Федеральное государственное учреждение «Федеральный научный центр
Научно-исследовательский институт системных исследований Российской академии наук»
(ФГУ ФНЦ НИИСИ РАН)

ТРУДЫ НИИСИ РАН

ТОМ 9 № 6

МАТЕМАТИЧЕСКОЕ И КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ СЛОЖНЫХ СИСТЕМ:

ТЕОРЕТИЧЕСКИЕ И ПРИКЛАДНЫЕ АСПЕКТЫ

МОСКВА
2019

Тематика номера:
Обучение алгоритмике и программированию, информационные системы и сети, моделирование динамических природных и рукотворных процессов, вопросы программирования, архитектура вычислительных систем.

Журнал публикует оригинальные статьи по следующим областям исследований: математическое и компьютерное моделирование, обработка изображений, визуализация, системный анализ, методы обработки сигналов, информационная безопасность, информационные технологии, высокопроизводительные вычисления, оптико-нейронные технологии, микро- и наноэлектроника, математические исследования и вопросы численного анализа, история науки и техники.

The topic of the issue:
Algorithmics and programming education, information system and network, dynamics natural and manmade process simulation, software development, computing system architecture.

The Journal publishes novel articles on the following research arias: mathematical and computer modeling, image processing, visualization, system analysis, signal processing, information security, information technologies, high-performance computing, optical-neural technologies, micro- and nanoelectronics, mathematical researches and problems of numerical analysis, history of science and of technique.

Заведующий редакцией: В.Е.Текунов

Издатель: ФГУ ФНЦ НИИСИ РАН,
117218, Москва, Нахимовский проспект 36, к. 1

СОДЕРЖАНИЕ

I. ОБУЧЕНИЕ АЛГОРИТМИКЕ И ПРОГРАММИРОВАНИЮ

II. ИНФОРМАЦИОННЫЕ СИСТЕМЫ И СЕТИ

III. МОДЕЛИРОВАНИЕ ДИНАМИЧЕСКИХ ПРИРОДНЫХ И РУКОТВОРНЫХ ПРОЦЕССОВ

IV. ВОПРОСЫ ПРОГРАММИРОВАНИЯ

V. АРХИТЕКТУРА ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

Академику В.П. Платонову – 80 лет!

1 декабря 2019 года исполнилось 80 лет выдающемуся математику, академику РАН Владимиру Петровичу Платонову!

Владимир Петрович стал работать в НИИСИ РАН в 2005 году, где в 2007 году им был создан новый отдел – Отдел теоретической и прикладной алгебры и теории чисел. За относительно короткое время Владимиру Петровичу совместно с молодыми учениками, удалось получить блестящие и прорывные результаты в новой для него области исследования – им был предложен принципиально новый подход к решению одной из классических проблем алгебры и алгебраической геометрии – проблеме кручения в якобиевых многообразиях гиперэллиптических кривых. Этот подход основан на глубокой связи проблемы кручения и проблемы вычисления фундаментальных S-единиц в гиперэллиптических полях. Благодаря симбиозу глубокой теории и высокопроизводительных вычислений, осуществленных в НИИСИ РАН, эти замечательные результаты имели широкий резонанс в научном мире. Они получили заслуженное признание как в России, так и за рубежом, являются гордостью НИИСИ РАН и на сегодняшний день без сомнения превосходят мировой уровень в данной области исследований.

Владимир Петрович явился инициатором создания в 2011 году научного журнала «Труды НИИСИ РАН», принимал активное участие в работе Редакционного совета и все эти годы оказывал ему поддержку в издании наиболее интересных работ сотрудников института. В настоящее время журнал индексируется в базе данных РИНЦ и выходит 6 раз в год.

Редакционный совет ФГУ ФНЦ НИИСИ РАН сердечно поздравляет Владимира Петровича со славным юбилеем и желает ему крепкого здоровья, счастья, благополучия и новых творческих успехов!

Графические исполнители в системах пиктограммного, блочного и текстового программирования

Д.Б. Аглямутдинова[1], А.В. Готвальд[2], К.А. Мащенко[3], А.Е. Орловский[4], Н. А. Серебрицкая[5]

ФГУ ФНЦ НИИСИ РАН, Москва, Россия

[1]dagliamutdinova@vip.niisi.ru, [2]gotwald@niisi.ru, [3]kirill010399@vip.niisi.ru, [4]orlovskiy@niisi.ru, [5]serebr@vip.niisi.ru

Аннотация. В этой статье рассказывается о целесообразности использования графических исполнителей во вводном курсе программирования и необходимости последовательно использовать в таких курсах пиктограммный, блочный и текстовый стили программирования. Описаны унифицированные комплекты графических исполнителей в отечественных учебных средах КуМир, ПиктоМир и ПиктоМир-К. Рассмотрены примеры решения некоторых алгоритмических задач с исполнителями Робот, Кузнечик, Водолей, Черепашка и Чертежник.

Ключевые слова: ПиктоМир, Кумир, пиктограмма, блочное программирование, графические исполнители, дошкольники, младшеклассники

1 Введение

Возрастающее влияние информационных технологий на современный мир, появление терминов «цифровые технологии», «цифровая экономика», «цифровое образование» и одноименных государственных и отраслевых программ, приводит к возрастанию спроса как на специалистов в области создания и разработки программного обеспечения и цифровых технологий, так и на специалистов, умеющих применять существующие и изобретать новые цифровые технологии. В связи с этим современная система образования, как в России, так и во всем мире, сталкивается с двумя задачами. В средне срочной перспективе необходима интенсификация подготовки специалистов для цифровой экономики в системе высшего образования. В долгосрочной перспективе, необходима перестройка системы школьного образования, с организацией возможно более раннего знакомства детей с цифровыми технологиями, информатикой и программированием. Для интенсификации процесса обучению программированию, его следует проводить в нескольких системах программирования, позволяющих составлять программы управления виртуальными, графическими исполнителями.

2 Необходимость использования графических сред программирования

Фундамент цифровых технологий образуют аппаратное и программное обеспечение. Поэтому освоение цифровых технологий невозможно без освоения азов программирования. Отдел учебной информатики ФГУ ФНЦ НИИСИ РАН в течение многих лет разрабатывает программно-методическое обеспечение вводных и специальных курсов программирования для обучаемых самых разных возрастов и уровней. Отдел учебной информатики учит дошкольников возраста 6+, школьников начальной, основной и старшей школы, младшекурсников и выпускников МГУ и МПГУ, воспитателей детских садов, учителей начальной школы и учителей информатики основной школы [1-7].

Еще в те времена, когда было принято начинать обучение программированию на младших курсах университетов, обнаружилось, что гораздо эффективнее начинать обучение программированию не с алгоритмов каких-либо вычислений, а с алгоритмов управления реальными и виртуальными движущимися объектами. Объекты эти назывались роботами или исполнителями, а программные системы, имитирующие на экране поведение объектов в некоторой обстановке, стали называть, графическими исполнителями или минимирами

[8]. Самым знаменитым объектом такого типа является Черепашка, придуманная Сеймуром Пейпертом в 1966-67 году с целью сделать программирование доступным американским семиклассникам. Историю этого изобретения можно найти в мемориальной статье А.Л. Семенова [9].

В системе образования России (точнее, СССР) подобные графические исполнители появились позже, в конце 70-х годов прошлого века, в Новосибирском университете и в МГУ [10], [11].

При введении в 1985 году в старших классах средней школы СССР обязательного предмета «Основы информатики и вычислительной техники» академик Андрей Петрович Ершов ввел паскале-подобный алгоритмический язык с русской лексикой. Он получил название Школьный алгоритмический язык и первоначально должен был служить псевдокодом для безмашинного курса информатики в школе. На механико-математическом факультете МГУ в сотрудничестве с Академией Наук СССР была в том же году разработаны учебная среда программирования Е-практикум, и годом позже – система КуМир, поддерживающие этот язык и предназначенные для обеспечения компьютерных практикумов по новому предмету. Название КуМир расшифровалось как Комплект Учебных Миров. Действительно, в системе КуМир по единой схеме был реализован ряд графических исполнителей (учебных Миров): Робот, Чертежник, Черепашка, Кузнечик и др. [12]. Система КуМир была ориентирована на школьников старших классов, а позднее, после понижения курса информатики в основную школу, успешно использовалась в 7-9 классах. Сегодня в системе КуМир реализованы практикумы с автоматизированной проверкой заданий, позволяющие школьникам основной школы (5 – 9 класс) освоить азы программирования, составляя от сотни до двух сотен задач по управлению исполнителями в графических средах. Однако это практика стремительно устаревает. Во всем мире усиливается тенденция радикального понижения возраста первого знакомства детей с программированием [13], [14]. Отдел учебной информатики НИИСИ РАН начал разрабатывать методику и ПО для обучения программированию дошкольников в 2010 году.

3 Необходимость использования пиктограммных, блочных и текстовых систем программирования при обучении дошкольников и младшеклассников

Наш опыт показал, что освоение азов программирования возможно начиная с возраста 6+ и требует освоения некоторого инвариантного базового набора понятий, практик и навыков, который от возраста не зависит. Этот набор на 100 процентов может быть освоен в графических средах программирования в процессе составления программ управления движущимися исполнителями-роботами или стационарными исполнителями типа «Кувшин с камнями» (имитация счетчика). По нашему опыту, начинать обучать детей программированию целесообразно, и даже оптимально, в возрасте 6-7. У детей такого возраста устойчивые навыки чтения и письма еще не сформированы, и единственным методом представления программ, доступным шестилеткам, оказывается бестекстовое, пиктограммное представление. Ни сложный текстовый школьный алгоритмический язык, ни система КуМир, с ее достаточно богатым интерфейсом для работы с дошкольниками и младшеклассниками не годятся.

Для работы в пиктограммных средах нужен язык программирования, в котором есть управляющие конструкции ветвления и повторения, подпрограммы без параметров, отсутствуют понятия числовой или символьной переменной, для эпизодического использования счета служит исполнитель «Кувшин», а для реализации механизмов однопоточного или многопоточного управления исполнителями-роботами используются числовые и логические значения, возвращаемые вызовами команд используемых исполнителей [15]. Наш опыт показал, что временные затраты на практикумы по практическому освоению базового набора понятий программирования от возраста обучаемых практически не зависят и составляют около десятка часов. Дошкольники дополнительно нуждаются в приобретении соответствующего лексикона и опыта его использования в коллективных обсуждениях. В НИИСИ РАН около 10 лет назад была реализована бестекстовая среда программирования ПиктоМир. Она разрабатывалась для дошкольников, но оказалось, что и школьники, и студенты, и взрослые легче и быстрее осваивают набор основных понятий в бестекстовой среде программирования ПиктоМир, а не в текстовой системе КуМир. Поэтому мы снабдили ПиктоМир набором графических сред и

успешно используем ПиктоМир во вводных курсах программирования для обучаемых самых разных возрастов.

3.1 Трудности перехода от пиктограммного стиля программирования к текстовому и пути их преодоления

И у младшеклассников и у обучаемых постарше, возникают два серьезных затруднения при переходе от бестекстового программирования к текстовому.

Первое затруднение состоит в том, что замедляется скорость ввода информации при составлении программы. В бестекстовой среде программирования команды исполнителей представляются атомарными объектами – пиктограммами и вставка в программу вызова команды исполнителя сводится к перемещению на сенсорном экране этого атомарного объекта на нужное место в программе. В текстовой среде программирования, размещение в программе вызова команды исполнителя требует текстового ввода имени команды, что занимает больше времени и, главное, может привести к появлению синтаксической ошибки.

Вторым затруднением является сама возможность возникновения синтаксических ошибок. Наличие такой возможности приводит к двум нежелательным последствиям. Во-первых, к необходимости, пусть на интуитивном уровне, обсуждать с обучаемыми сложное и глубокое понятие синтаксической правильности, что требует значительного времени и усилий. Во-вторых, приходится вводить в систему программирования новую и сложную для новичков подсистему диагностики синтаксических ошибок, которую обучаемые должны практически освоить, на что требуется значительное время.

Для обхода второго затруднения необходимо перейти от текстового представления программы к так называемому блочному, и от системы чисто текстового редактирования программы к синтаксически ориентированному редактированию, не позволяющему обучаемому совершить синтаксическую ошибку при вводе программы.

Синтаксически ориентированное блочное программирование успешно реализовано во многих учебных средах программирования 21 века. Самой известной средой такого рода является среда Scratch [16].

3.2 Блочная система программирования ПиктоМир-К – инструмент облегчения перехода от пиктограммного стиля программирования к текстовому

В НИИСИ РАН разработана бета-версия системы пиктограммно-текстового блочного программирования, которую мы назвали ПиктоМир-К (ПиктоМир плюс КуМир). В этой системе создаваемая программа представляется в текстовом виде на подмножестве Школьного Алгоритмического языка. Хотя внешнее представление программы и является текстовым, ввод программы может быть, за небольшими исключениями, сделан пиктограммным, то есть реализуемым перемещением атомарных пиктограмм команд и пиктограмм управляющих конструкций на сенсорном экране.

Тем самым, в НИИСИ РАН оказались реализованы три учебные системы программирования: ПиктоМир, ПиктоМир-К и КуМир, что позволяет по единой методике начать освоение азов программирования в любом возрасте, начиная с возраста 6+. С появлением систем ПиктоМир и ПиктоМир-К акценты смещаются и для обучаемых любого возраста отпадает необходимость систематически работать с графическими исполнителями в КуМире. Основной объем программ по управлению графическими исполнителями отрабатывается в системах ПиктоМир и ПиктоМир-К. И текстовая система программирования КуМир используется только на заключительном этапе, для закрепления навыков работы в текстовой системе программирования на материале привычных графических сред.

Для обеспечения гладкого переключения между системами ПиктоМир, ПиктоМир-К и КуМир, необходимо было решить задачу максимальной унификации учебных миров, доступных в каждой из этих систем.

Результат решения этой задачи описан ниже.

4 Графические исполнители

4.1 Графические исполнители в текстовой системе программирования КуМир

Комплект графических исполнителей (учебных миров) системы КуМир был спроектирован в расчете на поддержку школьного курса информатики в старших классах с целью позволить пользователю составлять алгоритмы управления различными роботами-исполнителями, используя понятия и конструкции последовательных языков программирования. Предполагалось, что в алгоритмах управления массивы не

используются вовсе, а используется лишь одиночные целые и вещественные переменные. Каждый робот-исполнитель КуМира действует в некоторой графически изображаемой обстановке и обладает компактной системой команд. В системе КуМир реализовано 7 графических исполнителей: Робот, Чертежник, Кузнечик, Вертун, Водолей, Черепаха и Рисователь.

Исполнители Робот и Чертежник использовались во многих школьных учебниках информатики, эти исполнители часто фигурируют в задачах ОГЭ и ЕГЭ по информатике. С использованием исполнителя Робот в системе КуМир авторитетными педагогами Д. Кириенко и К. Поляковым подготовлены практикумы по программированию для учеников 5-7 классов [17], [18].

Все перечисленные исполнители КуМира, за исключением Рисователя, в настоящий момент перенесены в ПиктоМир или в ПиктоМир-К. Функциональность этих исполнителей сохранена, но графическое оформление изменилось, его проработке было уделено больше внимания (см. ниже раздел Сравнение реализаций исполнителя Водолей в системах КуМир и ПиктоМир).

Поскольку богатый выбор учебных миров с разнообразными исполнителями помогает сделать процесс обучения дошкольников и младшеклассников более увлекательным и наглядным, в системах ПиктоМир или в ПиктоМир-К реализован ряд новых, по сравнению с КуМиром, исполнителей. Наконец, с целью повышения мотивации, один из графических исполнителей реализован в материальном мире в виде дистанционно управляемого робота-игрушки.

4.2 Графические исполнители в пиктограммной системе программирования ПиктоМир

В системе ПиктоМир можно реализовать только те исполнители, команды которых не имеют параметров. В ПиктоМире реализованы 4 исполнителя из КуМира: Робот, Кузнечик, Вертун и Водолей и еще 5 исполнителей, придуманных специально для дошкольников: двигающиеся по плоскости Вертун, Тягун, Двигун, Зажигун и неподвижный Кувшин. Поскольку команды исполнителей и управляющие конструкции ПиктоМира представляются наглядными пиктограммами, составлять программы управления исполнителями в ПиктоМире может ребенок шестилетнего возраста, еще не умеющий или не очень любящий читать и писать. Наглядная форма представления программы и планшетный интерфейс позволяют даже самым маленьким пользователям составлять достаточно сложные программы, используя «детский» пиктограммный язык программирования.

5 Описание Исполнителей, не имеющих параметров

5.1 Исполнитель Водолей

Исполнитель работает с тремя разными сосудами, которые обозначаются латинским буквами «A», «B» и «C» (см. рис. 1). Каждый из них имеет свою вместимость и начальный запас воды. Оба этих параметра указываются учителем при подготовке условия задачи (при подготовке уровня Игры в системе ПиктоМир). На рисунке 1.а сосуды имеют вместимости 6, 10 и 16 литров, а начальные запасы воды равны 2, 0 и 5 литров соответственно.

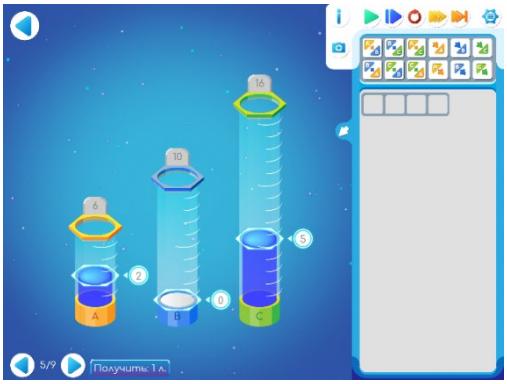


Рис. 1.а. Шаблон программы

Рис. 1.б. Решение задачи.

Аналогичная задача в КуМире выглядит, с точки зрения дошкольника или первоклассника, менее наглядно (см. рис. 2):

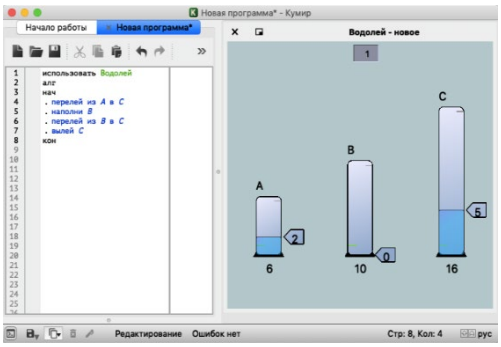


Рис. 2. Исполнитель Водолей в системе КуМир

В ПиктоМире над каждым сосудом написан его максимальный объем, сбоку от сосуда указан текущий объем воды в нём, на сосуды нанесены шкалы в виде белых полосок, каждая пятая из которых длиннее обычных (см. рис. 1.а).

Учитель создает задание, в котором указывает какое количество литров воды нужно получить. Задача ученика состоит в получении в одном из сосудов указанного числа литров воды и опустошении остальных сосудов путем последовательного выполнения команд Водолея.

Водолей выполняет следующие команды (см. таб. 1).

Таблица 1. Команды исполнителя Водолей.

опустошить любой из сосудов			
заполнить любой из сосудов до максимального объема			
перелить воду из любого сосуда в любой другой сосуд			
			

При выполнении команд Водолея состояния сосудов в КуМире меняются «скачком», а в ПиктоМире процесс выполнения команд «оживлен» мультипликацией, уровни воды в сосудах меняются плавно.

Число команд в разных правильных решениях может оказаться разным. В разобранном выше примере показано решение из четырех команд. В КуМире решение записывается в текстовом виде (рис. 2), а в ПиктоМире – в пиктограммном (рис. 1.б). Максимальное число команд в КуМире не ограничено, а в ПиктоМире ограничено размером шаблона программы.

5.2 Исполнитель Робот

Исполнитель Робот действует на прямоугольном поле из квадратных клеток. По границам клеток могут размещаться стены. У исполнителя Робот есть 5 команд-приказов (см. таб.2) и 10 команд-вопросов (см. таб.3).

Таблица 2. Команды-приказы Робота.

вверх	
вниз	
влево	
вправо	
закрасить	

Таблица 3. Команды-вопросы Робота.

клетка закрашена; клетка не закрашена;			
сверху свободно;			
слева свободно; справа свободно;			
снизу свободно;			
сверху стена;			
слева стена; справа стена;			
снизу стена.			

Клетки бывают трех типов: обычные (зеленые), закрашенные (синие), и подлежащие закрашиванию (серые). У Робота есть начальная точка, в которой он находится при старте. Если на поле задана «Финишная точка» (кружочек), то по выполнении программы робот должен оказаться в этой точке (см. рис. 3).

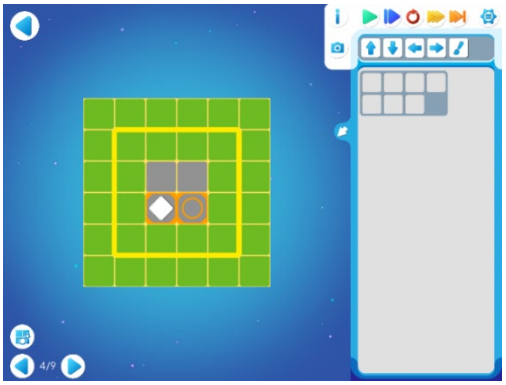


Рис. 3.а. Шаблон программы

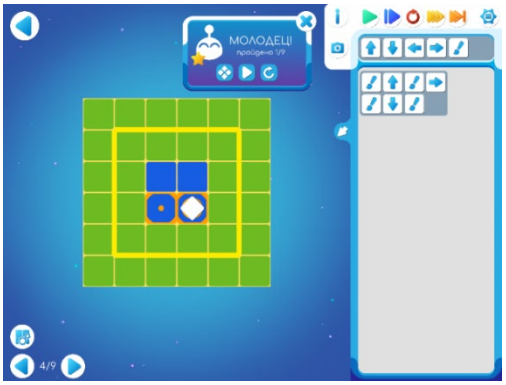


Рис. 3.б. Решение задачи

Типичная задача для исполнителя Робот состоит в том, чтобы закрасить на поле некоторые клетки и переместиться в финишную клетку. Программа, решающая задачу, должна уместиться в заданном шаблоне (см. рис. 3.а). На рисунке 3.б приведено решение задачи. Аналогичное решение в Кумире выглядит следующим образом (см. рис. 4).

