Эти обстоятельства вместе с неопределенностью рынка, недостаточной осведомленностью об условиях, в которых будет проходить выбор проектов, приводят к необходимости использования двух критериев – максиминного критерия эффективности Вальда [9, 10]

$$e(x, E) = \min_{i \in N_m} E_i x = \min_{i \in N_m} \sum_{j \in N_n} e_{ij} x_j \rightarrow \max_{x \in X}$$

и минимаксного критерия риска Сэвиджа [11]

$$r(x, R) = \max_{i \in N_m} R_i x = \max_{i \in N_m} \sum_{j \in N_n} r_{ij} x_j \rightarrow \min_{x \in X},$$

где  $E_i = (e_{i1}, e_{i2}, \dots, e_{in})$ ,  $R_i = (r_{i1}, r_{i2}, \dots, r_{in})$  –  $i$ -е строки матриц  $E$  и  $R$  соответственно.

Таким образом, согласно критерию Вальда инвестор в условиях непредсказуемости состояния рынка выбирает тот портфель, при котором величина суммарной эффективности принимает наибольшее значение в самой неблагоприятной ситуации, а именно тогда, когда эффективность минимальна. Следуя критерию Сэвиджа (критерию «узкого места»), инвестор также проявляет крайнюю осторожность, оптимизируя риск портфеля в предположении, что рынок находится в самом невыгодном для него состоянии. Очевидно, что оба подхода могут быть продиктованы лишь крайним пессимизмом в оценке рыночной ситуации и их использование целесообразно только тогда, когда речь идет о необходимости достижения гарантированного результата.

Отметим, что в теории оптимизации задачи с минимаксными и максиминными критериями занимают видное место [12–14]. Эти критерии характерны, в частности, для моделей игрового характера [15, 16]. Минимаксная концепция оптимизации реализуется и как способ наилучшего чебышевского приближения функции.

Под бикритериальной инвестиционной булевой задачей  $Z^m(E, R)$  будем понимать задачу поиска множества парето-оптимальных инвестиционных портфелей (множества Парето)

$$P^m(E, R) = \{x \in X : \nexists x' \in X (x \vdash_{E, R} x')\},$$

где символ  $\vdash_{E, R}$  обозначает бинарное отношение, задаваемое на множестве портфелей  $X$  формулой

$$x \vdash_{E, R} x' \Leftrightarrow e(x, E) \leq e(x', E) \ \& \ r(x, R) \geq r(x', R) \ \& \ f(x, E, R) \neq f(x', E, R).$$

Здесь  $f(x, E, R) = (e(x, E), r(x, R))$ .

Очевидно, что частным случаем инвестиционной задачи  $Z^m(E, R)$  (при  $m = 1$ ) является двухкритериальная задача линейного булева программирования. Такой случай можно трактовать как ситуацию, при которой состояние рынка не вызывает сомнений.

В пространстве портфелей  $\mathbf{R}^n$  зададим октаэдральную норму  $l_1$ , а в пространстве рыночных состояний  $\mathbf{R}^m$  – чебышевскую норму  $l_\infty$ , т. е. для любой матрицы  $D \in \mathbf{R}^{m \times n}$  положим

$$\|D_i\|_1 = \sum_{j \in N_n} |d_{ij}|, \quad i \in N_m;$$

$$\|D\|_\infty = \max_{i \in N_m} \|D_i\|_1 = \max_{i \in N_m} \sum_{j \in N_n} |d_{ij}|,$$

где  $D_i$  –  $i$ -я строка матрицы  $D$ . Таким образом,  $\|D_i\|_1 \leq \|D\|_\infty$  для любого индекса  $i \in N_m$ . Кроме того, используя очевидные при  $x \in \mathbf{E}^n$  неравенства  $D_i x \geq -\|D_i\|_1$ ,  $i \in N_m$ , для любых портфелей  $x$  и  $x'$  выводим

$$D_i x - D_i x' \geq -2\|D\|_\infty, \quad i, i' \in N_m. \quad (1)$$

Будем исследовать устойчивость парето-оптимального инвестиционного портфеля  $x^0 \in P^m(E, R)$  к возмущениям параметров векторного критерия  $f(x, E, R)$  путем прибавления к паре исходных матриц  $(E, R)$  пары возмущающих матриц множества

$$\Omega(\varepsilon) = \{(E', R') \in \mathbf{R}^{m \times n} \times \mathbf{R}^{m \times n} : \|E'\|_\infty < \varepsilon \ \& \ \|R'\|_\infty < \varepsilon\}, \quad \varepsilon > 0.$$

В этом контексте  $Z^m(E + E', R + R')$  – возмущенная задача с множеством Парето  $P^m(E + E', R + R')$ .

По аналогии с [17–19] радиусом устойчивости инвестиционного портфеля  $x^0 \in P^m(E, R)$  назовем число

$$\rho = \rho^m(x^0, E, R) = \begin{cases} \sup \Xi, & \text{если } \Xi \neq \emptyset; \\ 0, & \text{если } \Xi = \emptyset, \end{cases}$$

где  $\Xi = \{\varepsilon > 0 : \forall (E', R') \in \Omega(\varepsilon) \ (x^0 \in P^m(E + E', R + R'))\}$ . Таким образом, радиус устойчивости задает предельный уровень возмущений исходных данных задачи (элементов матриц  $E$  и  $R$ ), при которых сохраняется парето-оптимальность портфеля  $x^0$ .

