Визуальные средства программирования роботов и их использование в школах

**Ю.В.Литвинов**, ст. преп. кафедры системного программирования СПбГУ, yurii.litvinov@gmail.com

В российских школах и вузах активно внедряются робототехнические конструкторы как учебные пособия для преподавания информатики и кибернетики. В статье предложены критерии сравнения визуальных сред программирования роботов, существенные для школьной информатики, приведено описание популярных существующих сред, представлена разрабатываемая на кафедре системного программирования СПбГУ система QReal:Robots.

# Введение

В школьной информатике активно используется понятие «исполнитель», как некая сущность, которая выполняет команды, описанные в программе. В качестве исполнителя в российских школах активно внедряются робототехнические конструкторы, самым популярным из которых на данный момент является конструктор Lego Mindstorms NXT [1]. Он позволяет из блока управления, моторов, сенсоров и соединительных деталей собирать роботов, способных под управлением программы сложным образом взаимодействовать с окружающим миром.

Существует довольно много систем, позволяющих программировать такие роботы, как текстовых, так и визуальных. В текстовых средах, как правило, применяются языки программирования, похожие на язык C, поэтому они слишком сложны для первоначального обучения информатике в младших и средних классах школы. С педагогической точки зрения более интересны визуальные средства программирования роботов, пользоваться которыми могут иногда даже дошкольники, не умеющие ещё читать. Существует несколько популярных средств, шире всего распространена на данный момент среда Robolab [2], иногда в школах используются среды NXT-G и Microsoft Robotics Developer Studio [3]. В этой статье проводится анализ перечисленных сред с точки зрения пригодности их для преподавания информатики и кибернетики в школах, делаются выводы об имеющихся недостатках, определяются направления развития подобных систем, и представлена разрабатываемая на кафедре системного программирования Санкт-Петербургского Государственного Университета среда программирования QReal:Robots.

# Мотивация использования программирования роботов в школах

Идея использовать роботов при начальном обучении информатике родилась неслучайно. Проблема, на которую указывал ещё Ф. Брукс в своей известной книге «Мифический человеко-месяц» [4], заключается в том, что программы нематериальны, их невозможно увидеть. Кроме того, даже представить себе программу не так просто – каждый человек “видит” программу по-разному. Людям, которые программируют впервые, приходится сразу же работать с абстрактными понятиями, и судить о правильности своих программ они могут только по внешним проявлениям их работы – какой ответ программа выведет на экран. При этом может быть совсем не очевидно, как программа работает, и что делать, если выводимый ею ответ неправильный, что нужно делать, чтобы получить правильный ответ. К тому же часто случается так, что программа работает неправильно, но правильный ответ всё-таки выводит. Всё это делает изучение информатики весьма сложным.

И отечественные, и зарубежные методисты давно осознают эту проблему, поэтому традиционно начальное обучение информатике проводится с использованием концепции исполнителя – некоторого, зачастую воображаемого, устройства, способного выполнять простые команды в некотором простом окружении. Один из самых известных исполнителей, применяемых в школах – «черепашка» LOGO [5], разработанная американским программистом, психологом и педагогом Сеймуром Пейпертом в 1967 году. Исполнитель «черепашка» может перемещаться по экрану, оставляя за собой след, которым вычерчиваются различные фигуры. Черепашка подчиняется командам простого интерпретируемого языка, позволяющего описывать её перемещения и повороты. Таким образом, процесс исполнения программы визуализируется движением исполнителя по экрану, и если программа работает неправильно, это будет сразу видно.

В Советском Союзе преподавание информатики как школьного предмета началось во многом благодаря усилиям академика А.П. Ершова и его коллектива, в который входили Г.А. Звенигородский и Н.А. Юнерман. Ими была разработана отечественная учебная система «Робик» [6], основанная в основном на тех же принципах, что и LOGO. Ими же были разработаны методики и программы преподавания информатики в школах, где понятие «исполнитель» занимало ключевую позицию.

Однако исполнитель, перемещающийся по экрану, всё же недостаточно нагляден. Сеймур Пейперт в своих экспериментах использовал механическую черепашку [7] – реальный, осязаемый объект, исполняющий программу, что оказалось гораздо понятнее для школьников, чем черепашка, движущаяся по экрану. Современные технологии позволяют создавать недорогие механические устройства, управляемые загружаемой в них программой, либо непосредственно с компьютера, поэтому идея использования материальных исполнителей в школьной информатике получила второе рождение, из-за чего получил распространение конструктор Lego Mindstorms NXT. Для преподавания информатики с использованием этого конструктора существуют методические пособия (например, [8]).