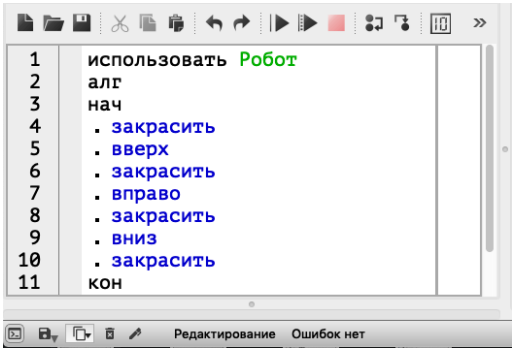


Рис. 4. Программа исполнителя Робот в системе Кумир.

В системе КуМир каждая клетка поля Робота имеет две не изображаемые графически числовые характеристики. В КуМире эти характеристики доступны с помощью вызовов текстовых команд-вопросов: температура и радиация, возвращающих вещественное значение. Поскольку вещественных значений в ПиктоМире нет, эти команды в ПиктоМире не реализованы. В ПиктоМир-К есть целые значения. Но нет вещественных значения, поэтому эти две команды реализованы, как команды-вопросы, возвращающие целое значение. Для этих команд вопросов выбраны пиктограммы, изображенные в таблице 5.

Таблица 5. Команды вопросы, возвращающие целые значения.

Температура	
Радиация	

5.3 Исполнитель Кузнечик

Кузнечик — исполнитель, перемещающийся по клеткам прямоугольной таблицы высотой в одну клетку и конечной шириной. В КуМире поле Кузнечика изображается как одномерный объект. Это изображение ориентировано на детей старшего возраста, уже освоивших понятие «числовая ось» и выглядит следующим образом (см. рис. 5):

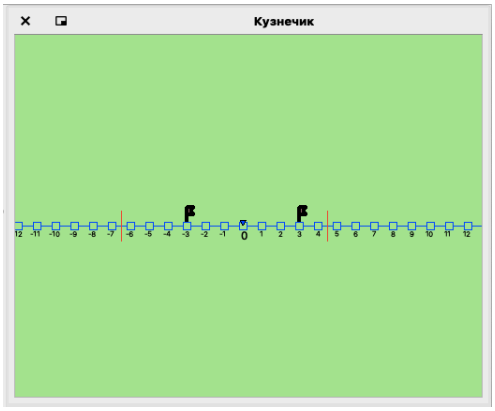


Рис. 5. Обстановка исполнителя Кузнечик

Такое представление для детей младшего возраста не годится, поэтому в ПиктоМире поле Кузнечика изображается (по выбору ребенка) либо как двумерный план, либо как трехмерная картинка (см. рис. 6.а и 6.б).

Рис. 6.а. Двумерное представление обстановки исполнителя Кузнечик.

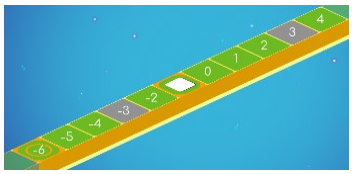


Рис. 6.б. Трехмерное представление обстановки исполнителя Кузнечик

У исполнителя Кузнечик есть 3 команды-приказа (см. таб. 6).

Таблица 6. Команды Кузнечика.

влево	
вправо	
закрасить	

Небольшая особенность Кузнечика состоит в том, что правила выполнения команд **влево** и **вправо** задаются в условии задачи. Например, в условии задачи может быть задано, что **влево** означает прыжок на 2 клетки влево, а **вправо** означает прыжок на 3 клетки вправо.

Клетки на поле Кузнечика бывают трех типов: обычные (зеленые), закрашенные (синие) и подлежащие закрашиванию (серые).

Клетки на поле Кузнечика пронумерованы. На каждой клетке указан ее номер, положительный, отрицательный или нулевой.

На поле указана начальная точка, в которой Кузнечик находится при старте. Если на поле задана «Финишная точка» (клетка с концентрическими окружностями), то по выполнении программы Кузнечик должен оказаться в этой точке (см. рис. 7.а).

Задание для Кузнечика состоит в том, чтобы закрасить на поле некоторые клетки и переместиться в финишную клетку.

Программа, решающая такую задачу, должна уместиться в заданном шаблоне (см. рис. 7.а и рис. 7.б).

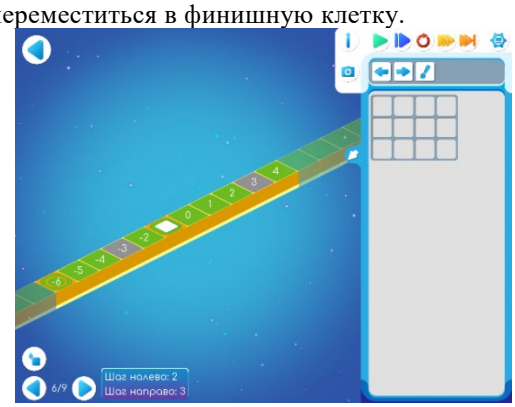


Рис. 7.а. Шаблон программы для задания с исполнителем Кузнечик

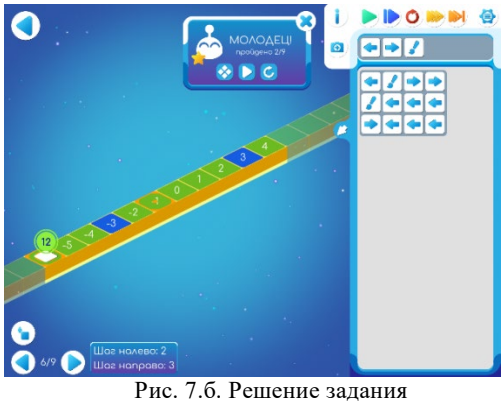


Рис. 7.б. Решение задания с исполнителем Кузнечик

6 Режим непосредственного выполнения команд в системах ПиктоМир и КуМир

И в КуМире и в ПиктоМире задачи со всеми исполнителями можно решать в режиме непосредственного выполнения команд.

В КуМире при составлении программ для различных исполнителей для непосредственного выполнения можно использовать пульт с командами исполнителя. Для каждого исполнителя в КуМире предусмотрен свой пульт (рис. 8.а и рис. 8.б).

Рис. 8.а. Пульт исполнителя Робот

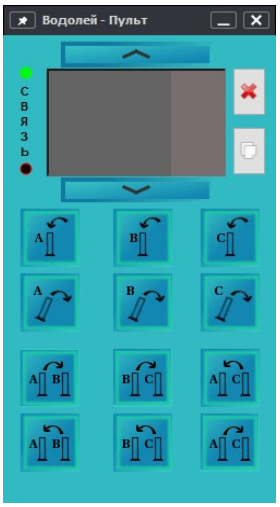


Рис. 8.б. Пульт исполнителя Водолей

Последовательность текстовых представлений команд, отданных с помощью пульта, может быть «запомнена» в текстовом буфере и затем «вспомнена» в нужном месте составляемой программы.

Рис. 9.а. Последовательность команд, отданных с помощью пульта, запоминается в текстовом буфере

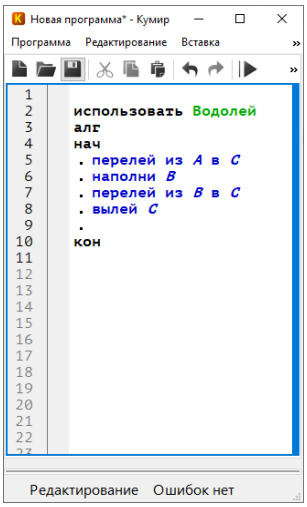


Рис. 9.б. Содержимое текстового буфера «вспоминается» в нужном месте КуМир-программы

Таким образом, в КуМире для реализации режима непосредственного управления используется универсальный механизм запоминания информации в текстовом буфере. Этот механизм работает не только в КуМире, но и в других приложениях, запущенных в используемой операционной системе.

Недостатком режима непосредственного исполнения команд в КуМире является то, что ошибочно отданная с пульта команда не может быть отменена. Этот недостаток исправлен в ПиктоМире. Для режима непосредственного выполнения команд в ПиктоМире реализован механизм «Копилка» (стек пиктограмм команд) (рис. 10). Пиктограммы отданных исполнителю команд добавляются в стек «Копилки». Накопленная в «Копилке» последовательность пиктограмм команд может быть вставлена в нужное место ПиктоМир-программы.

Рис. 10. Использование «Копилки» в системе Пиктомир

Если кликнуть на последней помещенной в «Копилку» пиктограмме (верхний элемент стека Копилки), то последняя отданная исполнителю команда будет отменена: исполнитель при этом будет возвращен в предыдущее состояние, а пиктограмма исчезнет из «Копилки».

Режим «Копилки» универсален, он применим к любому исполнителю в ПиктоМире. Более того, этот режим работает и в ПиктоМире и в ПиктоМир-К с той только разницей, что при добавлении в программу накопленных в «Копилке» команд в ПиктоМире в программу вставляются пиктограммы команд, а в ПиктоМир-К в программу вставляются текстовые представления этих команд.

Наличие Копилки, позволяющее составлять линейные участки программы методом проб и ошибок, является громадным преимуществом систем ПиктоМир и ПиктоМир-К над КуМиром. Использование этого режима позволяет радикально увеличить количество задач, решаемых ребенком в процессе короткой, 15-20-минутной компьютерной сессии каждого занятия курса «Азы программирования» [19].

7 Учебная среда ПиктоМир-К

7.1 Переход от пиктограммного программирования к текстовому с помощью системы ПиктоМир-К

Освоение детского «пиктограммного» языка является лишь промежуточным этапом изучения программирования. В конце вводного курса программирования обучаемые должны уметь работать в текстовых средах программирования. Переход из пиктограммной системы программирования в текстовую, даже упрощенную учебную по мнению многих экспертов [20, 21], оказывается слишком трудным. Для облегчения этого перехода разработана пиктограммно-текстовая среда блочного программирования ПиктоМир-К [22]. К работе в системе Пиктомир-К приступают дети уже набравшие опыт самостоятельного составления программ управления графическими исполнителями в системе ПиктоМир. Поэтому на первом этапе освоения текстового программирования в ПиктоМир-К обучаемым можно давать задачи управления графическими исполнителями, уже отработанные в среде ПиктоМир. Тем самым на этом первом этапе снимаются и трудность освоения новых исполнителей, и трудность придумывания новых алгоритмов управления этими исполнителями и трудность перехода к новой форме представления знакомых алгоритмов управления знакомыми исполнителями. В системе ПиктоМир-К доступны 9 исполнителей ПиктоМира, команды которых не имеют параметров. А именно, Вертун, Двигун, Тягун, Зажигун, Ползун, Робот, Водолей, Кузнечик и Кувшин. Кроме того, доступны два исполнителя системы КуМир, в командах которых задаются параметры: Черепаха и Чертежник. Особым случаем оказывается исполнитель «Кувшин». Без него в системе ПиктоМир-К можно было бы обойтись, поскольку в этой системе можно создавать и использовать целочисленные переменные. Тем не менее, по методическим соображениям, «Кувшин» также реализован в ПиктоМире-К. Тем самым, в ПиктоМир-К реализованы 11 графических исполнителей.

7.2 Подмножество школьного алгоритмического языка, реализованное в блочной системе программирования ПиктоМир-К

Система программирования ПиктоМир-К позволяет составлять на подмножестве Школьного Алгоритмического Языка программы управления одиннадцатью графическими исполнителями. Для краткости назовем это подмножество языком ПиктоК. Автор задания сначала указывает в редакторе шаблонов программ, какими именно исполнителями программа будет управлять. После этого появляется шаблон программы и головное меню редактирования программы, в котором представлены доступные для составления программы блоки, из которых автор может выбрать только те пиктограммы, которые необходимы для конкретной программы. Например, если программа предназначена для управления исполнителем Вертун, то ее шаблон и головное меню выбора блоков, доступных для составления программы, будут выглядеть так (см. рис.11 и рис.12.) По умолчанию на каждом уровне доступны пиктограммы команд-действий, поэтому их нет в головном меню.

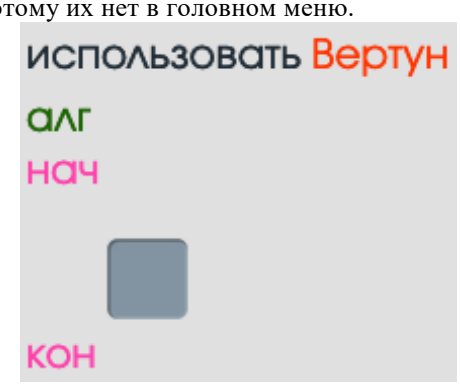


Рис. 11. Шаблон для исполнителя Вертун в системе ПиктоМир-К.

Рис. 12. Головное меню выбора блоков для задания.

В первом ряду на рисунке 12 представлены имена подпрограмм без параметров с предопределенными именами. Далее следуют пиктограммы команд-вопросов робота Вертуна. В следующем ряду показаны инструменты для сравнения числовых значений.

Затем показаны инструменты для создания управляющих конструкций и инструменты создания целочисленных и логических переменных. Завершают головное меню инструменты для создания арифметических выражений. В эти инструменты, входят используемые в Пиктомире пиктограммы числовых значений от 1 до 6. Ввод числовых значений возможен также с помощью экранной клавиатуры, изображенной на рисунке 13.

Рис. 13. Экранная клавиатура для ввода целочисленных значений.

Учитель при создании шаблона задания выбирает, какие пиктограммы будут доступны ученику. Поле для выбора учителем разрешенных компонент языка разделено на вкладки по группам: команды робота, блочные конструкции, инструментарий для использования переменных (см. рис 14).

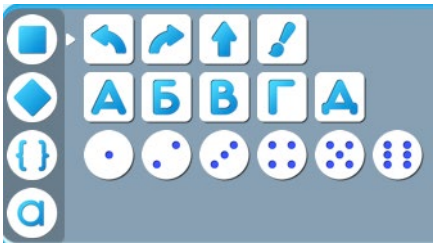


Рис.14. Поле выбора пиктограмм для ученика.

В редакторе языка ПиктоК поддерживаются уже освоенные обучаемыми в ПиктоМире соглашения: прямоугольные пиктограммы задают действия, круглые пиктограммы задают целые значения, а ромбические пиктограммы задают логические значения. Форма пиктограмм подсказывает, в каком именно месте программы возможно их использование.

В процессе создания программы пользователь перетаскивает пиктограммы на места, соответствующие их форме. К примеру, условия можно вставлять только в часть программы обозначенную ромбом. В редакторе также возможно удаление и перемещение, как единого целого, управляющей конструкции внутри программы. Созданная в редакторе системы Пиктомир-К программа соответствует синтаксису языка Кумир и отформатирована в соответствии с ее синтаксической структурой (см. рис. 15).

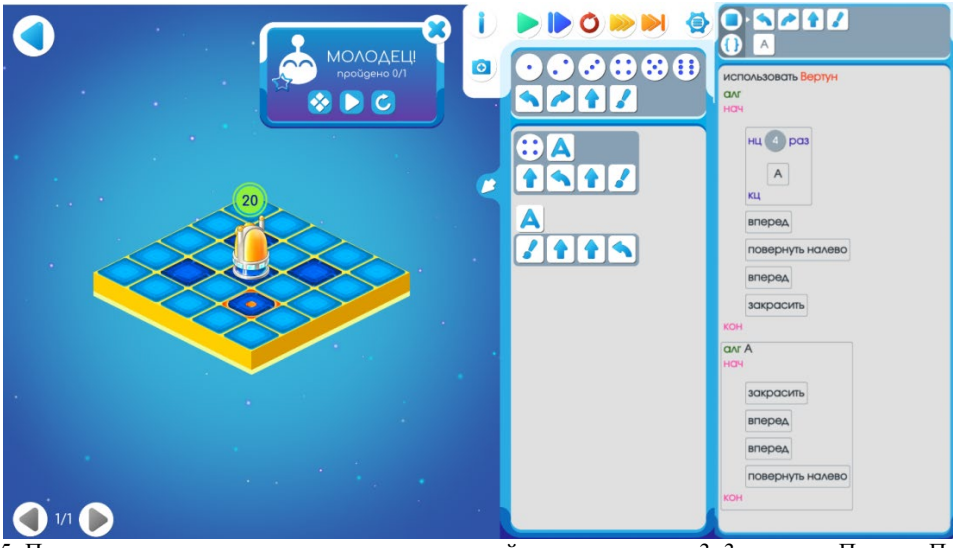
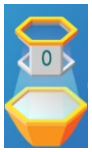
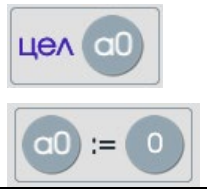
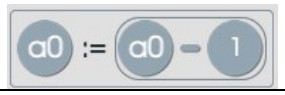
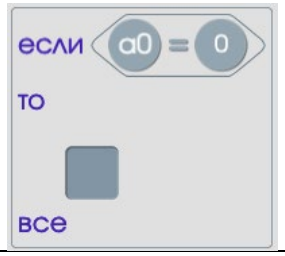
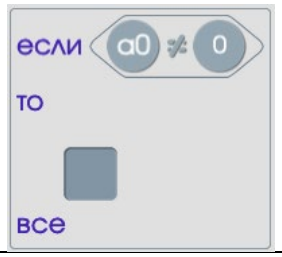
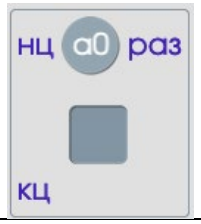


Рис. 15. Программа закрашивания углов и центральной клетки квадрата 3х3 на языке Пикто и Пикто-К

8 Исполнитель Кувшин. Реализация счетчика в системах ПиктоМир и ПиктоМир-К

Для использования элементов счета при составлении алгоритмов управления роботами, в системе ПиктоМир используется стационарный графический исполнитель «Кувшин». Кувшин имитирует поведение целочисленной переменной. Количество камней в Кувшине соответствует значению переменной, операция высыпания всех камней из Кувшина соответствует обнулению переменной, добавление камня в Кувшин соответствует увеличению значения переменной на единицу, а удаление одного камня из непустого Кувшина соответствует уменьшению значения переменной на единицу. Проверка пустоты Кувшина соответствует сравнению значения переменной с нулем. Кувшин реализован и в системе ПиктоМир-К. Таким образом, если в программе, составляемой в системе Пиктомир-К, необходима ровно одна целочисленная переменная-счетчик, то в этой программе ее можно заменить исполнителем Кувшин по следующим правилам (таблица 9).

Таблица 9. Использование счетчика в системах ПиктоМир и ПиктоМир-К.

Пиктомир	Пиктомир-К
	
	
	
	
	
	
	

9 Реализация в системе ПиктоМир-К исполнителей, команды которых имеют аргументы

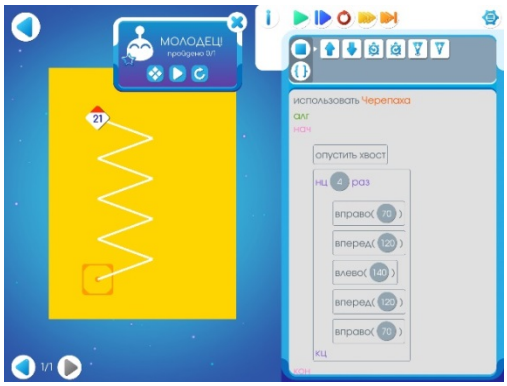
Некоторые команды исполнителей КуМира Черепашка и Чертежник имеют числовые аргументы, поэтому реализация этих исполнителей в системе Пиктомир невозможна.

Реализация же этих исполнителей в системе Пиктомир-К не вызывает затруднений.

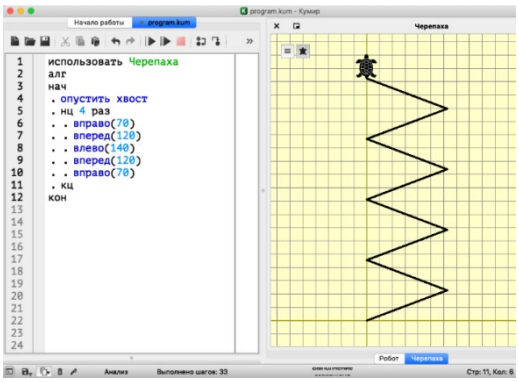
Пиктограмма команды, имеющей числовой аргумент, перетаскивается в программу на языке ПиктоК. При преобразовании пиктограммы команды в текстовую форму в ней возникает поле ввода числа с уже введенным дефолтным значением 0. Тем самым эта команда оказывается корректной синтаксически и выполнимой. Далее, в процессе редактирования, это дефолтное нулевое

значение может быть заменено другим числовым значением, введенным с помощью меню цифрового ввода или заменено арифметическим выражением, синтезируемым по правилам системы ПиктоМир-К. На любом этапе редактирования и редактируемая команда и программа в целом остаются синтаксически корректными.

На рисунках 16.а и 16.б показан пример одной и той же программы управления Черепашкой в средах ПиктомМир-К и Кумир.



16.а Исполнитель Черепашка в системе в системе Пиктомир-К



16.б Исполнитель Черепашка в системе в системе КуМир

На рисунках 17.а и 17.б показан пример одной и той же программы управления Чертежником в средах ПиктоМир-К и Кумир.

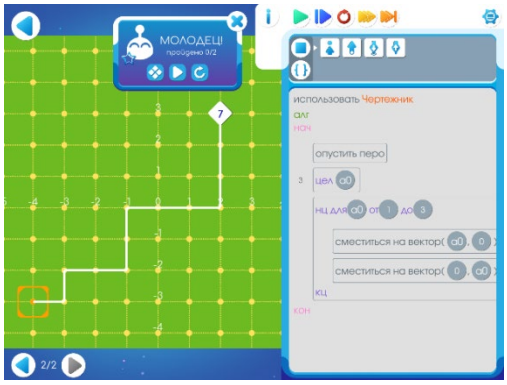


Рис. 17.а Исполнитель Чертежник в системе в системе Пиктомир-К

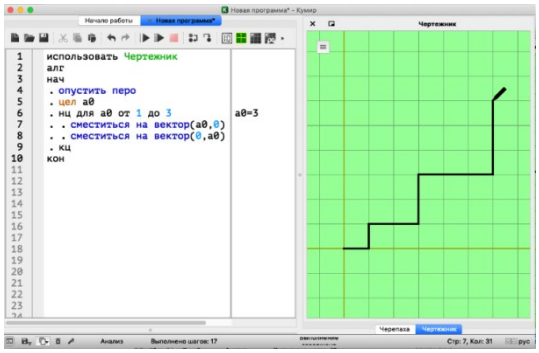


Рис. 17.б Исполнитель Чертежник в системе в системе КуМир

10 Заключение

Обучение программированию становится возможным не только для старшеклассников и взрослых, владеющих навыками чтения и клавиатурного ввода текста, а также для дошкольников и младшеклассников в доступной форме благодаря учебным средам разработки КуМир, ПиктоМир и ПиктоМир-К. Графические исполнители, реализованные в этих программных средах, помогают изучить основные конструкции процедурных языков программирования алгоритмического языка, освоить простейшие практики отладки.