Методом от противного нетрудно доказать следующую лемму.

**Лемма.** Пусть портфель  $x^0 \in P^m(E, R)$ ,  $\varepsilon > 0$ . Если при каждой паре возмущающих матриц  $(E', R') \in \Omega(\varepsilon)$  и любом портфеле  $x \neq x^0$  справедливо хотя бы одно из неравенств

$$e(x^0, E + E') > e(x, E + E'); \quad (2)$$

$$r(x^0, R + R') < r(x, R + R'), \quad (3)$$

то  $x^0$  является парето-оптимальным портфелем возмущенной задачи  $Z^m(E + E', R + R')$ , т. е.  $x^0 \in P^m(E + E', R + R')$  при  $(E', R') \in \Omega(\varepsilon)$ .

## 2. Оценки радиуса устойчивости

Для портфеля  $x^0 \in P^m(E, R)$  задачи  $Z^m(E, R)$  введем обозначение

$$\varphi = \min \{\psi(x^0, x) : x \in X \setminus \{x^0\}\},$$

где

$$\psi(x^0, x) = \max \left\{ \max_{i \in N_m} \min_{i' \in N_m} (E_{i'} x^0 - E_{i'} x), \min_{i \in N_m} \max_{i' \in N_m} (R_{i'} x - R_{i'} x^0) \right\}.$$

Поскольку  $x^0 \in P^m(E, R)$  и для каждого портфеля  $x \neq x^0$  выполняется соотношение

$$\psi(x^0, x) = \max \{e(x^0, E) - e(x, E), r(x, E) - r(x^0, E)\} \geq 0, \quad (4)$$

то очевидно неравенство  $\varphi \geq 0$ .

**Теорема 1.** Для радиуса устойчивости  $\rho^m(x^0, E, R)$  любого парето-оптимального портфеля  $x^0$  бикритериальной инвестиционной задачи  $Z^m(E, R)$ ,  $m \geq 1$ , справедливы оценки

$$\varphi/2 \leq \rho^m(x^0, E, R) \leq \varphi.$$

Доказательство. Сначала убедимся в справедливости неравенства  $\rho \geq \varphi/2$ , которое очевидно при  $\varphi = 0$ . Пусть  $\varphi > 0$ . Согласно определению числа  $\varphi$  для любого портфеля  $x \in X \setminus \{x^0\}$  выполняется неравенство  $\psi(x^0, x) \geq \varphi$ . Тогда возможны два случая.

*Случай 1.* Портфель  $x \in X \setminus \{x^0\}$  таков, что ввиду (4) имеет место неравенство

$$e(x^0, E) - e(x, E) \geq \varphi.$$

Тогда, учитывая (1), для любой матрицы  $E' \in \mathbf{R}^{m \times n}$  с нормой  $\|E'\|_\infty < \varphi/2$  выводим

$$\begin{aligned} e(x^0, E + E') - e(x, E + E') &= \min_{i \in N_m} (E_i + E'_i)x^0 - \min_{i \in N_m} (E_i + E'_i)x = \\ &= \max_{i \in N_m} \min_{i' \in N_m} (E_{i'}x^0 - E_{i'}x + E'_{i'}x^0 - E'_{i'}x) \geq e(x^0, E) - e(x, E) - 2\|E'\|_\infty \geq \\ &\geq \varphi - 2\|E'\|_\infty > 0, \end{aligned}$$

т. е. справедливо неравенство (2).

*Случай 2.* Портфель  $x \in X \setminus \{x^0\}$  таков, что ввиду (4) выполняется неравенство

$$r(x, E) - r(x^0, E) \geq \varphi.$$

Отсюда, вновь учитывая (1), для любой матрицы  $R' \in \mathbf{R}^{m \times n}$  с нормой  $\|R'\|_\infty < \varphi/2$  получаем

$$\begin{aligned} r(x, R + R') - r(x^0, R + R') &= \max_{i \in N_m} (R_i + R'_i)x - \max_{i \in N_m} (R_i + R'_i)x^0 = \\ &= \min_{i \in N_m} \max_{i' \in N_m} (R_{i'}x - R_{i'}x^0 + R'_{i'}x - R'_{i'}x^0) \geq r(x, R) - r(x^0, R) - 2\|R'\|_\infty \geq \\ &\geq \varphi - 2\|R'\|_\infty > 0, \end{aligned}$$

т. е. справедливо неравенство (3).

Итак, для любого портфеля  $x \neq x^0$  и любой пары возмущающих матриц  $(E', R') \in \Omega(\varphi/2)$  выполняется одно из неравенств (2) или (3). Поэтому согласно лемме заключаем, что портфель  $x^0 \in P^m(E + E', R + R')$  при любой паре возмущающих матриц  $(E', R') \in \Omega(\varphi/2)$ , т. е.  $\rho \geq \varphi/2$ .