Робототехнический конструктор довольно сложно программировать: из набора деталей могут быть собраны самые разные конструкции, поэтому программировать приходится в терминах оборотов моторов, подключённых к определённым портам управляющего блока, а не в терминах движений и поворотов. Это, безусловно, делает процесс обучения более творческим, поскольку школьники могут собрать своего собственного исполнителя, но и более сложным с точки зрения написания для этого исполнителя программ. Проблема сложности программирования преодолевается использованием наглядных визуальных языков и удобных графических редакторов для составления программ из блоков, представляющих элементарные команды, такие как «включить мотор», «гудок» и т.д. Таким образом, начинающие работают с графическими языками программирования, а более опытные школьники постепенно переходят на текстовые C-образные языки. В комплекте с конструктором поставляется графическая среда программирования NXT-G, поэтому визуальные языки среди использующих Mindstorms NXT весьма популярны.

# Критерии сравнения средств программирования роботов

Определим критерии, по которым будем сравнивать существующие средства программирования роботов. Поскольку в контексте данной работы нас интересует использование роботов в школьной информатике, основной акцент в обзоре будет сделан на свойства продуктов, которые важны именно в этой сфере.

1. Возможность создавать довольно сложные программы, включающие в себя нетривиальные математические выражения, циклы, ветвления, переменные, параллельные задачи – применение таких средств должно дать возможность иллюстрировать содержательный материал из информатики и кибернетики, например, понятие регуляторов.
2. Простота и удобство в работе. Неудобный пользовательский интерфейс создаёт дополнительную когнитивную нагрузку на школьников и усложняет восприятие и без того сложного материала.
3. Наличие встроенных средств отладки, чтобы школьники могли следить за ходом выполнения своей программы и её состояниями, имели бы инструмент для эффективного поиска ошибок.
4. Возможность перехода от графической формы программы к текстовой, чтобы школьники старших классов, серьёзно занимающиеся программированием, имели возможность смотреть на то, как их программа выглядит на более приближенном к индустриальному программированию текстовом языке, и могли бы, возможно, вносить в программу правки в той же среде, в которой они привыкли работать.
5. Необходима русскоязычная среда разработки, поскольку школьники зачастую ещё не владеют иностранными языками, а необходимость работать со словарём существенно усложняет восприятие материала.
6. Цена – каким бы хорошим ни был продукт, если он стоит дорого, не все школы могут себе его позволить.
7. Желательно, чтобы продукт продолжал развиваться и адаптироваться к новым операционным системам и аппаратному обеспечению.

# Среда NXT-G

Среда NXT-G – единственное средство программирования, которое поставляется в комплекте с конструктором Lego Mindstorms NXT. Среда базируется на системе LabView, среде визуального программирования от компании National Instruments. В LabView в качестве языка программирования используется визуальный язык G (из-за чего NXT-G и получила своё название). Язык G моделирует процесс вычислений, ориентированный на данные, в котором явно задаётся не последовательность выполнения операторов, а связи между блоками по данным. Блок программы может предоставлять некоторые выходные данные, которые могут служить входными данными для другого блока. Блоки начинают исполняться, когда имеют данные на всех входах. Если сразу несколько блоков имеют данные на всех входах, они исполняются параллельно. Такой подход довольно сильно отличается от подхода, принятого в императивном программировании, но тем не менее он широко распространён среди инженеров и учёных. Например, на тех же принципах основана другая известная визуальная среда программирования научных вычислений и моделирования Matlab/Simulink.

Основная проблема этой среды состоит в довольно слабой поддержке математических выражений. Математические формулы здесь, как и вся программа, строятся из блоков. Есть блоки арифметических операций, блоки чтения и записи значения в переменную, блок, считывающий значение константы, блоки, считывающие показания с сенсоров. Таким образом, даже чтобы запрограммировать несложную формулу, требуется изображать блоками дерево разбора выражения, которое эту формулу задаёт. Для иллюстрации серьёзности этой проблемы достаточно сказать, что программа, представляющая пропорционально-дифференциальный регулятор для движения робота вдоль линии или вокруг препятствия, на языке C занимает порядка десятка строк, тогда как на NXT-G не помещается на одном экране и весьма сложна для понимания. Таким образом, первому из предложенных критериев – пригодности для иллюстрации содержательного материала информатики и кибернетики – NXT-G не соответствует. В основном поэтому NXT-G и не получил широкого распространения в школах.