Публикация выполнена в рамках государственного задания по проведению фундаментальных исследований по теме «Разработка, реализация и внедрение семейства интегрированных многоязыковых сред программирования с автоматизированной проверкой заданий для учащихся образовательных организаций, ДОО, младшей, основной и старшей школы и студентов педагогических университетов» (№ 0065-2019-0010).

Visual environments in systems of pictogram, block and text programming

D.B. Agliamutdinova, A.V. Gotvald, K.A. Mashchenko, A.E. Orlovskii, N.A. Serebritskaia

Abstract. This article discusses the advisability of using graphical environments in an introductory programming course and the need to consistently use pictogram, block, and text programming styles in such courses. Unified sets of graphic artists are described in the domestic educational environments Kumir, PiktoMir and PiktoMir-K. Examples of solving some algorithmic problems with performers Robot, Grasshopper, Aquarius, Turtle and Draftsman are considered.

Keywords: PiktoMir, Kumir, block-based programming, visual environments, preschoolers, middle-schoolers, junior high school students

Литература

1. I.B. Rogozhkina, A.G. Kushnirenko. PiktoMir: teaching programming concepts to preschoolers with a new tutorial environment, «Procedia - Social and Behavioral Sciences», (2011). № 28, 2011, 601-605.

2. А.Г. Кушниренко, А.Г. Леонов, И.Б. Рогожкина. Пиктомир: пропедевтика алгоритмического языка (опыт обучения программированию старших дошкольников) «ИТО-РОИ, Большой московский семинар по методике раннего обучения информатике. СЕЗОН 2012/2013», Россия, 25 сентября 2012.

3. А.Г. Кушниренко, А.Г. Леонов, М.А. Ройтберг, Алгоритмика для дошкольников и младших школьников «Одиннадцатая конференция "Свободное программное обеспечение в высшей школе". — Альт Линукс», г. Москва, 2016, 66–71.

4. А.Г. Кушниренко, А.Г Леонов, Я.Н. Зайдельман, В.В Тарасова. Информатика. 7 – 9 классы. М., Дрофа, 2017.

5. Н.О. Бесшапошников, А.Г. Кушниренко, К.А. Прокин, А.Г. Леонов. Технологические инновации меняют методику курса “Алгоритмика для дошкольников”. «Воспитание и обучение детей младшего возраста: VIII Международная конференция (ECCE 2019) (Москва, МГИМО МИД России 29 мая — 1 июня 2019 г.)», Москва, 2019, М., издательство Московского университета, 128–129.

6. Н.О. Бесшапошников, А.Г. Леонов, А.Г. Кушниренко. Пиктомир: как и зачем мы учим бестекстовому программированию дошкольников, школьников и студентов педуниверситетов (Устный) «13-ая Международная конференция «Разработка ПО / SECR2017», г. Санкт-Петербург, Россия, 20-21 октября 2017.

7. А.Л. Семенов. Концептуальные проблемы информатики, алгоритмики и программирования в школе. «Вестник кибернетики», Т. 22(2016), №2, 11–15.

8. S. Papert. Mindstorms: Children, Computers and Powerful Ideas. New York, NY, USA: Basic Books Inc. Publishers, 1980.

9. А. Л. Семенов. Цели общего образования в цифровом мире. «Материалы III Международной конференции «Информатизация образования и методика электронного обучения»», Красноярск, СФУ, 24 сентября 2019 г. – 27 сентября 2019 г., В 2 ч. Ч. 2 , 383 – 388.

10. А.П. Ершов, А. Г. Кушниренко, Г. В. Лебедев, А. Л. Семенов, А. Х. Шень. Основы информатики и вычислительной техники: пробный учебник для средних учебных заведений. Под ред. А. П. Ершова. М., Просвещение, 1988.

11. Г. В. Лебедев, А. Г. Кушниренко. Программирование для математиков: Учебное пособие для вузов по специальностям Математика и Прикладная математика. М., Наука, 1988.

12. А.А. Дуванов, Я.Н. Зайдельман, Ю.А. Первин. Роботландия. «Информатика и образование», №1, 1988.

13. Richtel, Matt. 2014. Reading, Writing, Arithmetic, and Lately, Coding. New York Times. Available at: https://www.nytimes.com/2014/05/11/us/reading-writing-arithmetic-and-lately-coding.html (accessed

3.11.2019).

14. Глава профильного комитета Думы считает нужным ввести информатику в дошкольную программу, Материалы XVIII СЪЕЗДА "ЕДИНОЙ РОССИИ" 08.12.2018, https://tass.ru/obschestvo/5888487, (проверено 02.12.2019).

15. А.Г. Леонов, Н.О. Бесшапошников. Пиктограммный язык программирования Пикто. «Вестник кибернетики», Т. 28(2017), № 4, 2017, 173-180.

16. Scratch. Создавай истории, игры и мультфильмы. Делись с другими по всему миру. https://scratch.mit.edu/ (проверено 03.11.2019).

17. Д.П. Кириенко. Курс алгоритмизации с использованием исполнителей системы Кумир и автоматического тестирования, https://server.179.ru/wiki/?page=DenisKirienko/Kumir (проверено 03.11.2019).

18. А.Г. Кушниренко, Пять практикумов К. Ю. Полякова по программированию с автоматизированной проверкой в системе КуМир, Доклад на конференции OSEDUCONF-2015, (http://0x1.tv/Пять_практикумов_К._Ю._Полякова_по_программированию_с_автоматизированной_проверкой_в_системе_КуМир (проверено 03.11.2019).

19. А.Г. Кушниренко, А.Г. Леонов, М.В. Райко, И.Н. Грибанова. Курс «Азы программирования» для дошкольников, младшеклассников и студентов педуниверситетов. «Труды НИИСИ РАН», Т. 9 (2019), № 5, стр. 26-33.

20. M. Kölling, et al. Frame-Based Editing: Easing the Transition from Blocks to Text-Based Programming. «Proceedings of the Workshop in Primary and Secondary Computing Education (New York, NY, USA)», New York, NY, USA, 2015, 29–38.

21. D. Weintrop, et al. Teaching Text-based Programming in a Blocks-based World. «Proceedings of the 46th ACM Technical Symposium on Computer Science Education (New York, NY, USA)», New York, NY, USA, 2015, 678–678.

22. Н.О. Бесшапошников, А.Г. Кушниренко, А.Г. Леонов, А.А. Малый. Проект двуязыковой пиктограммно-текстовой учебной среды программирования ПиктоМир-К. «Свободное программное обеспечение в высшей школе», Переславль, 25-27 января 2019 г., Сборник тезисов Четырнадцатой конференции. – М., МАКС Пресс, 2019, 64–66.

Развитие психологических новообразований старших дошкольников в процессе обучения программированию на базе цифровой образовательной среды «ПиктоМир»

А.Г. Кушниренко[1], А.Г. Леонов[1,2,3], М.В. Райко[1], О.В. Собакинских[4], Л.В. Шибаева[5]

[1] ФГУ ФНЦ НИИСИ РАН, Москва, Россия;
[2] МГУ им. М.В.Ломоносова, Москва, Россия;
[3] МПГУ, Москва Россия;
[4] МБДОУ детский сад №20 «Югорка», Сургут, Россия;
[5] БУ «Сургутский государственный педагогический университет», Сургут, Россия

agk_@mail.ru, dr.l@vip.niisi.ru, mila.rayko@gmail.com, mixvel@yandex.ru, shibaeva2003@gmail.com

Аннотация. В образовательной парадигме мирового образовательного пространства, а вместе с этим и российского образования складываются определенные тенденции, по-новому представляющие как образ человека будущего, так и задачи, стоящие перед системой, как таковой. Федеральный государственный образовательный стандарт дошкольного образования направлен в том числе и на развитие способностей и творческого потенциала каждого ребенка как субъекта отношений с другими детьми, а также развития способностей и творческого потенциала ребенка. В статье рассматривается влияние курса «Алгоритмика» с использованием образовательной среды ПиктоМир на предметную, общепсихологическую ориентированность развития дошкольников, на варианты взаимодействия детей по принципу партнерства.

Ключевые слова: алгоритмика, партнерство, дошкольники, ПиктоМир, программирование

1 Введение

В национальной программе «Цифровая экономика Российской Федерации» [6] говорится о потребности в профессиональных кадрах, необходимых для развития цифровой экономики. Позицию, которая становится отправной точкой для разработки актуальных форм деятельности с обучающимися на всех уровнях образования – от дошкольного и выше, можно определить несколькими ключевыми аспектами: один из них – это цифровая компетентность, обусловленная глобальным вхождением ИКТ-технологий в жизнь современного человека. Она имеет и другую сторону – существует запрос на увеличение числа специалистов в сфере IT-технологий, что предполагает увеличение числа учащихся в по IT-направлениям. Другой аспект – смещение фокуса с профессиональных компетенций на надпрофессиональные, к важнейшим из которых относят умение кооперироваться, взаимодействовать в команде, способность к сотрудничеству при создании различных совместных продуктов [2,7]. В качестве важнейших надпрофессиональных компетенций в любой профессиональной сфере рассматриваются коммуникация и умение работать в команде.

Обращаясь к ФГОС дошкольного образования, мы можем выделить способность к сотрудничеству и выполнению как лидерских, так и исполнительских функций в совместной деятельности, а также владение средствами и способами взаимодействия (умение договориться, распределить действия при сотрудничестве) как важный целевой ориентир развития ребенка к завершению дошкольной ступени образования [9].

Таким образом, мы получаем представление о качествах, которые необходимо формировать у будущего специалиста цифровой экономики, закладывая основу на самой ранней ступени обучения – в дошкольном детстве.

В этой связи актуальным становится как поиск новых методов и систем работы с детьми, так и эффективное использование ресурсов уже существующих разработок. В данной деятельности необходимо руководствоваться важным условием – обучение должно способствовать развитию – что предполагает

психологический анализ и обоснование развивающего потенциала тех или иных методик.

2 Развивающий потенциал обучения программированию дошкольников.

Можно говорить о том, что введение детей в деятельность по подготовке к созданию компьютерных программ на текущем этапе осуществляется с 6-летнего возраста. Существует достаточно успешный опыт обучения детей программированию в процессе реализации курса занятий по Алгоритмике на основе программной среды ПиктоМир. Воспитанники старшего дошкольного возраста обучаются составлению линейных алгоритмов и алгоритмов с условием, знакомятся с понятиями программа, команда, программист. Эти занятия также позволяют формировать интерес ребенка к программированию, подготовить ребенка к деятельности по созданию компьютерных программ, что характеризует предметную ориентированность курса.

На данном этапе проходит исследование развивающего потенциала обучения старших дошкольников программированию на основе на основе системы ПиктоМир.

Варианты подготовки детей к вхождению в сферу программирования могут опираться на различные подходы. Учитывая требования к развивающему характеру обучения, среди них важно выделить такие, которые учитывают становление возрастных новообразований дошкольников и младших школьников, создают возможности их обогащения и динамики.

Одно и то же дидактическое средство может служить инструментом достижения локальных или масштабных развивающих результатов. Анализируя задания, в которые оказываются включены дети при применении системы обучения программированию на основе ПиктоМира можно обозначить не только предметную, но и общепсихологическую ориентированность на закономерности познавательного и личностного развития дошкольников [1,3,5].

На текущем этапе проходят исследования, направленные на изучение потенциала методического комплекта ПиктоМир и создаваемых на его основе образовательных программ на развитие познавательной и социально личностной сферы дошкольников. Реализация программы «Алгоритмика для дошкольников» на базе детского сада №20 «Югорка» г. Сургута позволяет охарактеризовать положительный развивающий эффект в познавательной сфере дошкольников[8]: из воспитанников подготовительных групп более 30% детей освоили курс на высоком уровне (без особых трудностей справились с предложенными заданиями), а более половины дошкольников испытывали лишь небольшие трудности, нуждались в более длительном времени для выполнения заданий, но в результате, также его успешно выполнили.

Психологической анализ методического комплекта обучения детей основам программирования ПиктоМир и процесса реализации, разработанной на его основе дополнительной общеобразовательной программы в подготовительных группах детского сада, позволил выделить спектр явных и потенциальных возможностей влиять на ход формирования возрастных новообразований. Были определены направления положительного влияния на динамику овладения способами знаково-символического мышления, овладение приемами пространственной ориентировки и умственными операциями планирования, прогнозирования сложных маршрутов передвижения разных по функции фигур при кодировании их маршрутов [3,4,5,8].

В ходе исследования была обоснована реализация в методах и приемах работы с дошкольниками принципов деятельностного подхода, в том числе при формировании у детей способов создания алгоритмов, прогнозирования маршрутов робота, планирования способов решения задачи.

Работа по выявлению развивающего эффекта заданий курса Алгоритмика предполагает дальнейший мониторинг динамики достижений воспитанников на основе разработанных диагностических процедур, а также возможное расширение направлений анализа влияния системы на формирование психических новообразований дошкольников.

Другое, не менее важное направление характеризуется включением в образовательный процесс заданий на кооперативное программирование, предусматривающих варианты взаимодействия детей по принципу партнерства [3,4]. Их применение позволяет преодолеть риски закрепления индивидуального стиля детей при использовании компьютеров, опереться на них в качестве средств развития такой социально-личностной компетенции как умение работать в команде.

Актуализация принципов развивающего обучения предполагают приоритеты

воспитания содержательных учебных взаимодействий. Это в дальнейшем оказывает влияние на способности к профессиональному и социальному партнерству.

Данное направление развития может быть поддержано на основе применения заданий системы ПиктоМир, предусматривающих партнерские задания. Системное применение подобных форм работы позволит воспитывать у детей культуру договоренностей, партнерских пристроек, прогнозирования соотношения планируемых способов деятельности и достижения совместного результата [3,4].

3 Заключение

Намечена траектория исследования влияния данной программы на развитие социально-коммуникативной сферы старших дошкольников в области освоения и применения ими форм партнерского взаимодействия.

В ходе исследования планируется разработать диагностический материал для анализа уровня готовности детей к реализации способов партнерского взаимодействия; осуществить мониторинг динамики показателей готовности к сотрудничеству в ходе выполнения детьми заданий по кооперативному программированию.

В качестве одного из важнейших вопросов выступает анализ методов и способов работы с дошкольниками при осуществлении ими работы по совместному выполнению задач на программирование, обоснование форм организации данной деятельности с воспитанниками 6-7 лет.

Публикация выполнена в рамках государственного задания по проведению фундаментальных исследований по теме «Разработка, реализация и внедрение семейства интегрированных многоязыковых сред программирования с автоматизированной проверкой заданий для учащихся образовательных организаций, ДОО, младшей, основной и старшей школы и студентов педагогических университетов» (№ 0065-2019-0010)

The development of new formations of senior preschool children in the process of teaching programming based on the digital educational environment "PiktoMir

"

A.G.Kushnirenko, A.G.Leonov, M. V. Raiko, O.V. Sobakinskikh, L.V. Shibaeva

Abstract. In the educational paradigm of the global educational space, and along with this, of Russian education, certain trends are taking shape, which in a new way represent both the image of the person of the future and the challenges facing the system as such. The federal state educational standard for preschool education is also aimed at developing the abilities and creative potential of each child as a subject of relations with other children, as well as developing the abilities and creative potential of the child. The article discusses the impact of the Algorithmic course using the PiktoMir educational environment on the subject-oriented, general psychological orientation of the development of preschool children, on options for the interaction of children on the principle of partnership.

Keywords: algorithmics, partnership, preschoolers, PiKtoMir, programming

Литература

1. Пиктомир - свободно распространяемая программная система для изучения азов программирования дошкольниками и младшими школьниками. https://www.niisi.ru/piktomir/.

2. А.М. Кондаков. Цифровое образование: матрица возможностей: презентация. http://ito2018.bytic.ru/uploads/materials/2.pdf.

3. И.Н. Грибанова, Я.Н. Зайдельман, А.Г. Кушниренко, М.В. Райко. Кооперативнопараллельное выполнение заданий при проведении дошкольных и школьных командных олимпиад по алгоритмике и программированию. VII Международная конференция «Воспитание и обучение

детей младшего возраста», Россия, Москва, 16–20 мая 2018.

4. И.Н. Грибанова, Я.Н. Зайдельман, А.Г. Кушниренко, М.В. Райко. Практикумы и олимпиады по кооперативному программированию в начальном курсе программирования для дошкольников и младшешкольников. «Вестник кибернетики», Т. 32 (2018), № 4, 159–169.

5. Методические указания по проведению цикла занятий «Алгоритмика» в подготовительных группах дошкольных образовательных учреждений с использованием свободно распространяемой учебной среды ПиктоМир. https://www.niisi.ru/piktomir/method2018.htm.

6. О проекте «Образование 2030» сайт ФГБУ «ФИОКО» (Федеральный институт оценки качества образования). https://fioco.ru/Contents/Item/Display/2201455.

7. О.В. Собакинских. Влияние деятельностного подхода на особенности формирования готовности к школе при овладении детьми 6–7 лет основами программирования. Международная научная конференция «Деятельностный подход к образованию в цифровом обществе», Россия, Москва, 13–14 декабря 2018.

8. Федеральный государственный стандарт дошкольного образования. https://fgos.

Курс «Азы программирования» для дошкольников, младшеклассников и студентов педуниверситетов

А.Г. Кушниренко [1], А.Г. Леонов [1,2,3], М.В. Райко [1], И.Н. Грибанова [1]

[1]ФГУ ФНЦ НИИСИ РАН, Москва, Россия;
[2]МПГУ, Москва, Россия;
[3]МГУ им. М.В.Ломоносова, Москва, Россия

agk_@mail.ru , dr.l@vip.niisi.ru , milya@niisi.ras.ru, nig@niisi.msk.ru

Аннотация. В статье изложены программные и методические принципы построения полуторагодового курса «Азы программирования» для детских садов, начальной школы и педуниверситетов. Описано предметное содержание курса, приведены примеры заданий, выполняемых обучаемыми. Программное и методическое обеспечение курса, включая подробные методички объемом более 300 страниц и комплект из 300+ компьютерных заданий, являются свободно распространяемыми и могут быть скачены с сайта ФГУ ФНЦ НИИСИ РАН https://www.niisi.ru/piktomir/.

Ключевые слова: ПиктоМир, методика преподавания программирования, алгоритмика, дошкольник, младшеклассник

1 Введение

Основой программного обеспечения курса является бестекстовая учебная система программирования ПиктоМир [1], разработанная в Федеральном Научном Центре «НИИСИ РАН». Система ПиктоМир функционирует на персональных компьютерах, работающих под управлением операционных систем MS Windows, Apple OS X, Linux, на планшетных компьютерах под управлением операционных систем Android и iOS. Более того, доступна и web-версия системы, работающая во всех популярных браузерах: Chrome, Firefox, Safari и др.

Программная система ПиктоМир и программно-методическое обеспечение курса – свободно распространяемые. Высокая кроссплатформенность и доступность программного и методического обеспечения позволяют использование как в системе дошкольного и начального школьного образования, так и в системе высшего образования - в педагогических университетах [2].

В системе ПиктоМир, как и в курсе в целом, алгоритмика изучается в процессе составления программ, управляющих виртуальными роботами-исполнителями. Каждый робот обладает небольшим, ограниченным набором команд-приказов и команд-вопросов. Число команд, в зависимости от робота, варьируется от трех-четырех до десяти-двенадцати.

Важной концепцией курса является овеществление виртуальных объектов. Так, например, в курсе предусмотрено использование в качестве дополнительного учебного пособия реального робота-игрушки, интегрированного с учебной системой ПиктоМир и управляемого из данной системы по звуковому каналу или радиоканалу [3].

Рис. 1.а. Реальный робот на полу

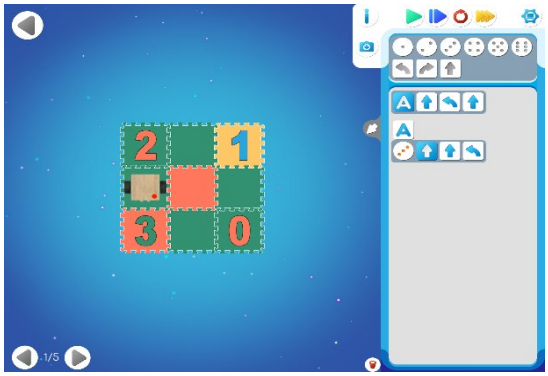


Рис. 1.б. Виртуальный Робот на экране

Обстановки, в которых действуют роботы-игрушки на полу игровой комнаты или их цифровые двойники на экране (см. рис. 1а, 1б), также собираются детьми в материальном мире из материальных объектов (сочленяемых разноцветных ковриков). Наконец, и программы управления роботами также могут собираться детьми из материальных объектов – кубиков, с нанесенными на грани пиктограммами команд роботов, повторителей и однобуквенных имен подпрограмм (см. рис. 2).