Далее докажем неравенство  $\rho \leq \varphi$ . Согласно определению числа  $\varphi$  существует такой портфель  $x^* \in X \setminus \{x^0\}$ , что справедливо равенство  $\psi(x^0, x^*) = \varphi$ . Отсюда с учетом (4) имеем

$$e(x^0, E) - e(x^*, E) \leq \varphi; \quad (5)$$

$$r(x^*, R) - r(x^0, R) \leq \varphi. \quad (6)$$

Так как  $x^* \neq x^0$ , то найдется такой индекс  $l \in N_n$ , что  $x_l^* \neq x_l^0$ . Полагая  $\varepsilon > \varphi$ , рассмотрим пару возмущающих матриц  $(E^0, R^0)$ ,  $E^0 = [e_{ij}^0] \in \mathbf{R}^{m \times n}$ ,  $R^0 = [r_{ij}^0] \in \mathbf{R}^{m \times n}$ , элементы которых зададим по формулам

$$e_{ij}^0 = \begin{cases} -\delta(x_j^0 - x_j^*), & \text{если } i \in N_m, j = l, \\ 0, & \text{если } i \in N_m, j \in N_n \setminus \{l\}; \end{cases}$$

$$r_{ij}^0 = \begin{cases} -\delta(x_j^* - x_j^0), & \text{если } i \in N_m, j = l, \\ 0, & \text{если } i \in N_m, j \in N_n \setminus \{l\}, \end{cases}$$

где  $\varphi < \delta < \varepsilon$ . Тогда

$$\|E^0\|_\infty = \|E_i^0\|_1 = \|R^0\|_\infty = \|R_i^0\|_1 = \delta, \quad i \in N_m.$$

Кроме того, все строки  $E_i^0$ ,  $i \in N_m$ , матрицы  $E^0$  одинаковые и состоят лишь из компонент  $\delta$  и  $-\delta$ . Обозначив такую строку через  $C'$ , получим

$$C'(x^0 - x^*) = -\delta, \quad \|C'\|_1 = \delta.$$

Используя данные выражения и неравенство (5), находим

$$\begin{aligned} e(x^0, E + E^0) - e(x^*, E + E^0) &= \min_{i \in N_m} (E_i + C')x^0 - \min_{i \in N_m} (E_i + C')x^* = \\ &= \min_{i \in N_m} E_i x^0 - \min_{i \in N_m} E_i x^* + C'(x^0 - x^*) = e(x^0, E) - e(x^*, E) + C'(x^0 - x^*) \leq \\ &\leq \varphi - \delta < 0. \end{aligned}$$

Аналогично и все строки  $R_i^0$ ,  $i \in N_m$ , матрицы  $R^0$  одинаковые. Поэтому, обозначив такую строку через  $C''$ , получим

$$C''(x^* - x^0) = -\delta, \quad \|C''\|_1 = \delta.$$

Отсюда и из (6) вытекает

$$\begin{aligned} r(x^*, R + R^0) - r(x^0, R + R^0) &= \max_{i \in N_m} (R_i + C'')x^* - \max_{i \in N_m} (R_i + C'')x^0 = \\ &= \max_{i \in N_m} R_i x^* - \max_{i \in N_m} R_i x^0 + C''(x^* - x^0) = r(x^*, R) - r(x^0, R) + C''(x^* - x^0) \leq \\ &\leq \varphi - \delta < 0. \end{aligned}$$

Резюмируя, убеждаемся в справедливости бинарного отношения

$$x^0 \underset{E+E^0, R+R^0}{\vdash} x^*.$$

Следовательно, для любого числа  $\varepsilon > \varphi$  существует такая пара возмущающих матриц  $(E^0, R^0) \in \Omega(\varepsilon)$ , что парето-оптимальный портфель  $x^0$  задачи  $Z^m(E, R)$  перестает быть таковым в возмущенной задаче  $Z^m(E + E^0, R + R^0)$ , т. е.  $x^0 \notin P^m(E + E^0, R + R^0)$ . Поэтому  $\rho \leq \varphi$ . ■

**Следствие 1.** Радиус устойчивости  $\rho^m(x^0, E, R) = 0$  тогда и только тогда, когда  $\varphi = 0$ .

### 3. Достижимость оценок

При  $m = 1$  инвестиционная задача  $Z^m(E, R)$  превращается в бикритериальную задачу линейного булева программирования  $Z^1(E, R)$ :

$$-Ex \rightarrow \min_{x \in X};$$

$$Rx \rightarrow \min_{x \in X},$$

где  $E = (e_1, e_2, \dots, e_n) \in \mathbf{R}^n$ ,  $R = (r_1, r_2, \dots, r_n) \in \mathbf{R}^n$ , а верхняя оценка согласно теореме 1 принимает вид

$$\rho^1(x^0, E, R) \leq \varphi = \min_{x \in X \setminus \{x^0\}} \max\{E(x^0 - x), R(x - x^0)\}.$$

Как известно [17], правая часть этого соотношения является выражением для радиуса устойчивости парето-оптимального решения  $x^0$  задачи  $Z^1(E, R)$ . Поэтому справедливо следующее утверждение.

**Теорема 2.**  $\rho^1(x^0, E, R) = \varphi$ .

Это свидетельствует о достижимости верхней оценки радиуса устойчивости, установленной теоремой 1.