Что касается простоты и удобства в работе, среда NXT-G специально создавалась для начинающих и поэтому довольно эргономична. По мнению некоторых пользователей она даже слишком эргономична, поскольку не даёт произвольно размещать блоки на диаграмме, автоматически (и не всегда удачно) прокладывает соединительные линии между блоками и т.д. Для применения NXT-G в школьных классах оказалась важна ещё такая его особенность: большая часть свойств элементов не отображается на диаграмме, а доступна только через редактор свойств, что делает невозможным показ всей программы на проекторе. Никаких средств отладки NXT-G не имеет, текстовая форма программы не порождается, русификация существует, но неофициальная. К плюсам продукта следует отнести то, что он распространяется вместе с конструктором и доступен для скачивания с сайта производителя бесплатно. Кроме того, продукт до сих пор развивается и обновляется. Средствами LabView возможно добавление сторонних блоков, кроме того, сам NXT-G позволяет выделить набор блоков в подпрограмму и использовать её как новый блок.

# Среда Robolab

Среда Robolab [2] так же, как и NXT-G, базируется на среде LabView. Robolab специально создавалась для школьного образования и c самого начала своего развития учитывала пожелания школьных учителей и специфику преподавания в школах. Пример специфичного для школ решения, реализованного в Robolab – наличие нескольких уровней возможностей среды. На самом простом уровне доступны только некоторые возможности визуального языка, и программа строится заполнением пустых мест в шаблоне посредством выбора блоков из всплывающего меню. Это позволяет создавать только самые простые программы, имеющие стандартную структуру: команды управления моторами, за которыми следует блок, ожидающий наступления какого-либо события. Причём, этот уровень имеет четыре подуровня, и на первых трёх подуровнях программа может состоять только из одного такого "шага". Сделано всё это для того, чтобы дать возможность детям в начальной школе или даже детском саду пользоваться программой – в столь раннем возрасте они вполне могут не уметь читать. На втором уровне (который тоже состоит из нескольких подуровней) пользователи могут рисовать уже настоящие диаграммы, размещая произвольным образом блоки из палитры и соединяя их линиями, определяющими поток управления. Разница между подуровнями заключается в количестве доступных в палитре блоков, первые подуровни имеют меньше блоков с меньшим количеством параметров. Разбиение на уровни и подуровни организовано так, чтобы дети могли осваивать среду программирования практически без помощи учителя, руководствуясь лишь интуицией. Отзывы учителей, приведённые в [2], показывают, что этой цели удалось достигнуть.

Математические выражения Robolab поддерживает гораздо лучше, чем NXT-G, позволяя писать произвольные выражения в текстовом виде. В Robolab имеется возможность использовать в формулах тригонометрические функции, обращаться напрямую к значениям показаний сенсоров. Циклы в Robolab реализованы довольно необычно – есть блок "метка" и блок "переход к метке", передача управления никак больше не визуализируется. Имеются условные операторы, возможность порождать параллельные процессы, блоки для управления этими процессами, а также средства работы с подпрограммами. На Robolab можно просто и довольно удобно реализовать даже довольно сложные программы, и Robolab вполне подходит для иллюстрации материала из кибернетики до младших курсов вузов.

По другим критериям Robolab показывает несколько худшие характеристики. Приложение было создано в конце 90-х годов, с тех пор его интерфейс практически не менялся, поэтому сейчас он выглядит несовременно. Кроме того, он довольно неудобен. Специализированных средств отладки в Robolab нет, хотя есть возможность снимать показания с робота и отображать на экране компьютера данные. Текстовое представление программы Robolab порождать не может. Русификация присутствует, но лишь частично, некоторые элементы управления не переведены. Robolab небесплатен, одна лицензия по стоимости сравнима с робототехническим набором, что для школ довольно дорого. Развитие Robolab идёт в основном путём добавления новых блоков, сама среда давно не изменялась.