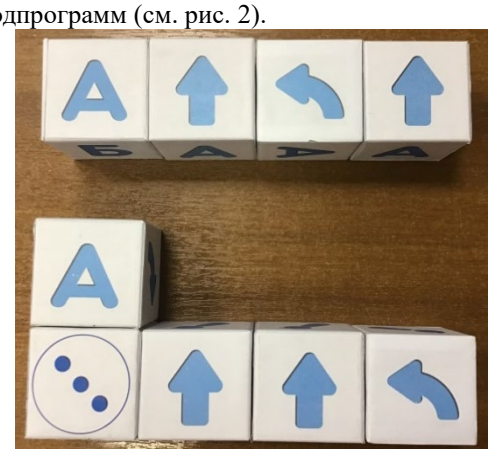


Рис. 2. Программа из пиктокубиков

Курс «Алгоритмика для дошкольников и первоклассников» расчитан на проведение в течение трех полугодий (полутора лет) по одному 35-45-минутному занятию в неделю, всего 45 занятий. Курс основан на корпусе из 300+ задач для системы ПиктоМир, разбитых на 45 компьютерных игр. В каждой игре от 5 до 10 уровней. Курс предусматривает решение учениками большинства (70%) задач корпуса в течение полутора лет, при этом какая-либо дополнительная работа обучаемых вне рамок занятий не предполагается. Высокий темп освоения материала (5 и более самостоятельно составленных программ на каждом занятии), стал возможным благодаря бестекстовой, пиктограммной форме составления программ учащимися и игровой форме проведения практикумов по программированию. Интерфейс как самой системы ПиктоМир, так и разработанных для курса 45 компьютерных игр и методики их использования отрабатывался в течение многих лет с учетом опыта работы с сотней образовательных учреждений и десятком тысяч детей в городах Москва и Сургут.

2 Структура курса «Азы программирования»

Курс основан на системе из 12 базовых понятий программирования, изложенных в статье «Основные понятия программирования в изложении для дошкольников» [4].

Курс состоит из 7 тем, последовательно знакомящих обучающихся с базовым набором понятий и конструкций программирования. Начинается курс с освоения простейших линейных программ, далее обучающиеся знакомятся с подпрограммами, циклами и условными конструкциями. В конце обучения курс предусматривает знакомство с азами параллельного программирования на примере двух роботов-исполнителей, действующих одновременно. При изучении курса обучающиеся решают задания по каждой теме по возрастающей сложности.

2.1 Первая часть курса: два полугодия, 30 занятий

Тема 1. Понятие программы и программного управления без обратной связи. Реальные и виртуальные роботы. Система команд исполнителя (робота). Постановка программных задач: карта-обстановка, в которой действует робот. Механизмы управления роботом-исполнителем: взаимодействие с роботом с помощью звука, света, радиосигналов. Выполнение команд исполнителем: подтверждение успешного выполнения команды-приказа, ответ на команду-вопрос, отказ. Реальный робот-игрушка и его виртуальный двойник. Реальная и виртуальная обстановки. Кодирование команд робота с помощью пиктограмм. Шаблон программы в системе ПиктоМир. Процесс управления роботом. Команды-приказы и команды-вопросы. Режим непосредственного управления роботом. Пульт управления роботом.

Программа, как план действий. Линейная программа. Программист – человек, составляющий программы. Возможность исполнения программы человеком. Компьютер – устройство для автоматического выполнения программ по управлению реальными и виртуальными роботами. Принцип программного управления – разделение процессов составления программ и выполнения программ.

Тема 2. Способы составления программ, уменьшение длины программы – повторители и подпрограммы. Пиктограммы повторителей. Обозначение подпрограммы одной пиктограммой. Технологии отладки программы: непрерывное и пошаговое исполнение программы. Технология составления программы: запоминание выполненных команд в «копилке», откатка. Совместное использование повторителей и подпрограмм.

На Тему 1 выделяется 10 занятий. Завершается этот цикл занятий игрой «Соревнование» (Олимпиада) [5].

Игра состоит из знакомых заданий, которые разбирались на предыдущих занятиях (см. рис. 3.а и 3.б, рис. 4.а и 4.б).

Рис. 3.а. Пример третьей задачи в игре «Соревнование» (шаблон задачи)

Рис. 3.б. Решение третьей задачи в игре «Соревнование»

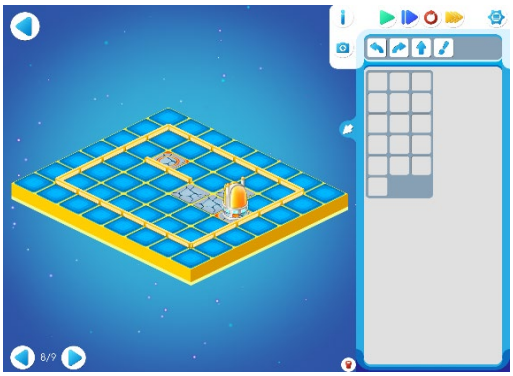


Рис. 4.а. Пример последней задачи в игре «Соревнование» (шаблон задачи)

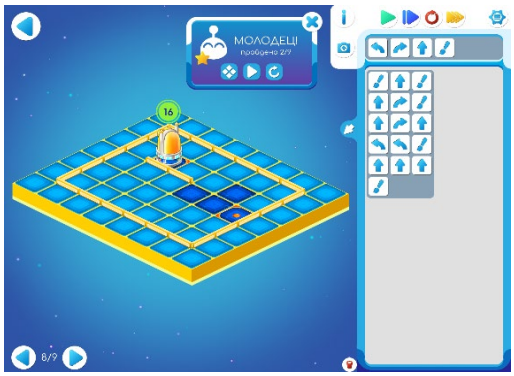


Рис. 4.б. Решение последней задачи в игре «Соревнование»

На Тему 2 выделяется 20 занятий: 5 занятий на знакомство с повторителями, 5 занятий на подпрограммы, 10 занятий на решение задач, требующих совместного использования повторителей и подпрограмм (см. рис. 5.а и 5.б, рис. 6.а и 6.б).

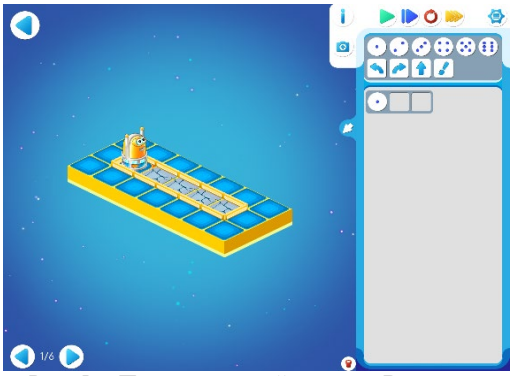


Рис. 5.а. Пример простой задачи с Вертуном (шаблон программы)

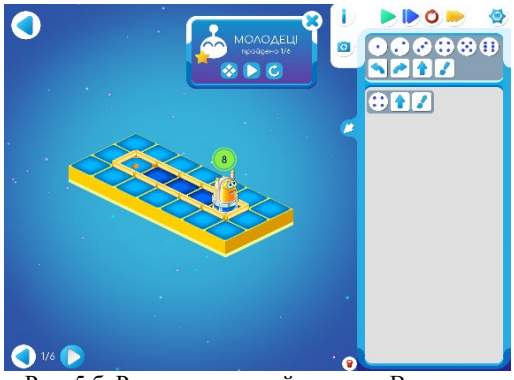


Рис. 5.б. Решение простой задачи с Вертуном

Рис. 6.а. Пример простой задачи с Двигуном (шаблон программы)

Рис. 6.б. Решение простой задачи с Двигуном

2.2 Вторая часть курса: одно полугодие, 15 занятий

Тема 3. Программное управление с обратной связью: цикл «пока». Решение задачи «дойти до стены». Способ записи решения - конструкция цикла «пока», команды-вопросы. Универсальное программирование — одна программа для нескольких однотипных обстановок.

Для решения заданий в данном разделе вводятся новые команды-вопросы. Получая команду-вопрос, Робот отвечает: да или нет. После введения команд с обратной связью появляется возможность дать детям задачи на составление «универсальных» программ. Для этого в заданиях дается несколько карт-обстановок, каждую из которых Робот может пройти, выполняя одну и ту же программу (см. рис. 7.а и 7.б).

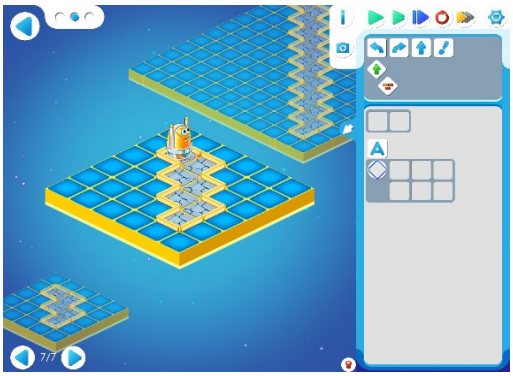


Рис. 7.а. Шаблон программы для задания с тремя обстановками

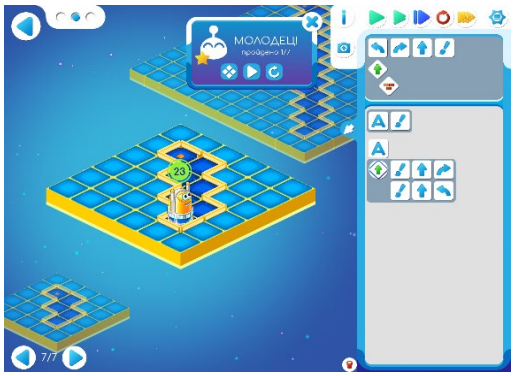


Рис. 7.б. Решение для задания с тремя обстановками

Тема 4. Программное управление с обратной связью: цикл со счетчиком. Исполнитель «Кувшин с камнями», его команды-приказы и команды-вопросы. Решение задачи «дойти до стены и вернуться». Использование значения счетчика (числа камней в Кувшине) в качестве повторителя.

Виртуальный исполнитель «Волшебный Кувшин» имеет свой набор команд-приказов, команд-вопросов, возвращающих логическое значение и команду-вопрос, возвращающую числовое значение, которое используется как повторитель. Применение счетчика позволяет решать задачи со счетом: «дойти до стены и вернуться в исходную точку», «закрасить клетку расположенную симметрично относительно отмеченной» и др. Ниже приведен пример задачи «дойти до стены, закрасить клетку и вернуться в исходную точку» (см. рис 8.а и 8.б)

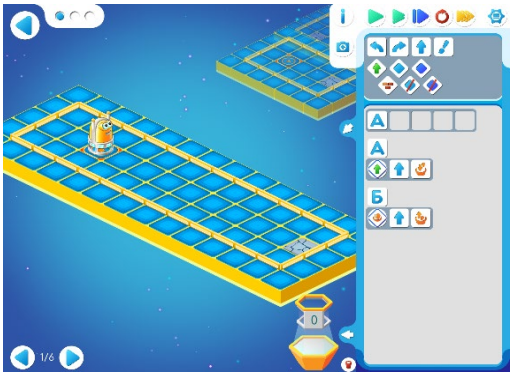


Рис. 8.а. Шаблон программы с Кувшином

Рис. 8.б. Решение

При выполнении подпрограммы «А», в Кувшине накапливается число камней, равное числу сделанных шагов, то есть подсчитывается расстояние до стены. (см. рис 9.а.). В подпрограмме «Б» при каждом шаге один камень выбрасывается из Кувшина и подпрограмма завершатся, когда в Кувшине не остается ни одного камня. (см. рис 9.б.) Это и обеспечивает возвращение Робота в исходную позицию.

Рис. 9.а. На табло Кувшина число 9, равное числу шагов Вертуна до стены

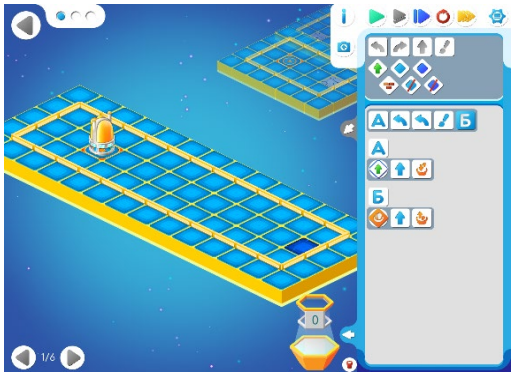


Рис. 9.б. После выполнения Подпрограммы «Б»: кувшин пуст (на табло 0) и Вертун вернулся в исходную позицию

Тема 5. Ветвление. Решение задач с выбором или пропуском действия в зависимости от выполнения условия. Примеры: «если клетка не синяя, то закрасить», «если впереди стена, то повернуться налево».

В этом разделе добавляется еще одна конструкция программирования — ветвление.

Пример задания с «если» (если «впереди стена», то «повернуться налево») (см. рис.10.а и 10.б):

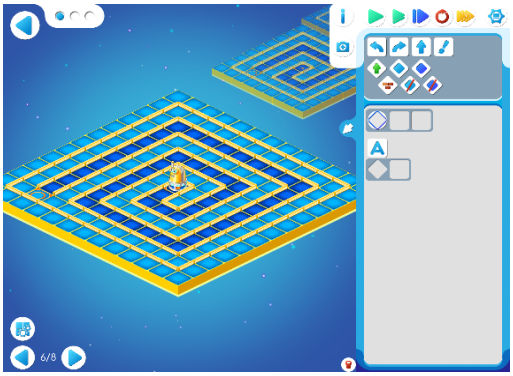


Рис. 10.а. Шаблон программы: в главном алгоритме используется «цикл пока», в подпрограмме используется ветвление

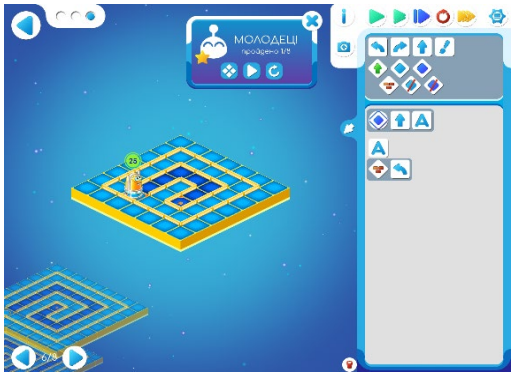


Рис. 10.б. Решение задания «выйти из лабиринта»

Тема 6. Параллельное управление множеством однотипных исполнителей (клонов) с помощью одной программы (SIMD) без обратной связи.

Совместная работа двух роботов в общей обстановке. Кооперативное составление программ для решения общей задачи. Параллельная работа двух программ. Отказы при параллельной работе. Синхронизация действий двух роботов с помощью команды «мигнуть».

Пример простого задания на параллельное программирование двух клонов роботов (клон голубых роботов и клон красных роботов) приведен на рисунках 11а, 11б.

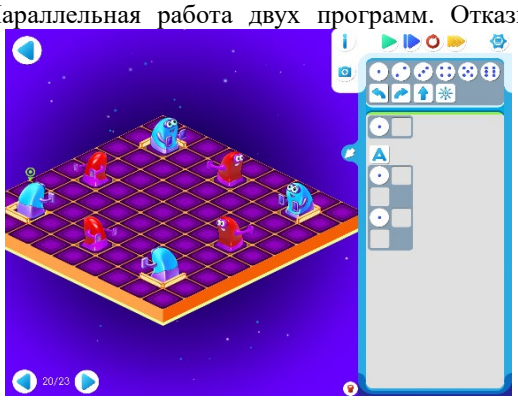


Рис. 11.а. Шаблон программы клона синих роботов

Рис. 11.б. Решение (программа для синих роботов)

Тема 7. Пиктограммные (детские) и текстовые (взрослые) языки программирования. Демонстрация примеров перевода программы управления реальным или виртуальным роботом из пиктограммной формы в текстовую.

Замечание. Темы 1-5 охватывают замкнутый набор понятий и практик программирования и могут быть освоены в комфортной обстановке, с использованием системы бестекстового программирования.

Темы 1-3 осваиваются детьми возраста 6-8 лет за 30 академических часов, темы 4-7 — за 15 академических часов. Студенты педагогических специальностей осваивают курс за 8-10 академических часов. Для них курс завершается контрольными работами. Изложение основ программирования в бестекстовой среде является введением в общий курс программирования для студентов педагогических специальностей и существенно облегчает им освоение текстового программирования.

Примеры задач одного из вариантов контрольной работы студентов (см. рис. 12.а и 12.б, рис. 13.а и 13.б):

Рис. 12.а. Шаблон программы простой задачи

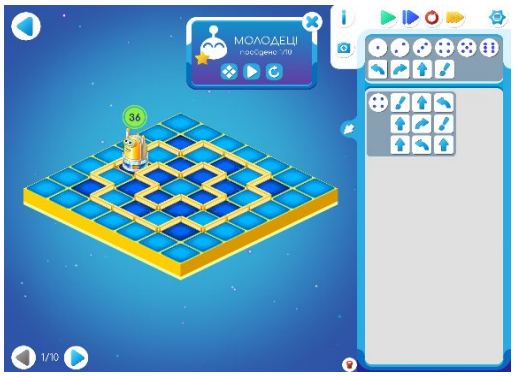


Рис. 12.б. Решение простой задачи

Рис. 13.а. Шаблон программы непростой задачи

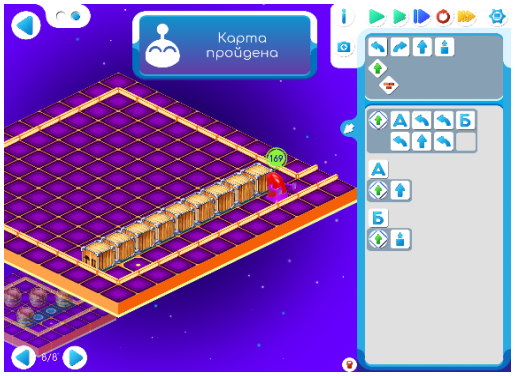


Рис. 13.б. Решение непростой задачи

6 Заключение

По завершению 45 занятий курса «Азы программирования» с использованием пиктограммного стиля программирования, учащиеся подготовлены к освоению текстового стиля программирования. По нашей методике это освоение проводится в системе ПиктоМир-К, которая позволяет начать с решения знакомых задач на управление знакомыми исполнителями, освоенными в системе ПиктоМир.

Для сравнения приведем пример решения одной и той же задачи управления исполнителем Вертун в системах ПиктоМир и Пиктомир-К (рис. 14.а и 14.б).

Рис. 14.а. Решение задания «выйти из лабиринта» в системе ПиктоМир-К

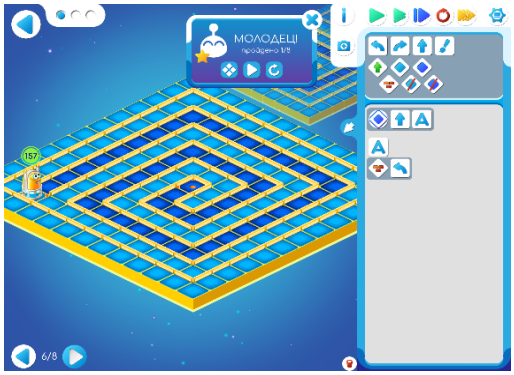


Рис. 14.б. Решение задания «выйти из лабиринта» в бестекстовой системе ПиктоМир

Публикация выполнена в рамках государственного задания по проведению фундаментальных исследований по теме «Разработка, реализация и внедрение семейства интегрированных многоязыковых сред программирования с автоматизированной проверкой заданий для учащихся образовательных организаций, ДОО, младшей, основной и старшей школы и студентов педагогических университетов» (№ 0065-2019-0010).

Course “ABC of programming” for preschoolers, junior high school students and students of pedagogical universities

A.G. Kushnirenko, A.G. Leonov, M.V. Raiko, I.N. Gribanova

Abstract. The article outlines the program and methodological principles for building a one and a half year course "ABC of Programming" for kindergartens, elementary schools and pedagogical universities. The subject content of the course is described, examples of tasks performed by students are given. The program and methodological support of the course, including detailed manuals of more than 300 pages and a set of 300+ computer tasks, are freely distributed and can be downloaded from the website of the Federal State Institution Scientific Center of the Russian Academy of Science “NIISI RAS”, https://www.niisi.ru/piktomir/ .

Keywords: PiktoMir, programming teaching methodology, algorithmics, preschooler, junior high school student

Литература

1. Пиктомир - свободно распространяемая программная система для изучения азов программирования дошкольниками и младшими школьниками. https://www.niisi.ru/piktomir/.

2. N.O. Besshaposhnikov, A.G. Kushnirenko, A.G. Leonov. Pictomir: how and why do we teach textless programming for preschoolers, first graders and students of pedagogical universities. “CEE-SECR '17 Proceedings of the 13th Central & Eastern European Software Engineering Conference in Russia, ACM New York, NY, USA”, (2017). https://dl.acm.org/citation.cfm?doid=3166094.3166115.

3. А.Г. Кушниренко, А.А. Малый, А.А. Левин, А.Г. Леонов. "Музыкальное" управление учебным роботом в курсе "Алгоритмика для дошкольников". "Воспитание и обучение детей младшего возраста: VIII Международная конференция (ECCE 2019). Москва, 29 мая — 1 июня 2019 г. (тезисы)", М., Изд-во Московского университета, 2019, 131–132.

4. В.Б. Бетелин, А.Г. Кушниренко, А.Г. Леонов. Основные понятия программирования в изложении для дошкольников. "Информатика и её применения", Т. (2020), № 3.

5. Грибанова И.Н., Райко М.В. Методика использование олимпиад в курсе «Алгоритмика для дошкольников» // Современный научный потенциал и перспективные направления теоретических и практических аспектов: Сборник научных статей по итогам международной научно-практической конференции, г. Санкт-Петербург 22-23 декабря 2017. – СПб.: Изд-во «КультИнформПресс», 2017. – С. 122-126.

Головоломки и игры в курсе «Алгоритмика для дошкольников»

В.А. Ковыршина[1], А.Г. Кушниренко[2]

[1] МГУ им. М.В.Ломоносова, Москва, Россия, potchtovy_jashik@mail.ru;
[2]ФГУ ФНЦ НИИСИ РАН, agk_@mail.ru

Аннотация. Предложена схема включения в курс программирования для дошкольников компьютерных заданий нового типа. Выполнение этих заданий, требуют не прямого составления программы управления роботом-исполнителем, а подготовки инструментов (вспомогательных алгоритмов) помогающих решать поставленную задачу в режиме пультового управления. Программное обеспечение для выполнения подобных заданий нового типа построено модернизацией бестекстовой системы программирования ПиктоМир.