Следующая теорема свидетельствует о том, что при  $m \geq 2$  радиус устойчивости портфеля  $x^0 \in P^m(E, R)$  может равняться нижней положительной оценке  $\varphi/2$ .

**Теорема 3.** Существует такой класс бикритериальных булевых задач  $Z^m(E, R)$ ,  $m \geq 2$ , что для радиуса устойчивости инвестиционного портфеля  $x^0 \in P^m(E, R)$  любой задачи из этого класса справедлива формула

$$\rho^m(x^0, E, R) = \varphi/2. \quad (7)$$

*Доказательство.* Покажем, что  $\rho \leq \varphi/2$ . Из определения числа  $\varphi > 0$  следует существование такого портфеля  $x^* \in X \setminus \{x^0\}$ , что  $\psi(x^0, x^*) = \varphi$ . Поэтому (ввиду (4)) справедливы неравенства (5) и (6).

Будем полагать, что существуют два индекса  $p \neq q$  множества  $N_n$ , для которых

$$x_p^0 = x_q^* = 1, \quad x_q^0 = x_p^* = 0. \quad (8)$$

Далее построим такую пару возмущающих матриц  $(E^0, R^0) \in \Omega(\varphi/2)$ , чтобы выполнялись неравенства

$$e(x^0, E + E^0) < e(x^*, E + E^0); \quad (9)$$

$$r(x^0, R + R^0) > r(x^*, R + R^0). \quad (10)$$

*1. Построение матрицы  $E^0$  и доказательство неравенства (9).*

Будем считать, что имеет место неравенство

$$(E_{i_1(x^0)} - E_{i_1(x^*)})x^* > \varphi/2, \quad (11)$$

где

$$i_1(x^0) = \arg \min\{E_i x^0 : i \in N_m\};$$

$$i_1(x^*) = \arg \min\{E_i x^* : i \in N_m\},$$

которое влечет неравенство  $i_1(x^0) \neq i_1(x^*)$ , поскольку  $\varphi > 0$ . Таким образом, необходимое нам неравенство (11) диктует выполнение условия  $m \geq 2$  (см. теорему 3).

Для всякого числа  $\varepsilon > \varphi/2$  элементы возмущающей матрицы  $E^0 = [e_{ij}^0] \in \mathbf{R}^{m \times n}$  зададим по формулам

$$e_{ij}^0 = \begin{cases} -\delta_1, & \text{если } i = i_1(x^0), j = p; \\ \delta_1, & \text{если } i \in N_m \setminus \{i_1(x^0)\}, j = q; \\ 0 & \text{в остальных случаях,} \end{cases}$$

где

$$\varphi/2 < \delta_1 < \min\{\varepsilon, (E_{i_1(x^0)} - E_{i_1(x^*)})x^*\}. \quad (12)$$

Отметим, что последние неравенства корректны благодаря (11).

Учитывая строение матрицы  $E^0$  и равенства (8), выводим

$$E_{i_1(x^0)}^0 x^0 = -\delta_1; \quad (13)$$

$$E_i^0 x^0 = 0, \quad i \in N_m \setminus \{i_1(x^0)\}; \quad (14)$$

$$E_i^0 x^* = \delta_1, \quad i \in N_m \setminus \{i_1(x^0)\}; \quad (15)$$

$$E_{i_1(x^0)}^0 x^* = 0, \quad (16)$$

$$\|E_i^0\|_1 = \|E_i^0\|_\infty = \delta_1 > \varphi/2, \quad i \in N_m, \quad E^0 \in \Omega(\varepsilon).$$

Теперь докажем неравенство (9). Согласно (13) и (14) имеем

$$\begin{aligned} e(x^0, E + E^0) &= \min_{i \in N_m} (E_i + E_i^0)x^0 = \min\{(E_{i_1(x^0)} + E_{i_1(x^0)}^0)x^0, \min_{i \neq i_1(x^0)} (E_i + E_i^0)x^0\} = \\ &= \min\{e(x^0, E) - \delta_1, \min_{i \neq i_1(x^0)} E_i x^0\} = e(x^0, E) - \delta_1. \end{aligned} \quad (17)$$

Покажем, что выполняется равенство

$$e(x^*, E + E^0) = e(x^*, E) + \delta_1. \quad (18)$$

Привлекая (15), получаем

$$\begin{aligned} e(x^*, E + E^0) &= \min\{(E_{i_1(x^*)} + E_{i_1(x^*)}^0)x^*, \min_{i \neq i_1(x^*)} (E_i + E_i^0)x^*\} = \\ &= \min\{e(x^*, E) + \delta_1, \min_{i \neq i_1(x^*)} (E_i + E_i^0)x^*\}. \end{aligned}$$

Поэтому, учитывая очевидные ввиду (15) неравенства

$$e(x^*, E) + \delta_1 \leq (E_i + E_i^0)x^*, \quad i \in N_m \setminus \{i_1(x^0)\},$$

для доказательства (18) остается убедиться, что

$$e(x^*, E) + \delta_1 \leq (E_{i_1(x^0)} + E_{i_1(x^0)}^0)x^*.$$

Для этого, воспользовавшись (12) и (16), выводим

$$e(x^*, E) + \delta_1 - (E_{i_1(x^0)} + E_{i_1(x^0)}^0)x^* = (E_{i_1(x^*)} - E_{i_1(x^0)})x^* + \delta_1 < 0.$$

Наконец, последовательно применяя (17), (18), (5) и (12), получаем необходимое неравенство (9):

$$e(x^0, E + E^0) - e(x^*, E + E^0) = e(x^0, E) - e(x^*, E) - 2\delta_1 \leq \varphi - 2\delta_1 < 0.$$

2. Построение матрицы  $R^0$  и доказательство неравенства (10).