Несмотря на указанные недостатки, Robolab на данный момент является основной средой, используемой в школах для разработки программ для роботов. По отзывам учителей, у многих имеется желание от него отказаться и заменить на что-нибудь более современное, однако пока на рынке не существует продуктов, которые могли бы составить ему серьёзную конкуренцию.

# Среда Microsoft Robotics Developer Studio

Среда Microsoft Robotics Developer Studio [3] – разработка компании Microsoft, предназначенная для программирования сложных многопоточных приложений с реактивной моделью поведения, используемых для управления робототехническими системами. Необходимость создания таких приложений есть не только в робототехнике, поэтому Robotics Developer Studio используется и для создания приложений, к робототехнике не относящихся (например, социальная сеть MySpace использует MRDS как составную часть серверного ПО [9]). Программы в Robotics Developer Studio рисуются в виде диаграмм на визуальном языке VPL (Visual Programming Language), представляющем собой по сути визуализатор связей между отдельными параллельно исполняемыми компонентами (или веб-сервисами), из которых состоит программа. Система состоит из следующих крупных частей.

1. Concurrency and Coordination Runtime (CCR) – библиотека для работы с параллельными и асинхронными потоками данных. Библиотека позволяет прозрачно организовывать распределённые и параллельные вычисления, исполняя задачи на разных вычислительных устройствах. Это весьма полезно при программировании роботов, поскольку программы для роботов по природе реактивны и требуют обработки потоков данных одновременно с нескольких сенсоров, причём часть вычислений может быть сделана прямо на роботе, часть – на компьютере вне него.
2. Decentralized Software Services (DSS) – среда времени выполнения, обеспечивающая представление компонентов программы в виде веб-сервисов и упрощающая организацию взаимодействия между ними. Взаимодействие между веб-сервисами ведётся по специальному протоколу Decentralized Software Services Protocol (DSSP). DSS позволяет создавать распределённые приложения, которым не важно, на каком вычислительном устройстве выполняется тот или иной компонент, на одном из компьютеров робота или на компьютере вовне, лишь бы они были связаны единой сетью.
3. Visual Programming Language (VPL) – визуальный язык и редактор для него, используемый для конфигурирования сервисов. Сервисы можно перетащить на диаграмму, связать их входы и выходы, настроить их атрибуты. Получающаяся на таком языке диаграмма сильно напоминает диаграммы LabView, отображая зависимость между компонентами по данным.
4. Visual Simulation Environment (VSE) – трёхмерная среда симуляции поведения робота в виртуальном мире. Обладает довольно богатыми возможностями по симуляции физики и богатыми средствами отображения трёхмерной графики, что позволяет строить сложные и красиво выглядящие модели мира, с которым робот может взаимодействовать. В поставку среды включено несколько моделей окружения, в том числе модель квартиры, в которой работает так называемая "стандартная модель" робота, трёхколёсная платформа с установленным на ней ноутбуком, сенсором Microsoft Kinect, инфракрасными датчиками расстояния и сонаром.

Необходимо отметить, что в сфере школьного образования Microsoft Robotics Developer Studio используется очень редко. Главная причина этого заключается в том, что среда рассчитана в основном на симуляцию и не может эффективно взаимодействовать с реальным роботом. Для LEGO Mindstorms NXT есть возможность управления по каналу Bluetooth, но залить программу на робот возможности нет – на роботе нет возможности запустить .NET-машину. Реальные роботы, управляемые MRDS, обычно гораздо сложнее и дороже того, что можно использовать в школах (стандартная платформа, например, имеет в своём составе ноутбук, который один, скорее всего, дороже всего набора Mindstorms). Управления по Bluetooth недостаточно для решения задач, требующих малого времени реакции робота, из-за больших задержек посылки-приёма Bluetooth-пакетов, что делает MRDS неприменимой для большой области решаемых в школе задач. Симуляции же тоже оказывается недостаточно, потому что даже с хорошим физическим движком MRDS создаёт модель некоторого идеального мира, в котором большого количества проблем, решаемых алгоритмами кибернетики, просто не возникает. Даже простая задача, решаемая на реальном роботе, может оказаться нагляднее и полезнее школьникам, чем сложная программа, исполняемая на модели в симуляторе.