Ключевые слова: информатика, Робот, Вертун, ПиктоМир, дошкольник, алгоритм, программа, язык программирования, вспомогательный алгоритм, головоломка, вероятностная игра

1 Введение

В курсе «Алгоритмика для дошкольников», разработанном в отделе учебной информатики ФГУ ФНЦ НИИСИ РАН в течение последних лет [1], каждое занятие разделяется на две половины – бескомпьютерную и компьютерную. Проанализировав подробную методичку курса [2], можно прийти к выводу, что в то время как индивидуальные и коллективные активности на бескомпьютерных половинах занятий достаточно разнообразны, компьютерные половины занятий в большинстве случаев проводятся по одной и той же схеме. На компьютерной части занятия ребенок выполняет практикум по программированию, как правило индивидуальный и изредка командный. Практикум обличен в форму прохождения уровней компьютерной Игры, заранее подготовленной авторами курса для данного занятия. Теоретически, на разных уровнях компьютерной игры ребенку могли бы быть предложены разные активности, но в настоящее время прохождение каждого уровня такой Игры проводится по одной и той же схеме - ребенку предъявляется картинка, изображающая робота в некоторой обстановке, и ребенок должен составить и отладить программу управления роботом, проводящую некоторые изменения в этой обстановке. Пример приведен на рисунке 1.

Рис. 1. Робот Двигун должен передвинуть ящик и бочку в клетки, отмеченные крестиком и кружочком соответственно, пользуясь счетчиком «Кувшин с камнями»

Дополнительно ребенку выдается шаблон программы, который играет двоякую роль. С одной стороны, он ограничивает возможности ученика по составлению программы, с другой стороны является сильной подсказкой.

На небольшой части занятий индивидуальный практикум заменяется на командный. Команде из двух детей, каждый из которых работает на своем компьютере, предъявляется картинка, изображающая двух роботов в некоторой обстановке. Каждый член команды составляет программу управления своим роботом, и задача команды - составить две параллельно выполняющиеся программы так, чтобы, работая совместно, роботы провели нужные изменения в общей обстановке. Пример приведен на рисунке 2.

Рис. 2. Работая совместно, два робота Вертуна должны починить девять поврежденных клеток мобильной космической платформы

Тем самым, до настоящего времени компьютерная половина каждого из 30 занятий курса «Алгоритмика для дошкольников», была посвящена одной и той же активности – составлению и отладке детерминированной однопоточной или параллельной двухпоточной программы управления роботами.

2 Общая схема новых активностей

В настоящей заметке предложены инструмент и методика введения новых активностей на компьютерных половинах занятий курса «Алгоритмика для дошкольников». Программное обеспечение для этих активностей, система «КубоРобот», получено модификацией системы ПиктоМир и расчитано на автономное использование. На следующем этапе работы предполагается интеграция «КубоРобота» с ПиктоМиром и включение новых активности в компьютерные Игры курса «Алгоритмика для дошкольников».

Замечание. Методически, новые активности призваны ускорить и улучшить усвоение одного из трудных понятий курса – понятие вспомогательного алгоритма.

Новые активности состоят в непосредственном, пультовом управлении роботом с целью скорейшего прохождения лабиринта из заданной стартовой точки в заданную финишную точку. Предлагаются три сценария:

- *Детерминированная головоломка.*
- *Вероятностная головоломка.*
- *Вероятностная игра с двумя или более участниками, которые соревнуются в скорости прохождения одного и того же лабиринта.*

В любом из сценариев, начальным этапом деятельности ребенка будет составление двух коротких вспомогательных алгоритмов с именами А и Б. Эти алгоритмы строятся ребенком заранее, после того, как ему в графической форме дается условие очередной головоломки или очередного тура игры. Далее ребенок использует эти придуманные им алгоритмы А и Б для непосредственного, пультового управления роботом Вертуном в процессе индивидуального решения головоломки или в процессе игры.

В случае *детерминированной головоломки*, ребенок старается выбрать вспомогательные алгоритмы так, чтобы с их помощью задача была решена за возможно меньшее число шагов.

В случае *вероятностной головоломки*, ребенок старается выбрать вспомогательные

алгоритмы так, чтобы с их помощью выполнить заданный в условии головоломки норматив.

В случае *вероятностной игры*, ребенок старается выбрать вспомогательные алгоритмы так, чтобы с их помощью довести своего робота до финиша быстрее, чем соперник (соперники).

3 Пример детерминированной головоломки

Требуется заполнить шаблоны А и Б так, чтобы за возможно меньшее число шагов перевести Вертуна из стартовой точки изображенного на рисунке 3 лабиринта в финишную точку. Шаблон алгоритма А состоит из трёх клеток и шаблон алгоритма Б из четырёх клеток. (см. рис. 3).

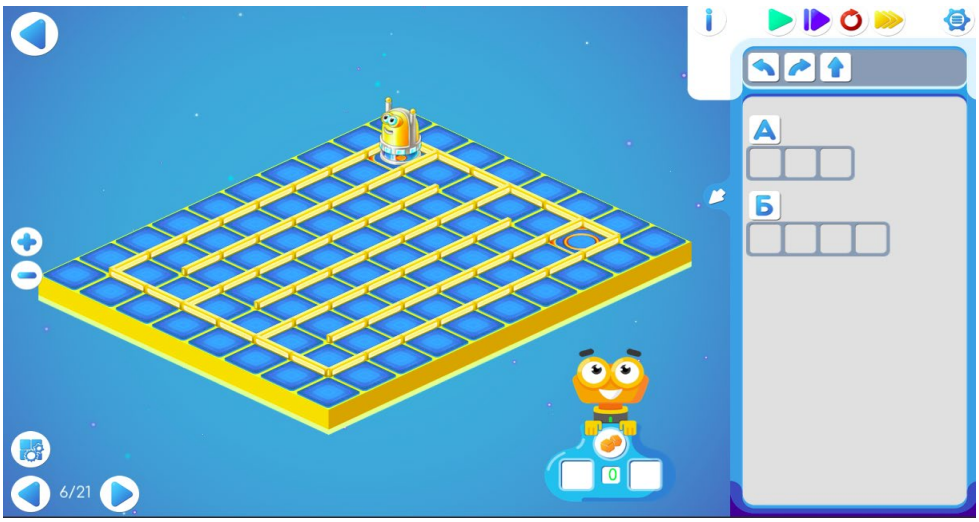


Рис. 3. Пример детерминированной головоломки

Правила заполнения шаблонов. Шаблон алгоритма А заполняется командами из таблицы 1, а шаблон алгоритма Б заполняется командами из таблицы 2.

Таблица 1. Разрешенные команды для алгоритма А

вперед	
повернуться налево	
повернуться направо	

Таблица 2. Разрешенные команды для алгоритма Б

вперед	
повернуться налево	
повернуться направо	
А	

Закончив заполнение шаблонов, ребенок открывает Копилку ПиктоМира, после чего изменение алгоритмов А и Б становится невозможным (блокируется).

С помощью Копилки ребенок старается

перевести Вертуна из начальной точки в конечную, сделав минимальное количество ходов, то есть выдавая по своему усмотрению команды (см. таблицу 3).

Таблица 3. Разрешенные команды для заполнения Копилки

вперед	
повернуться налево	
повернуться направо	
А	
Б	

В условии головоломки указывается норматив - максимальное число ходов, при котором головоломка считается решенной. Норматив определяется в зависимости от конкретной задачи, задачи подбираются так, чтобы при удачном подборе А и Б решение головоломки требовало от десятка до двух десятков ходов.

4 Исполнитель КубоРобот

Для описания вероятностных головоломок и вероятностных игр нам понадобится исполнитель КубоРобот. У этого исполнителя одна единственная команда – **бросить** пару ПиктоКубиков. Выбранные грани кубиков изображаются на табло КубоРобота (см. рис.4).

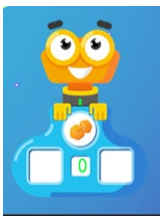


Рис. 4. Исполнитель Кубо Робот

У пиктокубика 6 граней. На них изображены пиктограмма «кисточка»

и 5 пиктограмм команд (см. таблицу 3),

В процессе игры пиктограмму «кисточка» (джокер) разрешается заменить на любую из пяти перечисленных выше команд (см. таблицу 3).

По команде **бросить** КубоРобот показывает на своем табло случайно выбранные грани двух пиктокубиков.

5 Пример вероятностной головоломки

Требуется за возможно меньшее число команд бросить перевести Вертуна из стартовой точки в финишную точку лабиринта. При каждом вызове команды **бросить** КубоРобот показывает две пиктограммы. Разрешается выбрать из них 0, 1 или 2 пиктограммы и выполнить их в любом порядке. Кисточка понимается как «джокер», ее можно заменить на любую из оставшихся 5 пиктограмм.

На рисунке 5 показано, что после выбора А и Б, игрок открыл Копилку, выполнил команду **бросить** и из выпавших граней А и «вперед» выбрал А.

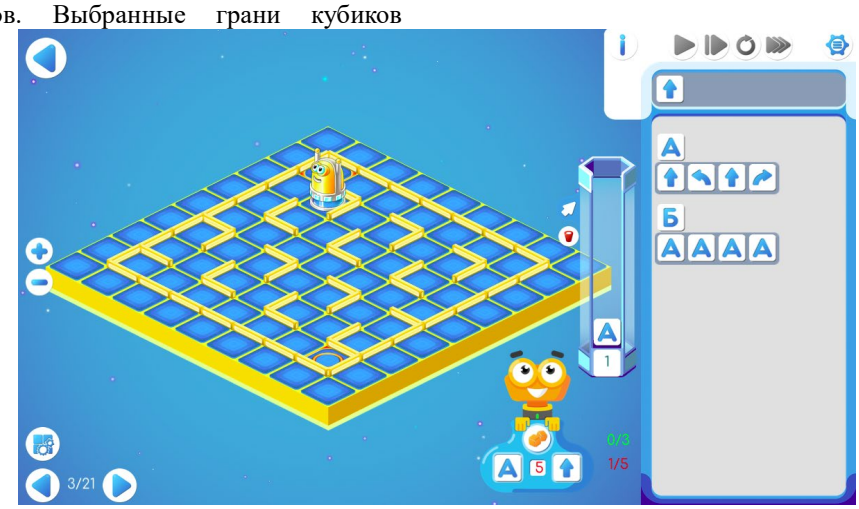


Рис. 5. Выполнение первого хода головоломки

После выполнения следующей команды **бросить** выпали грани «кисточка» и А. Согласно правилам, игрок может заменить кисточку на Б и положить в Копилку команды Б и А (см. рис. 6, 7). Это завершает решение головоломки.

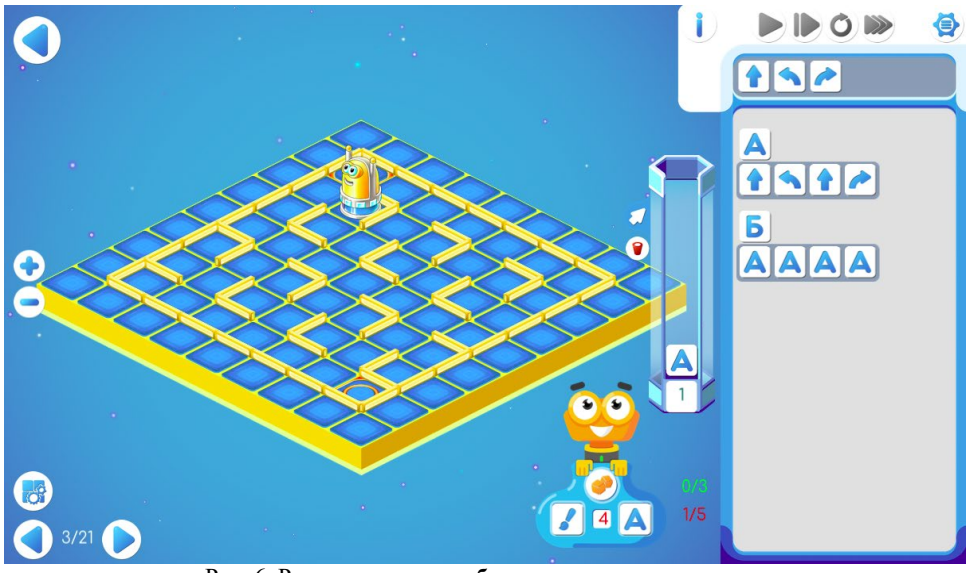


Рис. 6. Результат команды **бросить** после первого хода

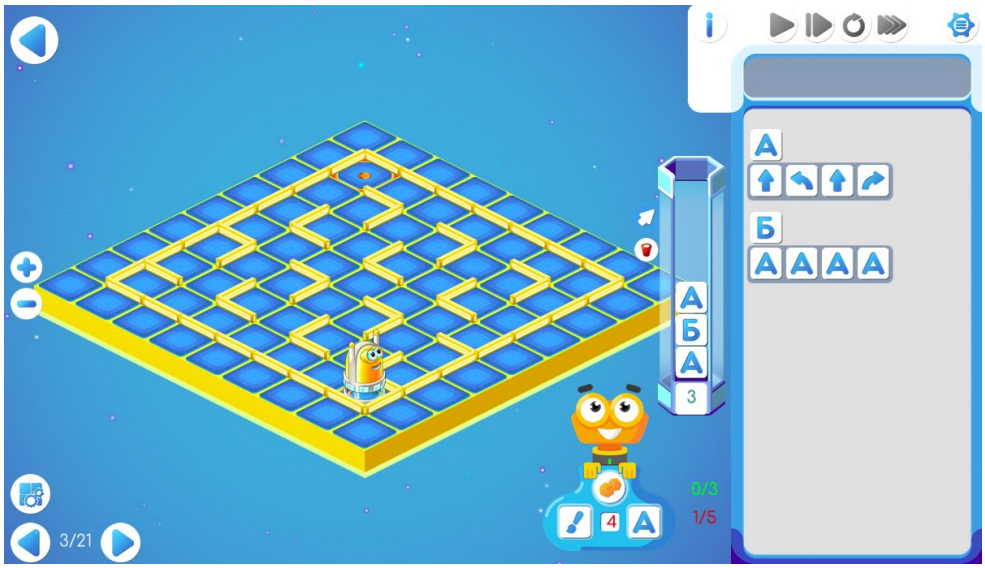


Рис. 7. Результат второго хода – головоломка решена

5 Пример вероятностной игры

Каждый игрок получает свой компьютер. Игра проходит в два этапа.

На первом этапе игрокам показывается на экране одна и та же карта-лабиринт, после чего игроки независимо друг от друга в течение одной минуты (песочные часы) составляют команды-алгоритмы А и Б таким образом, чтобы они помогали как можно быстрее пройти лабиринт. (рис. 8).

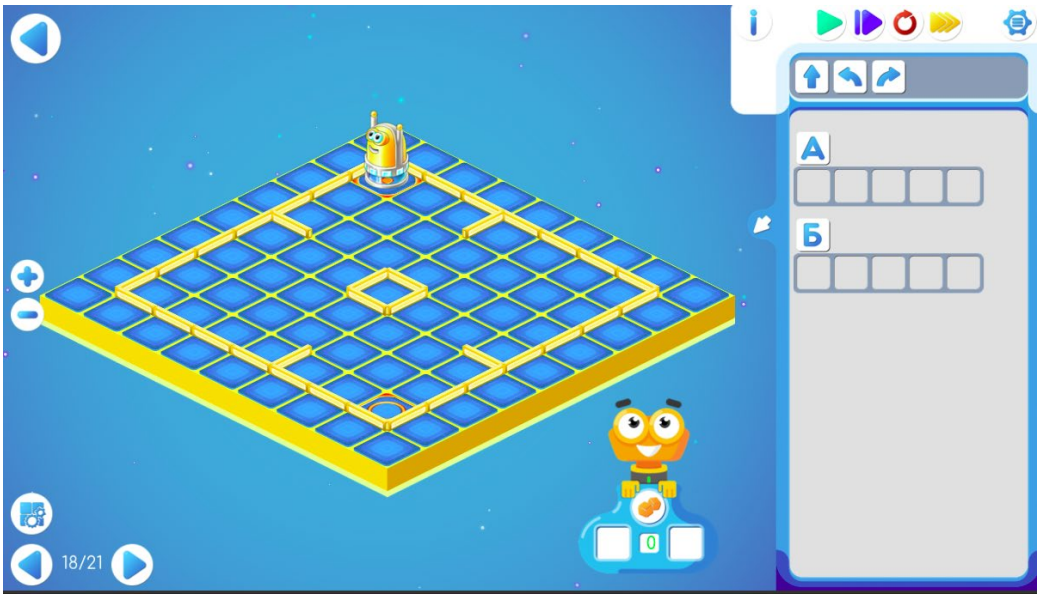


Рис. 8. Пример вероятностной игры

По истечении минуты начинается второй этап Игры. На карте изображаются одновременно роботы двух игроков, у каждого игрока открывается Копилка и начинается сам игровой процесс: Куборобот, изображенный на рисунке 3, выполняет команду бросить (это занимает 5 секунд) и показывает игрокам на своем табло результат бросания. В течение 15 секунд (песочные часы) каждый игрок должен выбрать (отправить в копилку) одну или две команды, или ни одной команды. Если игрок завершил выбор до истечения лимита времени 15 секунд, он может нажать кнопку «готово». Если выпала одна или две "кисточки", игрок может выбрать одну или две нужные ему команды. При выборе команды игроком его робот продвигается по лабиринту и это продвижение видно одновременно всем игрокам.

Каждый ход игры занимает не более 20 секунд. Игра продолжается до тех пор, пока хотя бы один робот не достигнет цели, или пока не будет исчерпано установленный автором данной игры лимит числа шагов. Условия игры подбираются так, чтобы она продолжалась не более 5 минут.

6 Заключение

Прежде всего заметим, что вероятностные головоломки и игры формируют у ребенка базовые представления о равновероятности событий и вероятности событий. Кроме того, предложенные игры и головоломки вырабатывают навыки составления и использования вспомогательных алгоритмов. А именно, на первом этапе Игры ребенку приходится представить себе мысленно кратчайший маршрут, ведущий из стартовой точки данного лабиринта в финишную и попытаться закодировать в алгоритмах А и Б наиболее часто встречающие фрагменты кратчайшего маршрута. А на втором этапе Игры ребенок на каждом шаге Игры должен твердо помнить эффект выполнения алгоритмов А и Б и быстро принимать решения, какие из доступных команд выполнять и в каком порядке.

Таким образом, новые активности формируют представления о вероятностях, побуждают к мысленному поиску путей в лабиринтах, вырабатывают навыки составления и использования вспомогательных алгоритмов.

Публикация выполнена в рамках государственного задания по проведению фундаментальных исследований по теме «Разработка, реализация и внедрение семейства интегрированных многоязыковых сред программирования с автоматизированной проверкой заданий для учащихся образовательных организаций, ДОО, младшей, основной и старшей школы и студентов педагогических университетов» (№ 0065-2019-0010).

Puzzles and games in the course "Algorithmics for Preschoolers

V.A. Kovyrshina, A.G. Kushnirenko

Abstract. A scheme of inclusion in the programming course for preschoolers of computer tasks of a new type is proposed. The fulfillment of these tasks does not require the direct preparation of a control program for the robot-executor, but the preparation of tools (subprograms) to help solve the assigned task in the remote control mode. The software for performing similar tasks of a new type was built by modernizing the PictoMir text-free programming system.

Keywords: informatics, Vertun the Robot, PictoMir, preschooler, algorithm, program, programming language, subprogram, puzzle, probability game

Литература

1. А.Г. Кушниренко, А.Г. Леонов, М.В. Райко, Курс «Азы программирования» для дошкольников, младшеклассников и студентов педуниверситетов. «Труды НИИСИ РАН», Т. 9 (2019), № 6, стр. 26-33.

2. А.Г. Кушниренко, А.Г. Леонов, М.В. Райко. Методические указания по проведению цикла занятий «Алгоритмика» в подготовительных группах дошкольных образовательных учреждений с использованием свободно распространяемой учебной среды ПиктоМир (2018 г.) https://www.niisi.ru/piktomir/Алгоритмика 2018 1-30 23.04.2019.1.pdf (проверено 06.12.2019)

Об одном подходе к построению системы мониторинга сети суперкомпьютерных центров

С.А. Лещев[1], А.И. Тихомиров[2]

[1] Межведомственный суперкомпьютерный центр РАН – филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия, sergey.leshchev@jscc.ru;
[2] Межведомственный суперкомпьютерный центр РАН – филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия, tema4277@rambler.ru

Аннотация: В статье рассматривается задача организации механизма глобального мониторинга сети суперкомпьютерных центров. Проанализирован мировой опыт по организации мониторинга в суперкомпьютерных центрах и grid-системах. Показано, что при построении полного профиля вычислительного задания для решения задач мониторинга производительности в сети суперкомпьютерных центров оказывается недостаточно обработанных данных существующих и перспективных систем мониторинга. Предложен способ сохранения всех данных о работе оборудования и программного обеспечения ВУ в “озерах данных” на ресурсах суперкомпьютерных центров, обоснована достаточность этих ресурсов. Предложена архитектура системы глобального мониторинга на базе стека программного обеспечения ELK (Elasticsearch, Logstash, Kibana), рассмотрены ее достоинства и недостатки, разработан план перехода на новую систему мониторинга.

Ключевые слова: суперкомпьютер, сеть суперкопьютерных центров, мониторинг, мониторинг вычислительного задания, СХД, озеро данных, Elasticsearch, ELK, ETL

1. Введение

В настоящее время в Межведомственном суперкомпьютерном центре РАН (МСЦ РАН) ведутся работы по проекту единой распределённой сети научных суперкомпьютерных центров (СКЦ) [1]. В рамках проекта ведется решение задач по созданию:

- децентрализованной автоматизированной системы управления заданиями и вычислительными ресурсами сети [2];
- единой системы доступа на основе удостоверяющей федерации СКЦ [3];
- общей облачной системы хранения данных, обеспечивающей единое файловое пространство для всех вычислительных установок (ВУ) СКЦ сети [4];
- набора средств контейнерной виртуализации ([5], [6] и [7]) для решения проблем бинарной переносимости параллельных программ между ВУ СКЦ сети.

Еще одной из решаемых в рамках проекта задач является создание единой системы мониторинга, которая позволит оперативно получать информацию о текущих состоянии и загруженности суперкомпьютерных ресурсов распределенной сети.

2. Архитектура систем мониторинга вычислительных установок

Классически в промышленных системах мониторинга выделяют три уровня:

- уровень сбора данных;
- уровень хранения данных;
- уровень представления данных.

Уровень сбора данных реализуется при помощи специальных процессов-агентов на узле оборудования, либо используется безагентный механизм передачи данных. В этом случае роль агентов выполняют системные сервисы операционной системы узла, например службы SNMP или SSH [8].