Будем считать, что имеет место неравенство

$$(R_{i_2(x^*)} - R_{i_2(x^0)})x^* > \varphi/2, \quad (19)$$

где

$$i_2(x^0) = \arg \max \{R_i x^0 : i \in N_m\};$$

$$i_2(x^*) = \arg \max \{R_i x^* : i \in N_m\},$$

которое влечет неравенство  $i_2(x^0) \neq i_2(x^*)$ , поскольку  $\varphi > 0$ .

Для всякого числа  $\varepsilon > \varphi/2$  элементы возмущающей матрицы  $R^0$  зададим по правилу

$$r_{ij}^0 = \begin{cases} \delta_2, & \text{если } i = i_2(x^0), j = p; \\ -\delta_2, & \text{если } i = N_m \setminus \{i_2(x^0)\}, j = q; \\ 0 & \text{в остальных случаях,} \end{cases}$$

где

$$\varphi/2 < \delta_2 < \min \left\{ \varepsilon, (R_{i_2(x^*)} - R_{i_2(x^0)})x^* \right\}. \quad (20)$$

Отметим, что последние неравенства корректны благодаря (19).

Принимая во внимание строение матрицы  $R^0$  и равенства (8), выводим

$$R_{i_2(x^0)}^0 x^0 = \delta_2; \quad (21)$$

$$R_i^0 x^0 = 0, \quad i \in N_m \setminus \{i_2(x^0)\}; \quad (22)$$

$$R_i^0 x^* = -\delta_2, \quad i \in N_m \setminus \{i_2(x^0)\}; \quad (23)$$

$$R_{i_2(x^0)}^0 x^* = 0, \quad (24)$$

$$\|R_i^0\|_1 = \|R^0\|_\infty = \delta_2 > \varphi/2, \quad i \in N_m, \quad R^0 \in \Omega(\varepsilon).$$

Далее докажем неравенство (10). Согласно (21) и (22) имеем

$$\begin{aligned} r(x^0, R + R^0) &= \max_{i \in N_m} (R_i + R_i^0)x^0 = \max \{ (R_{i_2(x^0)} + R_{i_2(x^0)}^0)x^0, \max_{i \neq i_2(x^0)} (R_i + R_i^0)x^0 \} = \\ &= \max \{ r(x^0, R) + \delta_2, \max_{i \neq i_2(x^0)} R_i x^0 \} = r(x^0, R) + \delta_2. \end{aligned} \quad (25)$$

Покажем, что выполняется равенство

$$r(x^*, R + R^0) = r(x^*, R) - \delta_2. \quad (26)$$

Привлекая (23), выводим

$$\begin{aligned} r(x^*, R + R^0) &= \max\{(R_{i_2(x^*)} + R_{i_2(x^*)}^0)x^*, \max_{i \neq i_2(x^*)} (R_i + R_i^0)x^*\} = \\ &= \max\{r(x^*, R) - \delta_2, \max_{i \neq i_2(x^*)} (R_i + R_i^0)x^*\}. \end{aligned}$$

Поэтому, учитывая очевидные ввиду (23) неравенства

$$r(x^*, R) - \delta_2 \geq (R_i + R_i^0)x^*, \quad i \in N_m \setminus \{i_2(x^0)\},$$

для доказательства (26) остается убедиться, что

$$r(x^*, R) - \delta_2 \geq (R_{i_2(x^0)} + R_{i_2(x^0)}^0)x^*.$$

Для этого, воспользовавшись (20) и (24), имеем

$$r(x^*, R) - \delta_2 - (R_{i_2(x^0)} + R_{i_2(x^0)}^0)x^* = (R_{i_2(x^*)} - R_{i_2(x^0)})x^* - \delta_2 > 0.$$

Наконец, последовательно применяя (25), (26), (6) и (20), получаем необходимое неравенство (10):

$$r(x^*, R + R^0) - r(x^0, R + R^0) = r(x^*, R) - r(x^0, R) - 2\delta_2 \leq \varphi - 2\delta_2 < 0.$$

В результате из доказанных неравенств (9) и (10) вытекает справедливость бинарного отношения

$$x^0 \underset{E+E^0, R+R^0}{\vdash} x^*.$$

Следовательно, для любого числа  $\varepsilon > \varphi/2$  существует такая пара возмущающих матриц  $(E^0, R^0) \in \Omega(\varepsilon)$ , что инвестиционный портфель  $x^0 \in P^m(E, R)$  задачи  $Z^m(E, R)$  перестает быть парето-оптимальным в возмущенной задаче  $Z^m(E + E^0, R + R^0)$ , т. е.  $x^0 \notin P^m(E + E^0, R + R^0)$ . Поэтому  $\rho^m(x^0, E, R) \leq \varphi/2$ .