Вторая важная причина очень узкого распространения MRDS в школах – модель вычислений, в ней используемая. Представление программы в виде набора взаимосвязанных распределённых веб-сервисов может быть удобным для опытных программистов, но начинающим тяжело понять принципы, лежащие в основе такой модели. Сложные механизмы взаимодействия веб-сервисов во многом спрятаны с помощью визуального языка VPL, но всё же требуется некоторое понимание происходящих в системе процессов для того, чтобы рисовать содержательные диаграммы. В целом можно сказать, что MRDS больше подходит для студентов или профессиональных программистов, чем для школьников. Среда хоть и позволяет писать сколь угодно сложные программы, но делать это можно довольно нетривиальным и специфическим образом, что сильно снижает её ценность как иллюстративного материала.

Что касается других критериев, система довольно удобна в работе, имеет средства отладки и кодогенерации, однако, в силу своей специфики, эти средства довольно сложны для использования школьниками. Русификация системы отсутствует, однако же, система распространяется свободно и активно развивается.

# Выводы

Сводная таблица с результатами сравнения существующих визуальных средств программирования роботов представлена ниже.

|  |  |  |  |
| --- | --- | --- | --- |
|  | NXT-G | Robolab | MRDS |
| Возможность создания сложных программ | Слабая поддержка сложных математических выражений | Присутствует всё, что требуется школьникам | Сколь угодно сложные программы |
| Простота и удобство в работе | Довольно эргономично, но неудобно для уроков | Удобна, но устаревший интерфейс | Довольно удобна, но сложна |
| Средства отладки | Нет | Нет | Есть отладчик |
| Текстовое представление программы | Нет | Нет | Генерирует код на C# |
| Русификация | Есть, неофициальная | Есть, частичная | Нет |
| Цена | Бесплатна | Порядка 5000 рублей за лицензию | Бесплатна |
| Развитие | Развивается | Сама среда не развивается, только добавляют новые блоки | Развивается |

Табл. . Сравнение средств программирования роботов

Можно сделать вывод, что существующие среды, кроме Robolab, слабо подходят для преподавания информатики и кибернетики в школах. Кроме того, как видно из таблицы, среда Robolab также имеет ряд существенных недостатков, наиболее важный из которых для школ – высокая стоимость. Таким образом, существует потребность в создании бесплатной среды программирования, схожей по функциональности с Robolab, но имеющей средства отладки, генерации кода, русификацию и при этом удобную в работе.

# Среда QReal:Robots

Среда программирования QReal:Robots [10] разрабатывается на кафедре системного программирования Санкт-Петербургского Государственного Университета с 2011 года на базе DSM-платформы QReal [11]. Разработка началась после того, как платформой QReal заинтересовались на кафедре прикладной кибернетики университета, и совместно со школьными учителями были определены требования к среде. На данный момент система находится в работоспособном состоянии, хотя некоторая нестабильность работы мешает её активному внедрению в школьный образовательный процесс.

В QReal:Robots модель вычислений строится вокруг понятия потока исполнения, в отличие от подхода, принятого в рассмотренных системах. Связи между блоками на диаграмме указывают, какой блок будет исполняться следующим, зависимости по данным между блоками не визуализируются. Такой подход оказывается более удобным для восприятия, поскольку ближе к императивным языкам программирования и интуитивному представлению о программе как последовательности команд исполнителю.

Рассмотрим QReal:Robots с точки зрения предложенных критериев. Математические выражения в QReal:Robots могут задаваться в текстовой форме, в любом месте, где требуется численное значение, кроме того, есть отдельный блок "функция", предназначенный для записи математического выражения. Имеется возможность использовать переменные и обращаться к текущим показаниям сенсоров прямо из выражения. В языке имеется поддержка алгоритмических конструкций – ветвлений, циклов, параллельных потоков исполнения. Удобство пользовательского интерфейса QReal:Robots обеспечивается, во-первых, возможностями базовой технологии QReal, во-вторых, исследованиями удобства пользовательского интерфейса конкретно QReal:Robots. Базовая технология обеспечивает, например, распознавание жестов мышью, которое используется в QReal:Robots для быстрого рисования связей между блоками – достаточно с зажатой правой кнопкой мыши провести линию между двумя блоками, чтобы соединить их. Проводилось анкетирование школьников по вопросам удобства использования QReal:Robots, был выявлен ряд замечаний, которые были исправлены.