Все поступающие с уровней сбора и обработки данные помещаются в уровень хранения, после чего становятся доступны для поисковых запросов пользователей. Объем поступающего от агента потоков данных может для крупных вычислительных установок может достигать десятков и сотен Гб в день. Выбор используемых систем хранения зависит от характера поступающих данных. Для сильно структурированных потоков данных, содержащих только отдельные метрики, на уровне хранения

данных целесообразно использовать InfluxDB TICK Stack или Yandex ClickHouse. В случае если поток данных не структурирован, поступает из разных источников или собирается разными агентами - данные сохраняются в неструктурированном виде. Для поиска по неструктурированным данным, используются механизмы полнотекстового поиска, реализованные в таких инструментах как Elasticsearch, Apache Solr, Sphinx. Для хранения неструктурированных широко применимы не реляционные модели хранения данных. В этом случае в качестве системы хранения может использоваться: Elsaticserch, Cassandra, MongoDB.

На уровне представления данных, как следует из его названия, размешаются средства представления поступивших на уровень хранения данных в удобном для восприятия человеком виде – табличном или графическом. Графическая информация может быть представлена в виде диаграмм, графиков или панелей индикаторов (dashboard) – графического представления ключевых показателей эффективности. Помимо простого представления данных на этом уровне также размещены механизмы формирования событий из поступивших на уровень хранения данных, а также средства оповещения инженеров служб эксплуатации. На сегодняшний момент наиболее распространенным способом представления информации мониторинга пользователю является отображение ее в виде динамически обновляемых html-страниц, например в Kibana или Grafana.

В современных системах мониторинга, как правило, объем собираемых данных становится настолько большим, что из массива неструктурированных данных мониторинга отбираются лишь определенные метрики. Для этого собранные данные проходят еще один уровень – обработки, располагающийся между уровнем сбора и уровнем хранения.

Появление в современных системах мониторинга средств машинного обучения (machine learning, ML) и искусственного интеллекта (artificial intelligence, AI), связанное с сокращением времени реакции на аварийные события и автоматизацией управления оборудованием привело к выделению еще одного уровня, расположенного над уровнем представления данных – уровня реакции.

3. Цели мониторинга

вычислительных установок

Основная цель традиционных систем мониторинга ВУ – отслеживание аварийных ситуаций, требующих немедленной реакции для сохранения дорогостоящего оборудования. Следующей целью является отслеживание исправности вычислительных узлов и их аппаратных компонентов. Своевременное обнаружение и отключение частично неисправного узла позволяет быстро вывести его в ремонт. Еще одна цель мониторинга – отслеживание пригодности вычислительных узлов для исполнения вычислительных задач. Своевременное обнаружение вычислительных сбоев позволяет избежать необоснованной траты ресурсов вычислительной установки.

В таблице 1 представлены параметры надежности и частоты опроса традиционных систем мониторинга вычислительных установок.

Таблица 1. Требования к надежности и частоте опроса традиционных систем мониторинга вычислительных установок

Цель мониторинга	**Требуемая надежность системы**	**Частота опроса**
Аварийные ситуации	Очень высокая	1-5 секунд
Работоспособность узлов	Средняя	1-5 минут
Функциональность узлов	Средняя	до 1 часа
Мониторинг эффективности	Низкая	0.2 секунды и чаще

В таблице 2 представлены параметры сложности обработки и объема данных традиционных систем мониторинга вычислительных установок.

Таблица 2. Параметры сложности обработки и объема данных традиционных систем мониторинга вычислительных установок

Цель мониторинга	**Сложность обработки данных**	**Объем данных**
Аварийные ситуации	Низкая	Небольшой
Работоспособность узлов	Средняя	Средний
Функциональность узлов	Высокая	Средний

Мониторинг эффективности	Низкая	Высокий

Самые низкие требования по надежности предъявляются к мониторингу производительности оборудования. Мониторинг производительности – это отслеживание различных параметров функционирования ОС и оборудования для того чтобы обнаружить факторы, мешающие выполняемой параллельной программе достигнуть максимально возможной производительности.

4. Сеть ВУ. Предпосылки перехода к мониторингу вычислительных заданий

При объединении СКЦ в сеть увеличивается сложность мониторинга выполнения программы на различных ВУ в силу неоднородности оборудования ВУ такой сети. Вычислительные задачи уже не имеют достаточно жесткой специализации по вычислениям, как это было обыкновенно для GRID-систем. Для ВУ такой сети характерно также отсутствие унификации систем сбора и представления данных мониторинга. Все это существенно затрудняет организацию распределенного мониторинга как состояния ВУ сети, так и процесса выполнения вычислительных задач.

С ростом производительности ВУ увеличивается количество и возрастает сложность технических подсистем СКЦ. Еще в первых сетях СКЦ - GRID'ax появляется мониторинг сетей WAN [9], [10]. Для исследования перспективных систем теплоотведения высокоплотных ВУ на горячей воде начинает использоваться мониторинг тепловыделения и энергопотребления [11], [12]. К задачам мониторинга ВУ начинают привлекать технологии машинного обучения [13].

В СКЦ кроме HPC начинают вестись работы на стыке различных областей Computer Scince, для удобства работы пользователей они начинают предоставлять в т.ч. появляются услуги контейнерной и гипервизорной виртуализации, облачные услуги (списки предлагаемых услуг доступны на сайтах суперкомпьютерного центра Лейбница, суперкомпьютерного центра “Политехнический” и лаборатории информационных технологий ОИЯИ). Все эти сервисы, предлагаемые СКЦ клиентам, также требуют своих служб мониторинга.

Рост вычислительной мощности, увеличение количества и усложнение технических систем, расширение функционала, предоставляемого СКЦ приводит к заметному росту объемов неструктурированных данных различных систем мониторинга. Предпринятые научным сообществом попытки анализа накопленных данных при помощи методов “больших данных” позволяют успешно выявлять неявные закономерности в работе оборудования и ПО ВУ, не обнаруживаемые другим способом, используя полученную информацию для повышения энергоэффективности и производительности ВУ. Одним из наглядных примеров такого применения данных мониторинга - использование данных энергопотребления узлов ВУ при выполнении пользовательских заданий позволяют строить профили управления питанием узлов ВУ в перспективных установках экзафлопсного класса [14], [15].

Т.о. исходные неструктурированные данные систем мониторинга постепенно превращаются в ценные научные данные, требуется организовать их долговременное хранение и обработку.

5. Организация хранения данных мониторинга

Необходимость хранения большого объема неструктурированных данных мониторинга, а также сложность создания централизованного хранилища этих данных (слишком велики требования к емкости и производительности хранилища, а постоянная передача больших объемов неструктурированных данных мониторинга по каналам связи приводит к их необоснованной постоянной загрузке) приводит нас к мысли организовать распределенное хранение данных на имеющихся ресурсах СКЦ в т.н. “озерах данных. Озеро данных (data lake) [16] – это инфраструктурное решение для хранения большого объема любых типов данных, поступающих из разных источников, а также предоставления единого интерфейса доступа к хранящимся данным [17]. Озера данных используются в крупных компаниях, таких как Microsoft, Amazon, Google. В частности, примером озера данных может служить технология Azure от компании Microsoft. В озере данных Azure предоставляются все возможности хранения данных любых объема, формата и скорости передачи, а также выполнения различных видов обработки и анализа на разных платформах и

языках. Озеро данных Azure решает задачу получения и хранения всех данных с возможностью пакетной, потоковой и интерактивной обработки. Возможность хранения в озере данных как структурированных, так и неструктурированных данных, позволяет направить в него потоки данных от уже развернутых на ВУ систем мониторинга, так и неструктурированные потоки данных не обрабатываемых существующими системами мониторинга ВУ. Это позволяет обеспечить поэтапное (бесшовный) переход на технологию озеро данных, не вмешиваясь в устоявшийся режим работы служб эксплуатации ВУ. Более подробно предлагаемые авторами этапы перехода будут рассмотрены дальше в статье.

При использовании “озер данных” основной опасностью является превращение “озера” в “болото” данных [18]. Наполнение озера данных происходит из нескольких источников, которые изменяются в течение времени эксплуатации ВУ (форматы файлов событий могут меняться в связи с обновлениями версий ПО, переходом на новые программные продукты или модернизацией аппаратной составляющей ВУ). В результате достаточно легко оказаться в ситуации, когда мы не знаем, что же именно за прошедшие годы эксплуатации накопилось в “озере”, и не имеем технической возможности разобраться в этом.

Т.к. поначалу информация, размещаемая в “озере” – хорошо структурированные файлы событий от уже используемых систем мониторинга ВУ, это облегчает задачу идентификации накопленных данных на первых этапах его функционирования. Чтобы избежать путаницы в дальнейшем, предлагается реализовать распознавание и хранение метаинформации об источниках данных, помещаемых в “озеро”, с учетом изложенного в работах [19] и [20] опыта.

В работах [21] 2011 года и [22] 2014 года показывается, что анализ производительности всех задач сопряжен с существенным повышением требований к производительности и удорожанию системы хранения данных. В качестве решения проблемы предлагается переход на системы хранения на дисках SSD. В следующем разделе мы покажем, что за минувшее время этот переход фактически состоялся.

6. Изменения в подходах к построению СХД СКЦ

Основой промышленной системы хранения данных уже многие десятилетия является накопитель на жестких магнитных дисках (НЖМД, HDD), и относительно недавно (немногим более десяти лет назад) появившийся т.н. накопитель на твердотельной памяти (SSD).

В таблице 3 представлены параметры максимальной емкости, средней стоимости устройства и единицы хранения на дисках SATA, SAS и SSD за последние пятнадцать лет с пятилетним интервалом. Для построения таблицы использовались усредненные данные прайс-листов интернет-магазинов компьютерных комплектующих nix.ru, oldi.ru и xcom-shop.ru, доступные в интернет-архиве archive.org. Рассматривались получившие наибольшее распространение в промышленных СХД модели дисков SATA с частотой вращения 7200 мин-1 и SAS - с частотой вращения 10000 мин-1. Данные по дискам SSD в таблице присутствуют начиная с 2009 года, т.к. эти диски появились в продаже с 2006 года. В 2004 году вместо данных по дискам SAS 10k использовались данные по дискам с интерфейсом Ultra320 LVD SCSI (предшественника SAS).

Таблица 3. Максимальная емкость, средняя стоимость устройства и единицы хранения данных для накопителей на жестких магнитных дисках и твердотельной памяти, применяющихся в промышленных СХД

Устройство хранения данных	**max емкость устройства, Гб**	**Стоимость устройства, USD**	**Средняя стоимость единицы хранения данных, USD/Гб**
2004 год			
НЖМД SAS 10k	146.8	500-540	3,54
НЖМД SATA 7.2k	400	260-280	0,65
SSD	-	-	-
2009 год			
НЖМД SAS 10k	600	450-480	0,77
НЖМД SATA 7.2k	2000	430-450	0,22
SSD	600	1400-1600	2,50
2014 год			
НЖМД	1200	470-530	0,42

SAS 10k			
НЖМД SATA 7.2k	4000	250-285	0,07
SSD	2000	5500-5700	2,80
2019 год			
НЖМД SAS 10k	2400	420-470	0,18
НЖМД SATA 7.2k	14000	440-500	0,03
SSD	7680	960-2330	0,21

На основании представленных в таблице 2 данных можно сделать следующие выводы:

- основные тенденции развития для всех видов накопителей – повышение плотности хранения данных и уменьшение стоимости единицы хранения данных;

- НЖМД SATA 7.2k с момента появления обладают самой низкой стоимостью единицы хранения данных, наибольшей скоростью роста плотности записи и наибольшей скоростью удешевления стоимости за единицу хранения данных;

- SSD по скорости роста плотности записи и скорости удешевления стоимости единицы хранения, занимая второе место после НЖМД SATA 7.2k, за последние пять лет значительно обогнали НЖМД SAS 10k, при этом по стоимости единицы хранения данных сравнялись с ними и вытеснили их с рынка быстродействующих устройств хранения;

- для построения СХД большой емкости используются НЖМД SATA 7.2k, для быстродействующих СХД – SSD.

Следует отметить, что в последние 2 года твердотельные накопители SSD с интерфейсом передачи данных SCSI (SAS или SATA) уступают место накопителям с интерфейсом NVMe, что существенно повлияло на их производительность, но не на исследуемые в таблице 2 параметры, т.к. в обоих случаях используются аналогичные микросхемы NAND памяти.

В отличие от параметров плотности записи и стоимости единицы хранения, значительный прогресс в которых за рассматриваемый в таблице 2 период несомненен, с быстродействием НЖМД все обстоит не так радужно

Быстродействие НЖМД, измеряемое в операциях ввода-вывода данных в секунду (input/output operations per second, IOPS) обратно пропорционально времени доступа к данным, состоящим из времени позиционирования блока считывающих головок диска над дорожкой с магнитной записью (seek time), задержки вращения участка магнитной дорожки с необходимыми данными (rotational time) и времени записи/считывания данных (transfer time). От плотности записи данных зависит лишь последний параметр, вносящий наименьшую задержку. Время позиционирования блока магнитных головок ограничено инерцией, задержка вращения - скоростью вращения пакета пластин из магнитного материала, технологический предел в отношении обоих параметров был достигнут более пятнадцати лет назад. В результате на протяжении исследуемого периода быстродействие НЖМД является практически константой, составляя 75-100 IOPS для НЖМД SATA 7.2k и 125-150 IOPS для НЖМД SAS 10k. Быстродействие твердотельных накопителей с момента их появления выросло от 5-8 тыс. IOPS для первых устройств с интерфейсом SATA-1, 300-330 тыс. IOPS для устройств с интерфейсом NVMe и 500 тыс. IOPS для устройств с интерфейсом PCIe.

Характерной особенностью высокопроизводительных вычислений (HPC) является массированный параллельный произвольный доступ к данным [23]. До появления и заметного удешевления SSD для обеспечения требуемой производительности СХД использовали крупные массивы самых быстродействующих на тот момент времени дисков – SCSI, а затем SAS. Интерфейс команд SCSI после появления SSD и последующего быстрого роста их быстродействия стал узким местом, появились твердотельные накопители с интерфейсом PCI Express, появился и развивается стандарт NVMe. В архитектуре СХД появился быстродействующий энергонезависимый кэш большой емкости и автоматическое перемещение данных между устройствами с разным быстродействием в зависимости от вида нагрузки (tearing) [23]. Еще одним изменением в архитектуре СХД стало использование твердотельный кэш-памяти не только для компенсации пиков нагрузки на чтение, но и для накопления и упорядочивания данных (сериализации данных) перед записью их крупными блоками на НЖМД. В результате вышеприведенных изменений архитектуры СХД (кэш чтения и записи, сериализация записи, tearing) среднее время доступа диска стало не критичным, критичным параметром становится скорость потоковой записи на диск (этим условиям дешевые диски SATA

удовлетворяют). Дорогие быстродействующие диски SAS в СХД оказались не нужны.

Если сопоставить скорость роста плотности записи и, как следствие, емкости дисков SATA 7.2k со скоростью падения стоимости единицы хранения данных, то можно заметить что каждое новое поколение дисков SATA 7.2k практически не меняет свою стоимость либо становится дешевле при соответствующем увеличении емкости. С выходом очередного поколения дисков SATA 7.2k производство дисков меньшей емкости предыдущих поколений становится невыгодным, их складские запасы распродаются по сниженным ценам, а производственные линии освобождаются под новые модели. Т.о. непрерывный рост емкости диска, а не его быстродействия становится одним из основных маркетинговых факторов. Эту же тенденцию наследуют СХД, основным компонентом которых являются НЖМД.

Возникла парадоксальная ситуация – в настоящее время (а с учетом основных тенденций в развитии устройств хранения и особенностей маркетинга дисков SATA 7.2k – и в будущем) построение СХД для СКЦ с быстродействующим уровнем хранения на SSD или накопителях с интерфейсом NVMe приведет к избытку производительности, а основным слоем хранения – на дисках SATA 7.2k приведет к избытку дискового пространства этой СХД, превышающему нужды СКЦ.

Мы полагаем, что эта тенденция избытка производительности и дискового пространства СХД СКЦ решает проблему долговременного хранения как неструктурированных, так и обработанных данных мониторинга в сети СКЦ в локальных “озерах данных”.

7. Выбор ПО для организации сбора и обработки данных мониторинга

Для сбора и обработки больших объемов данных мониторинга существует ряд достаточно хорошо зарекомендовавших себя решений - Loggly, Splunk, Logentries и другие. К сожалению, эти программные продукты распространяются на платной основе, имеют закрытый программный код, поэтому нам не подходят. Среди свободно распространяемых решений с открытым кодом наиболее популярными являются Graylog и стек программных продуктов ELK (Elasticsearch, Logstash, Kibana). Основным отличием Graylog является функционал, узко специализированный именно под сбор логов. Стек ELK помимо богатого функционала для сбора, обработки и хранения данных с каждым новым релизом дополняется функционалом, который превращает его в поисково-аналитическую платформу, поэтому мы выбрали стек ELK в качестве основного инструмента для реализации системы мониторинга сети СКЦ.

8. Архитектура системы мониторинга сети СКЦ

Уровень сбора данных, из “озера” и, при необходимости, из других источников организуется с помощью специальных процессов-агентов. В их роли могут выступать агент ПО мониторинга (Zabbix,, Nagios, Ganglia, Icinga, Cacti и др.), сервисы Beats, Fluentd, NXlog, Syslog и пр.

Уровень обработки данных реализован в Logstash, который поддерживает множество входных типов данных – это могут быть журналы, метрики разных сервисов и служб. Имеющийся в Logstash функционал позволяет структурировать, фильтровать, анализировать, идентифицировать и интегрировать собираемые данные (например связывать их с геокоординатами, IP-адресами), упрощая тем самым последующий анализ. В настоящее время для Logstash разработано большое количество библиотек, позволяющих упростить процедуру получения данных (предопределены правила разбора журнала событий) Filebeat. Кроме этого, также доступны Winlogbeat (события Windows Event Logs), Metricbeat (метрики), Packetbeat (сетевая информация) и Heartbeat (uptime), также имеются библиотеки позволяющие настроить вывод информации практически в любой источник, а не только в Elasticsearch.

На уровне хранения собранные данные помещаются в Elasticsearch, являющийся основой стека ELK. Основные функции Elasticsearch: индексирование и поиск. Elasticsearch поддерживает работу в режиме кластера отказоустойчивости и кластера распределения нагрузки.

Для представления данных, индексированных в Elasticsearch, используется веб-интерфейс Kibana. Результатом представления данных может быть не только текстовая или табличная информация, но и диаграммы или графики. Функционал Kibana позволяет визуализировать данные и строить

табличные отчеты. Для построения дашбордов инженеры используют дополнительный инструмент - Grafana.

Для формирования реакций на события используется функционал Watcher пакета X-Pack, программный код которого был предоставлен разработчиками в открытый доступ в 2018 году.

Путем управления настройками серверов хранения распределенного кластера Elasticsearch размещение данных мониторинга СКЦ ограничивается узлами кластера, размещенными в СКЦ. Это позволяет при реализации функций системы мониторинга для СКЦ локализовать обмен служебными данными внутри сети этого СКЦ. При обработке данных мониторинга сети СКЦ задействуются ресурсы всего распределенного кластера Elasticsearch.

После развертывания кластера Elasticsearch формируются собственные представления данных для служб эксплуатации каждого СКЦ и для службы централизованного мониторинга сети СКЦ. Пользователь вычислительными ресурсами получает доступ к унифицированному интерфейсу доступа к данным мониторинга на любом СКЦ - т.е. пользователь имеет возможность получить нужное ему представление данных при решении проблем мониторинга вычислительных заданий.

9. Развертывание системы мониторинга сети СКЦ

На первом этапе в СКЦ организуется “озеро данных”, куда организуется транспортировка данных от уже имеющихся систем мониторинга, как наиболее ценных и структурированных для дальнейшей обработки, а также неструктурированные данные мониторинга.

На втором этапе в СКЦ развертывается кластер серверов Logstash и части распределенного кластера Elasticsearch. Количество серверов Logstash зависит от количества оборудования ВУ и объема обрабатываемых данных. Конфигурация серверов хранения данных, индексов и поисковых серверов кластера Elasticsearch также зависит от количества эксплуатируемого оборудования и объема данных уже имеющихся систем мониторинга. В качестве серверов кластера может использоваться как физическая инфраструктура, так и виртуальная на базе услуг виртуализации, предлагаемых СКЦ. Второй вариант предпочтительнее, т.к. позволяет использовать функционал развертываемой системы мониторинга для динамического управления ее вычислительными ресурсами.

В развернутой системе мониторинга описываются правила выделения нужных метрик из входного потока неструктурированных данных, формируются представления отфильтрованного потока событий инженерам службы эксплуатации и производится настройка реакций системы – т.о. выполняется перенос функций эксплуатируемых в СКЦ систем мониторинга на внедряемую систему.

На третьем этапе производится построение агрегированной статистики по сети СКЦ на базе собираемых после первых двух этапов данных, формируется представления отфильтрованного потока событий для службы централизованного мониторинга сети СКЦ и, если администрация СКЦ принимает такое решение - отказ от старых систем мониторинга. При этом сохраняется возможность старые системы мониторинга СКЦ параллельно с внедряемой системой централизованного мониторинга сети СКЦ.

На последнем этапе развертывания системы мониторинга сети СКЦ производится доработка прогнозирования и реакций на события на глобальном уровне, при необходимости - выделение новых метрик из потока неструктурированных данных.

10. Достоинства и недостатки предложенной архитектуры

Во время эксплуатации прототипа мы успели для себя определить некоторые положительные и отрицательные стороны системы мониторинга на основе стека ELK.

Достоинства:

- единая экосистема хорошо интегрированных друг с другом продуктов, в которой есть практически весь необходимый для системы мониторинга функционал;
- Logstash представляет собой очень гибкий и мощный препроцессор, имеющий широкий функционал обработки неструктурированных данных, легко расширяемый при необходимости;
- Elasticsearch поддерживает SQL в качестве языка запросов, что упрощает его интеграцию с другими компонентами системы мониторинга СКЦ и облегчает освоение нового инструмента персоналом службы эксплуатации;
- качественная документация, которая позволяет быстро вводить новых инженеров

в проект;

- отсутствие необходимости заранее знать о структуре получаемых данных до начала их сбора: можно начать агрегировать события как есть, а затем, по мере появления понимания, какую полезную информацию можно из них извлечь, менять подход к их обработке, не теряя «обратную совместимость».