Резюмируя, убеждаемся в справедливости равенства (7). ■

Приведем числовой пример, иллюстрирующий теорему 3.

**Пример.** Пусть  $m = 2$ ,  $n = 3$ ,  $X = \{x^0, x^*\}$ ,  $x^0 = (1, 1, 0)^T$ ,  $x^* = (0, 1, 1)^T$ ,

$$E = \begin{pmatrix} 6 & 3 & 5 \\ 8 & 2 & 3 \end{pmatrix}, \quad R = \begin{pmatrix} 2 & 1 & 6 \\ 3 & 2 & 1 \end{pmatrix}.$$

Тогда находим

$$e(x^0, E) = 9, \quad e(x^*, E) = 5,$$

$$r(x^0, R) = 5, \quad r(x^*, R) = 7.$$

Поэтому  $x^0 \in P^2(E, R)$ ,  $\varphi = 4$ . Пусть пара возмущающих матриц  $(E^0, R^0)$  имеет вид

$$E^0 = \begin{pmatrix} -\delta_1 & 0 & 0 \\ 0 & 0 & \delta_1 \end{pmatrix}, \quad R^0 = \begin{pmatrix} 0 & 0 & -\delta_2 \\ \delta_2 & 0 & 0 \end{pmatrix},$$

где  $2 < \delta_1 < 3$ ,  $2 < \delta_2 < 4$ . Теперь легко убедиться, что

$$\begin{aligned}\|E^0\|_\infty &= \delta_1 > 2, \quad \|R^0\|_\infty = \delta_2 > 2; \\ e(x^0, E + E^0) &= 9 - \delta_1 < 5 + \delta_1 = e(x^*, E + E^0); \\ r(x^0, R + R^0) &= 5 + \delta_2 > 7 - \delta_2 = r(x^*, R + R^0).\end{aligned}$$

Это означает, что  $x^0 \not\vdash_{E+E^0, R+R^0} x^*$ , т. е.  $x^0 \notin P^2(E + E^0, R + R^0)$ , поэтому  $\rho^2(x^0, E, R) \leq 2$ .

Следовательно, в силу теоремы 1  $\rho^2(x^0, E, R) = \varphi/2 = 2$ .

Заметим, что задача  $Z^2(E, R)$ , рассмотренная в приведенном выше примере, принадлежит классу задач с условием  $\rho = \varphi/2$ , так как (см. теорему 3) выполняются необходимые неравенства (11) и (19), которые принимают вид

$$\begin{aligned}(E_1 - E_2)x^* &= 3 > 2 = \varphi/2; \\ (R_2 - R_1)x^* &= 4 > 2 = \varphi/2,\end{aligned}$$

поскольку, как легко видеть,  $i_1(x^0) = i_2(x^*) = 1$ ,  $i_1(x^0) = i_2(x^*) = 2$ .

#### 4. Условия устойчивости

Парето-оптимальный инвестиционный портфель  $x^0 \in P^m(E, R)$  назовем устойчивым, если  $\rho^m(x^0, E, R) > 0$ . Кроме того, введем традиционное множество Смейла [20], т. е. множество портфелей задачи  $Z^m(E, R)$ , оптимальных по Смейлу:

$$Sm^m(E, R) = \{x \in X : \nexists x' \in X \setminus \{x\} \quad (x \not\vdash_{E,R} x')\},$$

где

$$x \not\vdash_{E,R} x' \Leftrightarrow e(x, E) \leq e(x', E) \ \& \ r(x, R) \geq e(x', R).$$

Очевидно, что  $Sm^m(E, R) \subseteq P^m(E, R)$  при любой паре матриц  $E \in \mathbf{R}^{m \times n}$  и  $R \in \mathbf{R}^{m \times n}$ . Кроме того, понятно, что множество  $Sm^m(E, R)$  может быть пустым.

**Теорема 4.** Для парето-оптимального инвестиционного портфеля  $x^0$  бикритериальной булевой задачи  $Z^m(E, R)$  эквивалентны следующие утверждения:

- (i)  $x^0 \in Sm^m(E, R)$ ;
- (ii) портфель  $x^0$  устойчив;
- (iii)  $\varphi > 0$ .

Доказательство. (i)  $\Rightarrow$  (ii). Если  $x^0 \in Sm^m(E, R)$ , то легко видеть, что для любого портфеля  $x \in X \setminus \{x^0\}$  выполняется неравенство  $\psi(x^0, x) > 0$ . Поэтому в силу теоремы 1 радиус устойчивости удовлетворяет условиям

$$\rho^m(x^0, E, R) \geq \varphi/2 = \min\{\psi(x^0, x)/2 : x \in X \setminus \{x^0\}\} > 0,$$

т. е. портфель  $x^0$  устойчив.

(ii)  $\Rightarrow$  (iii). Эта импликация согласно следствию 1 очевидна.