Наличие развитых средств отладки является основным преимуществом QReal:Robots перед рассмотренными системами. Первое – это возможность интерпретации программы на компьютере с дистанционным управлением роботом по Bluetooth или USB. При этом среда подсвечивает текущий исполняемый блок, визуализируя ход выполнения программы и давая возможность понять, какой блок вызвал ошибку. Второе – наличие двухмерной модели робота, которая может быть использована как исполнитель программы вместо реального робота. Двухмерная модель имитирует трёхколёсную тележку, применяемую, например, в футболе роботов. Модель позволяет задавать расположение и тип сенсоров, и объекты реального мира, такие как стены и линии на полу, с которыми взаимодействуют сенсоры. Двухмерная модель позволяет отлаживать программы без использования настоящего робота, что делает цикл отладки гораздо быстрее, и позволяет использовать QReal:Robots без доступа к роботу вообще, что может быть полезно для школ, пока не закупивших робототехнические конструкторы.

Среда QReal:Robots имеет генератор кода на языке C по диаграммам. Сгенерированный код можно скомпилировать и загрузить на робот для автономного исполнения, при этом его можно просмотреть прямо в среде QReal. Среда изначально разрабатывалась для русскоязычной аудитории, поэтому не требует русификации, бесплатна (и разрабатывается как продукт с открытым исходным кодом), продолжает активно развиваться силами студентов.

# Заключение

Рассмотренные в статье среды программирования роботов обладают рядом недостатков, затрудняющих их применение в российских школах. Наиболее соответствует потребностям школьных учителей на данный момент среда Robolab, но она несколько устарела и небесплатна, поэтому существует потребность в новой среде программирования роботов. Такой средой может стать QReal:Robots.

Среда QReal:Robots была представлена на "Открытых состязаниях Санкт-Петербурга по робототехнике" и на робототехническом фестивале "Робофест 2012" в Москве. В качестве доказательства применимости среды QReal:Robots к реальным задачам, решаемым школьниками, команда студентов приняла участие в соревнованиях в движении робота по линии с программой, реализованной целиком на QReal:Robots. Несмотря на то, что для студентов участие в соревнованиях стало практически первым опытом решения задач робототехники, им удалось показать довольно неплохие результаты, заняв места в середине таблицы, несмотря на то, что довольно многие участники использовали специально созданные для этой задачи роботы. QReal:Robots представлялся также в виде стендовых докладов, где вызвал большую заинтересованность у потенциальных пользователей. Было проведено анкетирование удобства пользовательского интерфейса, которое показало, что продукт достаточно хорош, чтобы вызывать у пользователей симпатию и желание им пользоваться.

Литература

1. LEGO Mindstorms homepage,

URL: http://mindstorms.lego.com/en-us/Default.aspx

2. Portsmore, Merredith, ROBOLAB: Intuitive Robotic Programming Software to Support Life Long Learning, APPLE Learning Technology Review, Spring/Summer 1999

3. Microsoft Robotics Developer Studio homepage,

URL: http://www.microsoft.com/robotics/

4. Брукс Ф., Мифический человеко-месяц, или Как создаются программные системы, Символ-Плюс, 2010г. 304 С.

5. MyRobot, Язык программирования Лого,

URL: http://myrobot.ru/logo/aboutlogo.php

6. Звенигородский Г.А., Описание языка Робик,

URL: http://ershov.iis.nsk.su/archive/eaindex.asp?lang=1\&did=7639

7. cyberneticzoo.com, 1969 – The Logo Turtle – Seymour Papert et al,

URL: http://cyberneticzoo.com/?p=1711

8. Филиппов С.А., Робототехника для детей и родителей, Наука, 2011г. 264 С.

9. Michael S. Scherotter, CCR at MySpace,

URL: http://channel9.msdn.com/Shows/Communicating/CCR-at-MySpace

10. Брыксин Т.А., Литвинов Ю.В., Среда визуального программирования роботов QReal:Robots // Материалы международной конференции «Информационные технологии в образовании и науке». Самара. 2011. С. 332--334.

11. Терехов А.Н., Брыксин Т.А., Литвинов Ю.В., Смирнов К.К., Никандров Г.А., Иванов В.Ю., Такун Е.И., Архитектура среды визуального моделирования QReal. // Системное программирование. Вып. 4. СПб.: Изд-во СПбГУ. 2009, С. 171-196