К недостаткам стека ELK, как и любого активно разрабатываемого программного продукта, можно отнести достаточно частые релизы, в которых нередко нарушена обратная совместимость. Это приводит к необходимости иметь достаточно квалифицированную службу эксплуатации, виртуальный стенд для тестирования процедуры обновления, или оставаться на старой стабильной версии.

10. Выводы

Развитие суперкомпьютерных центров привело к постоянному росту количества систем мониторинга вычислительных установок суперкомпьютерных центров. По мере роста сложности и производительности вычислительных установок появляется ряд научных задач, требующих построения профиля вычислительного задания путем объединения данных о работе различных систем оборудования и программного обеспечения вычислительной установки, причем зачастую не все из них охватываются имеющимися системами мониторинга. Объединение СКЦ в сеть еще больше усложняет решение этих задач.

В качестве способа решения поставленной проблемы в статье предложен способ сохранения как структурированных данных систем мониторинга вычислительных установок, так и неструктурированных первичных данных о работе оборудования и программного обеспечения вычислительных установок суперкомпьютерных центров в “озерах данных”, размещаемых внутри суперкомпьютерных центров. Анализ тенденции развития систем хранения данных показал что в системе хранения данных суперкомпьютерного центра неизбежно образуется избыток дискового пространства, достаточного для создания “озер данных”. Разработанная архитектура системы мониторинга сети суперкомпьютерных центров на основе стека программного обеспечения ELK (Elasticsearch, Logstash, Kibana) позволяет интегрировать данные существующих локальных служб мониторинга вычислительных установок суперкомпьютерного центра в глобальную распределенную структуру, позволяя при необходимости использовать также и не обработанные ранее первичные данные. Распределенный кластер серверов ELK позволяет обеспечить отказоустойчивость работы, локальность хранения данных мониторинга вычислительной установки, специализированные интерфейсы для служб эксплуатации суперкомпьютерного центра, пользователей вычислительными ресурсами и службы глобального мониторинга в сети суперкомпьютерных центров.

Дальнейшие исследования планируется сосредоточить на использовании средств машинного обучения и искусственного интеллекта для прогнозирования локальных относительно отдельного суперкомпьютерного центра и глобальных событий сети суперкомпьютерных центров, автоматизации формирования реакций на эти события и реализации элементов проактивного мониторинга.

Работа выполнена в МСЦ РАН – филиале ФГУ ФНЦ НИИСИ РАН в рамках Государственного задания. В исследованиях использовался суперкомпьютер МВС-10П.

One approach to build a monitoring system for a supercomputer centers network

S.A. Leshchev, A.I. Tikhomirov

Abstract: The article considers the problem of the global monitoring organizing in the federation of supercomputer centers. The article analyzes the world experience in organizing monitoring in supercomputer centers and grid-systems. It is shown that in order to build a complete profile of the computational job in the federation of supercomputer centers, insufficiently to process data of existing and promising monitoring systems. A method is proposed for storing complete data on the operation of the equipment and software of supercomputer center in “data lakes” on the supercomputer centers resources, the sufficiency of these resources is substantiated. The architecture of a global monitoring system based on the ELK software stack (Elasticsearch, Logstash, Kibana) is proposed, its advantages and disadvantages are considered, a plan for the transition to a new monitoring system is developed.

Keywords: HPC, supercomputer, computing federation, monitoring, job-centric monitoring, data storage, data lake, Elasticsearch, ELK, ETL

Литература

1. Б.М. Шабанов, А.П. Овсянников, А.В. Баранов, С.А. Лещев, Б.В. Долгов, Д.Ю. Дербышев. Проект распределенной сети суперкомпьютерных центров коллективного пользования. «Программные системы: теория и приложения», (2017), № 4(35), 245–262. DOI:10.25209/2079-3316-2017-8-4-245-262

2. А.В. Баранов, А.И. Тихомиров. Методы и средства организации глобальной очереди заданий в территориально распределенной вычислительной системе. «Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика», Т.6 (2017), № 4, 28–42. DOI:10.14529/cmse170403

3. А.В. Баранов, А.П. Овсянников, Б.М. Шабанов. Федеративная аутентификация в распределенной инфраструктуре суперкомпьютерных центров. «Труды научно-исследовательского института системных исследований Российской академии наук», Т.8 (2018), № 6, 79–83. DOI:10.25682/NIISI.2018.6.0011

4. A.V. Baranov, S.A. Leshchev. Methods and means of distributed storage systems implementation. “Software Journal: Theory and Applications” V.3 (2019), 14-24. DOI:10.15827/2311-6749.19.3.3

5. А.В. Баранов, А.С. Шитик. Способы и средства динамической реконфигурации сетей суперкомпьютера при представлении пользовательских заданий в виде контейнеров. «Программные продукты, системы и алгоритмы», (2018), № 3, 62–70. DOI:10.15827/2311-6749.18.3.8

6. Г.И. Савин, П.Н. Телегин, А.В. Баранов, А.С. Шитик. Способы и средства представления пользовательских суперкомпьютерных заданий в виде контейнеров Docker. «Труды научно-исследовательского института системных исследований Российской академии наук», Т. 8 (2018), № 6, 84–93. DOI:10.25682/NIISI.2018.6.0012

7. А. В. Баранов, Д. С. Николаев. Использование контейнерной виртуализации в организации высокопроизводительных вычислений. «Программные системы: теория и приложения», Т.7 (2016), №1, 117–134. DOI: 10.25209/2079-3316-2016-7-1-117-134

8. Zenoss Community – Open Source Network Monitoring and Systems Management. http://www.zenoss.org

9. R. Wolski. Forecasting network performance to support dynamic scheduling using the network weather service. “Proceedings. The Sixth IEEE International Symposium on High Performance Distributed Computing”, Portland, OR, USA, 1997, 316-325. DOI:10.1109/HPDC.1997.626437

10. R. Wolski, N.T. Spring, J. Hayes. The network weather service: a distributed resource performance forecasting service for metacomputing. “Future Generation Computer Systems”, V. 15, I. 5–6 (1999), 757-768. DOI:10.1016/S0167-739X(99)00025-4.

О направлениях развития зарубежных национальных научно-образовательных сетей

Гончар А.А.[1], Данилов С.А.[2], Овсянников А.П.[3]

Межведомственный суперкомпьютерный центр РАН – филиал ФГУ ФНЦ НИИСИ РАН,
E-mails: [1]andrey.gonchar@jscc.ru, [2]sdanilov@jscc.ru, [3]ovsyannikov@jscc.ru

Аннотация. В статье рассматривается тенденции функционирования и развития национальных научно-образовательных сетей в последние годы. Актуальность исследования обусловлена необходимостью планирования развития Национальной исследовательской компьютерной сети. Отмечено, что несмотря на возрастающую конкуренцию со стороны коммерческих провайдеров интернет и цифровых сервисов, национальные сети науки и образования сохраняют свое значение и обеспечивают уникальные услуги для своих пользователей. Указаны некоторые изменения функциях научно-образовательных сетей. Сделан вывод о сохранении и укреплении роли национальных сетей науки и образования в проведении научных исследований.

Ключевые слова: национальные сети науки и образования, научно-образовательные сети, NREN

Введение

Телекоммуникации и цифровые технологии играют важную роль для проведения современных научных исследований и обеспечения образовательной деятельности. С момента возникновения сети Интернет он рассматривался научно-образовательным сообществом как платформа для интеграции научных информационных ресурсов и открытая коммуникационная среда для неограниченного обмена информацией [1]. Результатом стало повсеместное создание и развитие национальных научно-образовательных сетей (National Reserach and Educational Networks - NREN) как средства ускорения и поддержки научных исследований, сотрудничества и коммуникаций между учеными с одной стороны, и экономически эффективного распределения и совместного использования дорогих вычислительных ресурсов с другой.

Как правило NREN - это высокоскоростные телекоммуникационные сети, уникальные в каждой стране и независимые от коммерческого интернета. Во многих странах они являются основой инфраструктуры для развития образования, обеспечивая надежный, эффективный и экономичный информационный обмен, доступ к компьютерным ресурсам и их совместное использование. NREN обеспечивают глобальную научную кооперацию, например, участие исследователей в таких международных научных проектах, как Большой адронный коллайдер, PRACE, eVLBI.

Каждая страна определяет способы организации, функционирования своих научно-образовательных сетей и направления их развития. В данной статье предпринята попытка обобщения зарубежного опыта последних пяти-восьми лет для того, чтобы прояснить тенденции и определить возможные направления развития NREN. Последнее актуально в связи с созданием в 2019 году Национальной исследовательской компьютерной сети (НИКС) [2] на основе интеграции федеральной университетской сетей RUNNet [3] и академической сети RASNet [4] и необходимостью планирования ее развития.

Национальные научно-образовательные сети

Обеспечение конкурентоспособности научных исследований на мировом уровне требует новых информационно-телекоммуникационных услуг, которые еще не успели появиться на коммерческом рынке. Эти передовые услуги разрабатываются и предоставляются национальными сетями науки и образования, которые будучи сами частью научно-образовательного сообщества, способны наиболее ясно понять возникающие потребности научных исследований и немедленно на них отреагировать.

Научно-образовательные сети появились одновременно с появлением Интернет, когда предложений коммерческих операторов связи было недостаточно. Научное сообщество пользователей NREN приобрело большой опыт в разработке

коммуникационных протоколов и поиске инновационных способов использования сетей для науки и образования. Согласно [5], несмотря на развитие сетевых и телекоммуникационных технологий промышленностью, значительная часть современных интернет-инноваций по-прежнему осуществляется в рамках образовательного и исследовательского сообщества, так как коммерческие интернет-провайдеры не имеют достаточной мотивации для достижения требуемого ему уровня инноваций. Подчеркивается, что некоторые продвинутые интернет-провайдеры могут предлагать некоторые основные услуги NREN по цене, которая ниже, чем у NREN, однако при этом может не обеспечиваться полный спектр услуг, необходимых сообществу. Кроме того, использование таких коммерческих услуг ведет к ослаблению инноваций, осуществляемых под руководством научно-образовательного сообщества. Научно-образовательные сети по-прежнему могут предоставлять услуги, которые, оказываются более удобными или более экономичными, чем использование внешних услуг коммерческих организаций (например, Google). Утверждается, что пользователи NREN ценят предоставляемые в рамках NREN передовые сетевые сервисы и повышенное качество обслуживания и готовы платить за них.

Формально NREN предоставляет услуги связи научно-образовательным организациям, однако главная задача заключается в создании условий для повышения качества исследований и образования.

Помимо обеспечения надежной сетевой связности в задачи NREN входят:

- идентификация, разработка и распространение сетевых сервисов, необходимых научно-образовательному сообществу;
- анализ и внедрение новых сетевых технологий;
- обучение, помощь и поддержка пользователей;
- участие в международных сетевых организациях, консорциумах NREN;
- передача новых сервисов и передовых сетевых технологий организациям и предприятиям;
- вклад в разработку и осуществление национальных стратегий распространения информационных технологий.

Собственно именно инновационная функция NREN, возможность использования сети в качестве тестового стенда для новых, пусть еще не достаточно опробованных и отлаженных технологий, и отличает NREN от коммерческих провайдеров.

В 2019 году на момент написания статьи у авторов имелись сведения о функционировании научно-образовательных сетей национального масштаба в 142 странах. Для сравнения в 2013 году таких стран было 137, в 2009 — 121, в 2005 — 98 [6].

Таким образом, несмотря на ускорение внедрения новых технологий и услуг у коммерческих провайдеров, роста цифровизации, снижения позиций национальных научно-образовательных сетей не наблюдается.

Следует отметить, что термин «национальная научно-образовательная сеть» в контексте данной статьи и исследования [6] не подразумевает, что в стране имеется единственная сеть, наделенная статусом национальной, а является указанием на масштаб ее деятельности. В ряде стран имеется несколько сетей, имеющих «национальный» (или «федеральный») масштаб, например в США такими сетями являются Internet2 и EsNet, в Китае CSTNET, CERNET. Наличие NREN в стране рассматривается как фактор, определяющий возможности страны для участия в национальных и международных научных исследованиях [6].

Для эффективной реализации международных проектов и взаимной кооперации NREN объединяются в наднациональные объединения — консорциумы научно-образовательных сетей: NORDUnet [7] (Скандинавские страны), GÉANT [8] (Европа), Asi@Connect/TEIN [9] и APAN [10] (Азия), RedClara [11] (Южная Америка), AfricaConnect [12].

NREN обычно имеет иерархическую структуру (рис. 1).

На верхнем уровне находится национальная опорная сеть, к которой подключаются организации, допустимые правилами использования сети: организации науки и высшего образования, возможно правительственные учреждения, школы и организации здравоохранения. Возможно как непосредственное подключение таких организаций к опорной сети NREN, так и подключение через региональные или проблемно-ориентированные сети науки и образования.

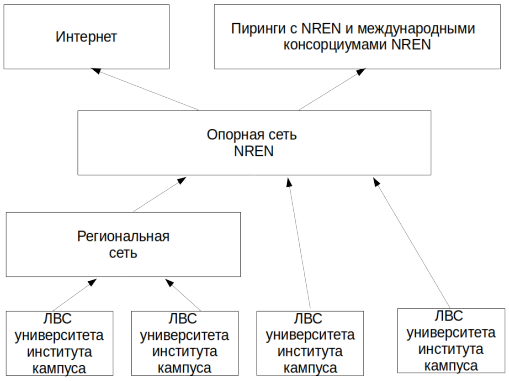


Рис. 1. Иерархическая структура NREN

Иногда под термином NREN подразумевают совокупность опорной сети, подключенных к ней региональных сетей и организаций.

Финансирование NREN

Национальные научно-образовательные сети разнообразны по своему организационному устройству, правовому статусу, источникам финансирования.

NREN могут быть правительственными учреждениями и некоммерческими организациями, например, отдельными юридическими лицами или органами (подразделениями) в составе государственных учреждений. Даже в случае отсутствии прямых связей с правительством и функционировании NREN как коллегиального органа, следует учитывать, что организации науки и образования, которые представляет этот орган, напрямую финансируются государством.

Основными источниками финансирование NREN являются прямые государственные субсидии, средства, получаемые в рамках международных проектов (например, GEANT) или за счет взносов пользователей — организаций науки и образования.

Следует отметить, что прямая государственная поддержка оказывается более 80% NREN, источники финансирования которых известны. Около 70% NREN получают средства от организаций-пользователей науки и образования и более 25% от предоставления коммерческих услуг связи [13-17].

Отметим также, что несмотря на колебания финансирования NREN тенденция увеличения бюджета отмечается более чем у 50% NREN (с известным объемом финансирования) [13-17] (см. рис. 2).

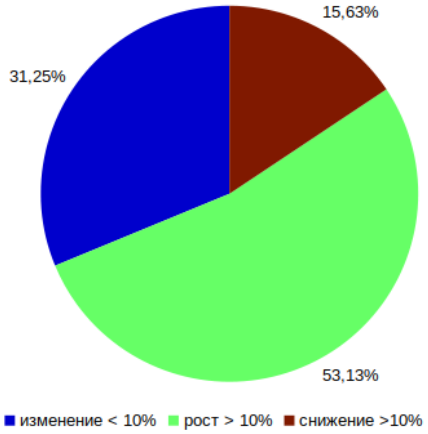


Рис. 2. Распределение доли NREN по изменению бюджета (2016-18 гг. в сравнении с 2014-16 гг.)

Таким образом, можно говорить о преобладающей стабильности и тенденции роста финансирования национальных научно-образовательных сетей, что означает, что роль национальных научно-образовательных сетей в последние годы не снижается, несмотря на достижения коммерческого IT-сектора.

Функции NREN и развитие сервисов

Еще в [18] отмечалось, что национальные научно-образовательные сети должны сконцентрироваться на пользователе и интегрировать в себе функции:

- глобального сервис-провайдера (глобальная и локальная сетевая связность, открытые точки обмена, виртуализация, мобильность, сенсорные сети, доступ к зарубежным ресурсам);
- предоставления услуг сообществу пользователей (аутентификация пользователей научно-образовательных сетей, услуги для кампусов, видеоконференцсвязь, средства совместной работы ориентированные на веб, сервисы доставки и распределения контента, облачные сервисы, социальные сети);
- поддержки научных исследований (средства и инструменты электронной науки и организации научных исследований);
- поддержки электронного образования (средства и инструменты электронного образования, в т.ч. производства контента и его распространения с контролем авторских прав, виртуальные классы);
- предоставления инновационных платформ (цифровых платформ для

разработки новых технологий; разработка, внедрение новых сетевых технологий, которые могут быть использованы в научных исследованиях и обучение им пользователей NREN; обеспечения механизма разработки сервисов на основе федеративных ресурсов; обеспечение сохранения прав на интеллектуальную собственность).

В 2018 году пользователям европейских NREN были доступны 74 различных сервиса [13] в 6 основных категориях (см. рис.3).

Коммерческие операторы связи составляют все большую конкуренцию национальным научно-образовательным сетям в сетевых сервисах, в том числе выделения высокоскоростных каналов связи, каналов на основе волнового уплотнения, применению протокола IPv6 и оказание услуг связи с заданным уровнем сервиса. Однако предоставляемый национальными научно-образовательными сетями уровень поддержки конечного пользователя неизмеримо выше и включает, например, доступ к средствам NetFlow, Perfsonar, базы знаний для поиска и устранения сетевых проблем (eduPERT).

Рис. 3. Сетевые сервисы европейских NREN по категориям

Значительным преимуществом предоставляемых NREN сетевых сервисов является возможность конечным пользователю проследить весь путь их взаимодействия через научно-образовательные сети или их консорциумы (см. рис. 4)

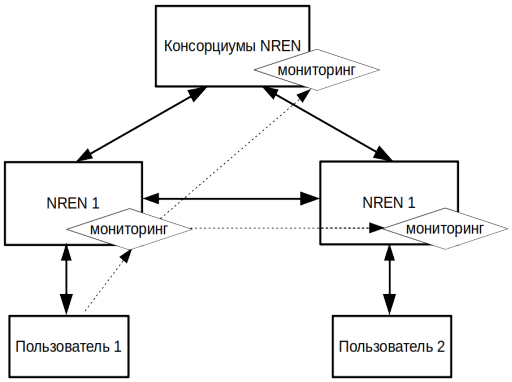


Рис. 4. Взаимодействие NREN при предоставлении сетевых сервисов

Научно-образовательные сети как и раньше оказываются впереди в развитии инновационных сервисов. Так на 2018 год 23 европейских NREN применяют или внедряют для своих пользователей передовые технологии SDN (программно-определяемые сети) и 14 NREN - NFV (виртуализация сетевых функций) [13].

Важнейшей частью сервисов NREN являются сервисы безопасности и идентификации. Их централизованная реализация на инфраструктуре научно-образовательной сети позволяет выделить эту сеть из Интернет, организовать доверенную среду для пользователей и снизить расходы организаций-пользователей на информационную безопасность. К указанным централизованным сервисам относятся, например, ослабление DDoS-атак, обнаружение вторжений, защитные экраны, фильтрация трафика, обнаружение уязвимостей, центры реагирования на компьютерные инциденты. Такие сервисы достаточно часто оказываются доступны и пользователям коммерческих Центров обработки данных (ЦОД), но безопасность на границе ЦОД и заканчивается, NREN же охватывает все научно-образовательное сообщество страны. Сервисы идентификации, позволяющие создать доверенную среду: удостоверяющие федерации, eduroam - уникальны для научно-образовательного сообщества и работают не только на национальном, но и международном уровне.

Рост спроса на облачные сервисы в рамках научных исследований в последние годы определяет существенную роль NREN в их предоставлении пользователям. В [13] рассмотрены четыре основных роли, которые при этом могут играть национальные научно-образовательные сети.

1. Центр компетенции. NREN может использовать полученные знания и опыт для консультаций организаций-пользователей по внедрению облачных технологий. Подход ускоряет и облегчает внедрение облачных технологий в научные исследования.

2. Агрегатор. С использованием механизмов, организованных на национальном уровне или международном уровне (например, в консорциуме GÉANT) NREN могут использовать совокупную покупательную способность своих пользователей для снижения расходов путем заключения и поощрения рамочных соглашений о закупках. В дальнейшем организации-пользователи заключают контракты непосредственно с поставщиком облачных услуг, но по специальной цене.

3. Реселлер. Там, где это разрешено политикой NREN, она может выступать в качестве реселлера услуг, принимая на себя договорные отношения и выступая в качестве посредника для пользователей и перепродавая услуги. Преимуществами подхода является простота и определенность для поставщика, что обеспечивает лучшие условия, но он может повлечь более высокие риски для NREN.

4.Интегратор. В рамках моделей SaaS NREN могут потенциально принимать услуги более низкого уровня и развивать на их основе свои сервисы. Этот подход предъявляет дополнительные финансовые и кадровые требования для NREN, что ограничивает его привлекательность.

Заключение

Несмотря на возрастающую конкуренцию со стороны коммерческих провайдеров интернет и цифровых сервисов, сохраняется ряд сервисов, требуемых научно-образовательным сообществом, которые может обеспечить только NREN. Сохраняются доминирующие возможности NREN по обеспечению информационной безопасности, опережающему развитию, внедрению и распространению инновационных технологий и сервисов для научно-образовательного сообщества. Сохраняются функции NREN как агрегатора средств и ресурсов. Вместе с тем прослеживается постепенное смещение функций национальной научно-образовательной сети от предоставления технологий в сторону экспертной и посреднической активности.

Прослеживаемые тенденции финансирования национальных научно-образовательных сетей также свидетельствуют о сохранении и укреплении их роли в проведении научных исследований.

Статья подготовлена в рамках государственного задания ФГУ ФНЦ НИИСИ РАН на 2019 год (тема № 0065-2019-0014).

About directions of development of foreign national research and educational networks

Gonchar A.A., Danilov S.A., Ovsyannikov A.P.

Abstract. The article considers the trends of functioning and development of national research and educational networks (NREN) in recent years. The relevance of the study is due to the need to plan the development of the National research computer network in Russia. It is noted that despite increasing competition from commercial providers Internet and digital services, NREN retain their importance and provide unique services for their users. Some changes in the functions of NREN are indicated. It is concluded that the role of NREN in research is preserved and strengthened.