(iii)  $\Rightarrow$  (i). Из определения числа  $\varphi$  непосредственно следует, что для любого портфеля  $x \neq x^0$  справедливо неравенство

$$\max\{e(x^0, E) - e(x, E), r(x, R) - r(x^0, R)\} \geq \varphi.$$

Поэтому из неравенства  $\varphi > 0$  вытекает включение  $x^0 \in Sm^m(R)$ . ■

### Заключение

В настоящей работе на базе классической портфельной теории Марковица сформулирована векторная задача выбора парето-оптимального инвестиционного портфеля, состоящая в одновременной максимизации дохода (эффективности портфеля) и минимизации риска упущенной выгоды. Такая постановка приводит к бикритериальной булевой задаче с противоречивыми экономическими критериями. Непредсказуемость состояния рынка и его нестабильность учитываются путем использования максиминного критерия Вальда и минимаксного критерия Сэвиджа, а учет неточности и некорректности исходных данных, характерных для реальных задач инвестиционного анализа, основан на проведении параметрического анализа устойчивости парето-оптимального инвестиционного портфеля к возмущениям исходных данных задачи. Получены нижняя и верхняя оценки (границы) радиуса устойчивости. Построение конкретных классов бикритериальных булевых задач позволило доказать неулучшаемость найденных оценок в рамках используемых параметров. Полученные здесь результаты могут быть использованы для определения границ надежности решений, принимаемых инвестором при формировании инвестиционного портфеля, оптимального по двум критериям, а именно: по максиминному критерию Вальда для доходности инвестиционного портфеля и по минимаксному критерию Сэвиджа для риска упущенной инвестором выгоды.

Работа выполнена при поддержке Белорусского республиканского фонда фундаментальных исследований (проект Ф11К-095).

### Список литературы

1. Емеличев, В.А. Постоптимальный анализ многокритериальной инвестиционной задачи Марковица / В.А. Емеличев, В.В. Коротков // Информатика. – 2011. – № 4 (32). – С. 5–14.
2. Сотсков, Ю.Н. Теория расписаний. Системы с неопределенными числовыми параметрами / Ю.Н. Сотсков, Н.Ю. Сотскова. – Минск : ОИПИ НАН Беларуси, 2004. – 290 с.
3. Сотсков, Ю.Н. Исследование устойчивости оптимальных решений / Ю.Н. Сотсков // Информатика. – 2004. – № 4. – С. 65–75.
4. Markowitz, H.M. Portfolio selection: efficient diversification of investments / H.M. Markowitz. – Oxford : Blackwell Publ., 1991. – 384 p.
5. Виленский, П.Л. Оценка эффективности инвестиционных проектов: теория и практика / П.Л. Виленский, В.Н. Лившиц, С.А. Смоляк. – М. : Дело, 2008. – 1104 с.
6. Бронштейн, Е.М. Сравнительный анализ показателей эффективности инвестиционных проектов / Е.М. Бронштейн, Д.А. Черняк // Экономика и математические методы. – 2005. – Т. 41, № 2. – С. 21–28.
7. Бронштейн, Е.М. Управление портфелем ценных бумаг на основе комплексных квантильных мер риска / Е.М. Бронштейн, М.М. Качкаева, Е.В. Тулупова // Известия РАН. Теория и системы управления. – 2011. – № 1. – С. 178–183.
8. Царев, В.В. Оценка экономической эффективности инвестиций / В.В. Царев. – СПб. : Питер, 2004. – 464 с.
9. Wald, A. Statistical decision functions / A. Wald. – N.Y. : John Wiley, 1950. – 179 p.
10. Вальд, А. Статистические решающие функции / А. Вальд // Позиционные игры / под ред. Н.Н. Воробьева, Н.Н. Врублевской. – М. : Наука, 1967. – С. 300–522.
11. Savage, L.J. The Foundations of Statistics / L.J. Savage. – N.Y. : Dover Publ., 1972. – 384 p.
12. Демьянов, В.Ф. Введение в минимакс / В.Ф. Демьянов, В.Н. Малоземов. – М. : Наука, 1972. – 368 с.

13. Федоров, В.В. Численные методы максимина / В.В. Федоров. – М. : Наука, 1979. – 280 с.
14. Minimax and applications / ed. by D.-Z. Du, P.M. Pardalos. – Dordrecht : Kluwer Acad. Publ., 1995. – 308 p.
15. Фон Нейман, Дж. Теория игр и экономическое поведение / Дж. Фон Нейман, О. Моргенштерн. – М. : Наука, 1970. – 707 с.
16. Петросян, Л.А. Теория игр / Л.А. Петросян, Н.А. Зенкевич, Е.А. Семина. – М. : Высшая школа, 1998. – 304 с.
17. Емеличев, В.А. Общий подход к исследованию устойчивости парето-оптимального решения векторной задачи целочисленного линейного программирования / В.А. Емеличев, К.Г. Кузьмин // Дискретная математика. – 2007. – Т. 19, вып. 3. – С. 79–83.
18. Emelichev, V.A. Stability analysis of the Pareto optimal solutions for some vector boolean optimization problem / V.A. Emelichev, K.G. Kuzmin, Yu.V. Nikulin // Optimization. – 2005. – Vol. 54. – P. 545–561.
19. Емеличев, В.А. Многокритериальная инвестиционная задача в условиях неопределенности и риска / В.А. Емеличев, В.В. Коротков, К.Г. Кузьмин // Известия РАН. Теория и системы управления. – 2011. – № 6. – С. 157–164.
20. Smale, S. Global analysis and economics V: Pareto theory with constraints / S. Smale // J. Mathematical Economics. – 1974. – Vol. 1, № 3. – P. 213–221.

Поступила 31.01.12

*Белорусский государственный университет,  
Минск, пр. Независимости, 4  
e-mail: emelichev@bsu.by,  
wladko@tut.by*

**V.A. Emelichev, V.V. Korotkov**

**STABILITY ANALYSIS OF PARETO-OPTIMAL PROJECT  
PORTFOLIO OF BICRITERIA INVESTMENT PROBLEM  
WITH WALD'S AND SAVAGE'S CRITERIA**

Tight lower and upper bounds for the stability radius of a Pareto optimal portfolio of the two-criteria investment Boolean problem with maximin efficiency criteria and minimax loss profit risk criteria are given.

## ПРАВИЛА ДЛЯ АВТОРОВ

1. Статьи принимаются в редакцию через электронную систему подачи по адресу <http://jinfo.bas-net.by> в формате файлов текстовых редакторов Microsoft Word 97 и Word 2000 для Windows. Основной текст статьи набирается с переносами шрифтом Times New Roman 11 пт, интервал между строками – одинарный, абзацный отступ 1 см, поля по 2,5 см со всех сторон.

2. Объем статьи не должен превышать 12 страниц (включая таблицы, иллюстрации, список литературы), количество иллюстраций – не больше пяти. Допускаются краткие сообщения до трех страниц.

3. Статья должна иметь индекс УДК (универсальная десятичная классификация).

4. Название статьи, фамилии всех авторов и аннотация должны быть переведены на английский язык. Для каждого из авторов приводится развернутое название учреждения с полным почтовым адресом, а также номер телефона и электронный адрес (e-mail) для связи с редакцией.

5. Формулы, иллюстрации, таблицы, встречающиеся в статье, должны быть пронумерованы в соответствии с порядком цитирования в тексте. Ссылки на рисунки и таблицы в тексте обязательны. Необходимо избегать повторения одних и тех же данных в таблицах, графиках и тексте статьи.

Рисунки должны быть представлены в виде файлов формата .cdr, .ai, .wmf, .psd, .jpg, .tif (.tiff) и выполнены с хорошим разрешением в масштабе, позволяющем четко различать надписи и обозначения. Подрисовочные подписи с расшифровкой всех позиций, представленных на рисунке, набираются шрифтом гарнитуры основного текста, размер символов 9 пт. Цветные иллюстрации печатаются только в том случае, когда это необходимо для понимания излагаемого материала.

6. Набор формул выполняется в формульных редакторах Microsoft Equation или Math Type и должен быть единообразным по применению шрифтов и знаков по всей статье.

Прямо (□) набираются: греческие и русские буквы; математические символы (sin, lg, ∞); символы химических элементов (C, Cl, CHCl<sub>3</sub>); цифры (римские и арабские); векторы; индексы (верхние и нижние), являющиеся сокращениями слов.

Курсивом (~) набираются: латинские буквы – переменные, символы физических величин (в том числе и в индексе).

7. Сокращения в тексте статьи (за исключением единиц измерения) могут быть использованы только после упоминания полного термина. Единицы измерения физических величин следует приводить в Международной системе СИ.

8. Литература приводится автором общим списком в конце статьи. Ссылки на литературу в тексте идут по порядку и обозначаются цифрой в квадратных скобках. Ссылаться на неопубликованные работы не допускается. С примерами оформления библиографического описания в списке литературы можно ознакомиться в приложении 2 к *Инструкции по оформлению диссертации, автореферата и публикаций по теме диссертации* на сайте Высшей аттестационной комиссии Республики Беларусь <http://vak.org.by>.

9. Поступившие в редакцию статьи направляются на рецензирование специалистам. Основным критерием целесообразности публикации является новизна и информативность статьи. Если по рекомендациям рецензента статья возвращается автору на доработку, а переработанная рукопись вновь рассматривается редколлегией, датой поступления считается день получения редакцией ее окончательного варианта. Статьи не по профилю журнала возвращаются авторам после заключения редколлегии.

10. Статьи, направляемые на доработку, должны быть возвращены в исправленном виде с ответами на все вопросы.

11. Редакция журнала предоставляет возможность первоочередного опубликования статей, представленных лицами, которые осуществляют послевузовское обучение (аспирантура, докторантура, соискательство) в год завершения обучения.

12. Авторы несут ответственность за направление в редакцию статей, уже опубликованных ранее, или статей, принятых к публикации другими изданиями.

13. Редакция оставляет за собой право на редакционные изменения, не искажающие основное содержание статьи.

***Журнал «Информатика» включен Высшей аттестационной комиссией Республики Беларусь в список научных изданий для опубликования результатов диссертационных исследований.***

## **Индексы**

**00827**

для индивидуальных  
подписчиков

**008272**

для предприятий и  
организаций