Keywords: National Research and Educational Network, NREN

Литература

1. Кулагин М.В., Серебряков В.А. Информационное пространство РАН (Проекты и реализация, 1998-2013) // Научный сервис в сети Интернет: труды XVIII Всероссийской научной конференции (19-24 сентября 2016г., г.Новороссийск). — М.: ИПМ им. М.В.Келдыша, 2016. — С. 194-222. — doi:10.20948/abrau-2016-40

2. Г.И. Савин, Б.М. Шабанов, А.В. Баранов, А.А. Гончар, А.П. Овсянников. Об использовании федеральной научной телекоммуникационной инфраструктуры для суперкомпьютерных вычислений // Суперкомпьютерные дни в России: Труды международной конференции. 23-24 сентября 2019 г., г. Москва / Под. ред. Вл.В. Воеводина – М.: МАКС Пресс, 2019. С. 101-112.
3. Абрамов А.Г., Евсеев А.В. RUNNET как научно-образоваельная сеть России: цели, основные задачи, телекоммуникационная инфраструктура и сервисы // Информатизация образования и науки, Москва, ЦРГОП и ИТ, 2018, № 4 (40), С. 3-15
4. Боков Д.Ю., Репин Д.С. Состояние и тенденции развития современных научно-образовательных сетей // ИТНОУ: Информационные технологии в науке, образовании и управлении, Москва, 2018, № 2 (6), С. 69-76
5. Dyer J. The case for National Research and Education Networks (NRENs) // Terena. 2009 URL: https://www.terena.org/publications/files/20090127-case-for-nrens.pdf. [дата обращения: 18.11.2019]
6. ITU (2014) Partnership on measuring ICT for development. Final WISIS targets review. Achievements, challenges and the way forward. Geneva, Switzerland
7. NORDUnet. Nordic gateway for Research and Education URL: https://www.nordu.net/ (дата обращения: 18.11.2019).
8. GÉANT // URL: https://www.geant.org/ (дата обращения: 18.11.2019).
9. Asi@Connect // URL: http://www.tein.asia (дата обращения: 18.11.2019).
10. Asia Pacific Advanced Network // URL: https://apan.net/ (дата обращения: 18.11.2019).
11. RedCLARA. Latin American Cooperation of Advanced Networks //
URL: https://www.redclara.net/ (дата обращения: 18.11.2019).
12. AfricaConnect2 // URL: https://www.africaconnect2.net/ (дата обращения: 18.11.2019).
13. GN4-3-19-25587fa. 2018 Compendium Study // GÉANT Association, 2018 URL: https://www.geant.org/Resources/Documents/2018-Compendium-Study.pdf (дата обращения: 18.11.2019)
14. GÉANT Compendium of National Reserach and Educational Networks in Europe 2017 // GÉANT Association, 2017 URL:
https://www.geant.org/Resources/Documents/Compendium%20Layout%20ONLINE.pdf (дата обращения: 18.11.2019)
15. GÉANT Compendium of National Reserach and Educational Networks in Europe 2016 // GÉANT Association, 2016 URL: https://www.geant.org/Resources/Documents/Compendium2016.pdf (дата обращения: 18.11.2019)
16. GÉANT Compendium of National Reserach and Educational Networks in Europe 2015 // GÉANT Association, 2016 URL: https://compendiumdatabase.geant.org/compendium-2015-updated.pdf (дата обращения: 18.11.2019)
17. GÉANT Association Compendium of National Reserach and Educational Networks in Europe 2014 // GÉANT Association, 2014 URL: https://www.terena.org/publications/files/Compendium-2014.pdf (дата обращения: 18.11.2019)
18. The Role of NREN's in 2020 // NORDUnet, 2011, URL:
https://www.nordu.net/sites/default/files/article_attachments/NORDUnet%20NREN%202020%20Inspiration%20Paper-1.pdf (дата обращения: 18.11.2019)

О востребованности интернет-ресурсов в научно-образовательной сети

Ю. Н. Морин[1], Д.В.Вершинин[2]

МСЦ РАН – филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия

E-mail`s: [1]ymorin@jscc.ru, [2]versh@jscc.ru

Аннотация. В статье рассматривается рейтинг популярности категорий интернет-ресурсов в научно-образовательной сети RASNet в зависимости от времени суток и за сутки на примере 9 недели 2019 года. Доминирующую позицию по востребованности занимают поисковые системы, что подчеркивает важность поиска информации и выбора ее источников. Источники видео информации в сети Интернет опережают по востребованности текстовые, что подтверждается прогнозом и динамикой роста трафика в сети Интернет.

Ключевые слова. RASNet, DNS, база сайтов, поисковые машины, мультимедийный портал, новости.

1. Введение

В 2018 году Международный союз электросвязи (МСЭ) [1] (основанный как Международный телеграфный союз в 1865 году, с 1947 года является специализированным учреждением ООН с штаб квартирой в Женеве, Швейцария [2]) опубликовал доклад, что 3,9 миллиарда человек во всем мире подключены к всемирной сети. Большинство пользователей Интернета проживают в развитых странах, где к сети подключено 80 процентов населения. В то же время использование Интернета постепенно растет и в развивающихся странах: с 7,7 процента в 2005 году до 45,3 процента в 2019. Почти 96 процентов населения сегодня имеют доступ к мобильной связи, а из них 90 процентов охвачены доступом на уровне 3G. Это высокий показатель, и благодаря ему мы можем понять, почему теперь 51 процент населения Земли пользуется Интернетом.

К сети RASNet подключены несколько десятков институтов Российской академии наук, ведущих научных организаций страны. Таким образом, целью данной работы является оценка, какие категории сайтов используют ученые во временном разрезе.

2. RASNET

Исследование проводилось на сети RASNet. Это сеть Российской академии наук, которая объединяет региональные сети отделений и научных центров РАН. Сеть RASNet предназначена для организации информационного обмена институтов и подразделений внутри РАН с российскими и зарубежными научно-образовательными сетями и сетью Интернет [3].

Кроме того, в рамках ОТС поддерживается сервис eduroam для аутентификации и авторизации в беспроводных сетях, входящих в мировое научно-образовательное сообщество [4].

3. Domain Name System

Domain Name System (DNS) - это сервис для преобразования доменных имен в IP адреса [5]. В первую очередь этот сервис призван облегчить поиск Интернет ресурсов.

Имя хоста и IP-адрес не тождественны — хост с одним IP может иметь множество доменных имён, что позволяет поддерживать на одном компьютере множество веб-сайтов. Обратное тоже справедливо: одному доменному имени может быть сопоставлено множество IP-адресов – это позволяет создавать балансировку нагрузки. Запрос на определение имени обычно не идёт дальше кэша DNS, который помнит (ограниченное время) ответы на запросы, проходившие через него ранее.

Система DNS – это, по сути, сеть в сети. Если один DNS сервер не знает правильного IP, то он спрашивает у другого и так далее, пока не получит ответ. Распределенная база данных DNS поддерживается с помощью иерархии DNS-серверов, взаимодействующих по определенному протоколу [6, 7].

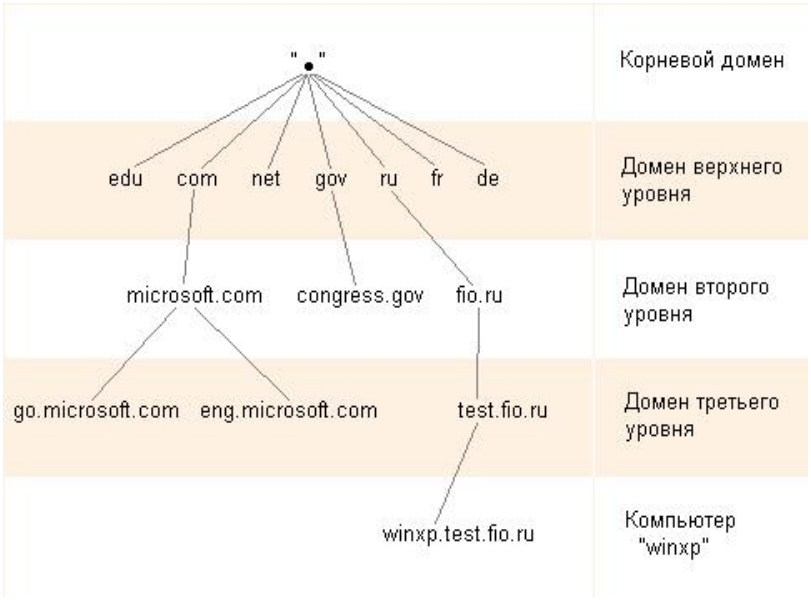


Рис. 1. Пример структуры доменного имени

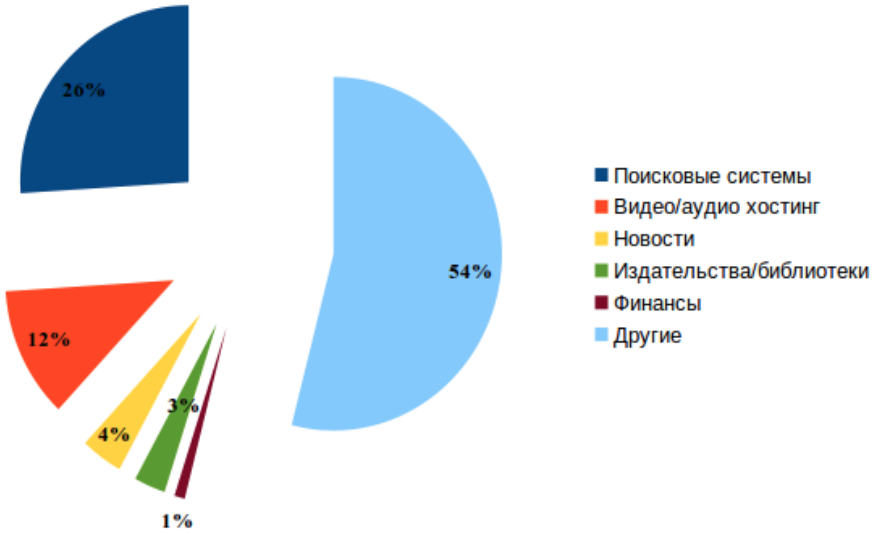


Рис. 2. Популярность категорий в течении суток

4. База фильтрации Интернет сайтов по категориям содержания

База фильтрации Интернет сайтов по категориям содержания - это списки интернет ресурсов (DNS-имена), разделенные по группам в зависимости от их содержания. Один интернет ресурс может быть включен в несколько тематических групп.

Одной из крупных баз является часть коммерческого продукта Cisco Talos [8]. В Cisco Talos эти базы используются для разделения прав доступа к интернет ресурсам корпоративных пользователей и обеспечения анализа угроз всемирной паутины – сети Интернет. Существуют аналогичные международные базы, такие как Shalla Secure Services [9] и The Université Toulouse 1 Capitole [10]. Для исследования были использованы последние две базы данных, содержащие большое количество русскоязычных ресурсов и предоставляемые по свободной лицензии для исследовательских целей.

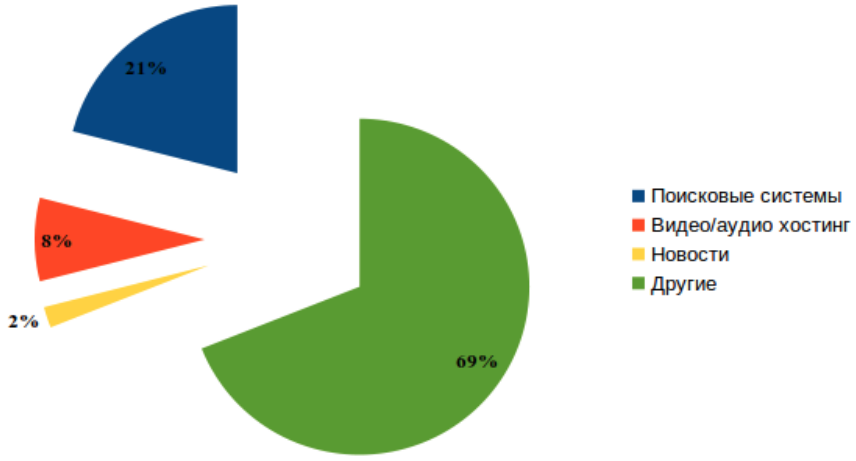


Рис. 3.
Популярность категорий в нерабочее и ночное время

5. Категории сайтов используемые ученными в их профессиональной деятельности

Как уже было отмечено в разделе 2 к сети RASNet подключено множество научных организаций мирового уровня.

Не трудно показать, что в процессе доступа к ресурсам сети Интернет в подавляющем большинстве случаев используется запрос DNS-записи у DNS-сервера. Следует особо подчеркнуть, что анализируя статистику запросов DNS-записей к DNS-серверам сети RASNet, можно оценить какие сайты посещают ученые в рабочее время. Таким образом, используя базы фильтрации Интернет сайтов по категориям содержания, перечисленные в разделе 4, можно произвести категоризацию сайтов используемые российскими ученными.

В течении девятой недели 2019 года была собрана поминутная статистика запросов ресурсов всемирной сети. На основе этих данных была получена диаграмма популярности категорий в течении всего рабочего дня (см. рис. 2). Дополнительно были построены диаграммы по трем временным отрезкам: первая половина рабочего дня (см. рис. 4), вторая половина рабочего дня (см. рис. 5) и ночное время (см. рис. 3).

Стоит отметить, что в категорию другие попали ресурсы, либо не отмеченные в каталогах, приведенных в разделе 4, либо имевшие малое количество запросов в исследуемый период.

Хотелось бы особо оговорить, что статистика анализировалась для доменов не более второго уровня (см. рис. 1). Например, домены третьего уровня вида username.livejournal.com объединялись до домена livejournal.com. Полагается, что в базе данных фильтров не будет записей вида username.livejournal.com, а будет только livejournal.com, относящийся к категории личных интернет-дневников.

Оказалось, что доминирующую позицию в рейтинге занимают поисковые системы. Возможно, такое доминирование связано не только с тем что пользователи сети Интернет ищут информацию, но и с тем, что сайты поисковых систем превратились в большие порталы, на которых размещены другие сервисы помимо поиска.

Вторую строчку заняли ресурсы, относящиеся к площадкам, на которых размещена видео-/аудио- информация, что говорит о важности визуализированной информации в работе. Это подтверждается тем, что средний объем интернет видео, передаваемого в мире по сетям в 2018 году, составит более 98 ПБ в месяц [11], при этом среднегодовой темп роста объема интернет видео в 2016-2021 гг. составляет 31%.

6. Заключение

Установлено, что доминирующую позицию в рейтинге занимают поисковые системы, что указывает на особую важность поиска информации и выбора ее достоверных источников.

Прослеживается тенденция вытеснения текстовых коммуникаций другими видами: изобразительными, аудио, видео. Объем видеоконтента постоянно возрастает в образовании, средствах массовой

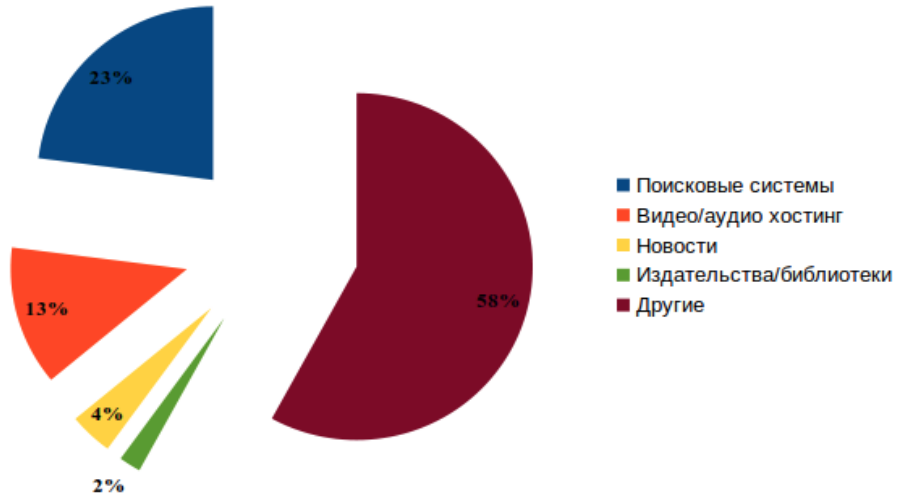


Рис. 4. Популярность категорий первой половины рабочего дня

информации. Следовательно, необходимо развивать средства и механизмы снижения нагрузок на сети операторов связи, связанные с передачей больших объемов видеоинформации, например, избегая дублирования данных и используя механизмы кеширования [12].

Статья подготовлена в рамках государственного задания ФГУ ФНЦ НИИСИ РАН на 2019 год (тема №0065-2019-0014). Для исследований использовался суперкомпьютер МВС-10П.

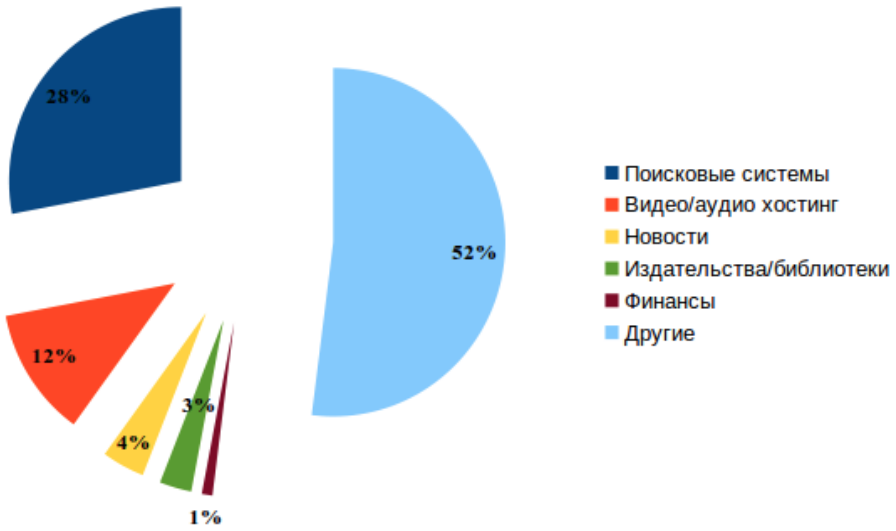


Рис. 5. Популярность категорий во второй половине рабочего дня

On the demand for Internet resources in the scientific and educational network

Y.N. Morin, D.V. Vershinin

Abstract. This article discusses the popularity rating of the categories of Internet resources in the RAS Research scientific and educational network (RASNet) depending on the time of day and per day on the example of the 9th week of 2019. The dominant position in demand is occupied by search engines. This emphasizes the importance of searching of information and choosing sources of information. Video sources on the Internet are ahead of the demand for text. It confirmed by the forecast and growth dynamics of traffic in the Internet.

Keywords. RASNet, DNS, site databases, search engine, multimedia hosting, news.

Литература

1. Международный союз электросвязи // [электронный ресурс] URL: http://www.itu.int/ (дата обращения 03.07.2019)

2. International Telecommunication Union. From Wikipedia, the free encyclopedia. // [электронный ресурс] URL: https://en.wikipedia.org/wiki/International_Telecommunication_Union (дата обращения 06.08.2019)

3. Корпоративная сеть Российской академии наук (RASNet) // [электронный ресурс] URL: http://old.jscc.ru/rasnet.shtml (дата обращения 24.01.2019)

4. Вершинин Д.В., Морин Ю.Н., Овсянников А.П., Шабанов Б.М. О реализации российского сегмента Eduroam // ТРУДЫ НАУЧНО-ИССЛЕДОВАТЕЛЬСКОГО ИНСТИТУТА СИСТЕМНЫХ ИССЛЕДОВАНИЙ РОССИЙСКОЙ АКАДЕМИИ НАУК. 2017. Т. 7, № 4. С. 162-167.

5. Ли Крикет, Альбитц Пол. «DNS и BIND», 2002, стр. 24-29.

6. DOMAIN NAMES - CONCEPTS AND FACILITIES // [электронный ресурс] URL: https://tools.ietf.org/html/rfc1034 (дата обращения 22.09.2019)

7. DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION // [электронный ресурс] URL: https://tools.ietf.org/html/rfc1035 (дата обращения 22.09.2019)

8. Cisco Talos // [электронный ресурс] URL: https://talosintelligence.com (дата обращения 12.10.2019)

9. Shalla Secure Services // [электронный ресурс] URL: http://www.shallalist.de/ (дата обращения 12.10.2019)

10. The Université Toulouse 1 Capitole // [электронный ресурс] URL: http://dsi.ut-capitole.fr/ (дата обращения 12.10.2019)

11. Cisco Visual Networking Index: Forecast and Methodology, 2016–2021 // Cisco Systems Inc., 2017 [электронный ресурс] URL: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html (дата обращения 22.09.2018).

12. Морин Ю.Н. On the Issue of Multimodal Data Flow Control on Limited Communication Channels // Lobachevskii Journal of Mathematics. 2019. Vol. 40, Iss. 5. С. 562–565.

Современные подходы и технологии, применяемые при создании и модернизации перспективных систем на ранних стадиях жизненного цикла

А.Д. Чопорняк, Б.М. Шабанов
МСЦ РАН – филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия
{chopor,shabanov}@jscc.ru

Аннотация. В настоящей работе приведены подходы и технологии, которыми могут воспользоваться предприятия из различных отраслей промышленности при создании современной высокотехнологичной конкурентоспособной продукции на ранних стадиях жизненного цикла системы (замысел, НИР, формирование ТЗ на аванпроект/техническое предложение). Данные подходы позволяют выявить потребность в разработке данной системы, определить ее роль и место на рынке, оценить потенциальную прибыль от реализации проекта, выявить всевозможные риски.

Ключевые слова. Системная инженерия, успешная система, управление требованиями, MBCD Model Base Conceptual Design.

Введение

При создании современной высокотехнологичной конкурентоспособной продукции (авиационная, космическая техника, электроника и т.п.) необходимо учитывать постоянно повышающиеся требования пользователей/заказчиков, предъявляемые к ней, причем нужно рассматривать совокупный комплекс предъявляемых требований. Помимо того, что продукт должен обладать улучшенными характеристиками по сравнению с конкурентами, он должен иметь: высокое качество, требуемую надежность, безопасность, конкурентную стоимость (как конечную стоимость продукта, так и затраты на опытно-конструкторские работы), удобство эксплуатации и техническое обслуживание. Успешный продукт - продукт, который отвечает требованиям «заинтересованных сторон» стейкхолдеров (stakeholders) [1]: пользователей, заказчиков, потребителей, поставщиков, разработчиков – всех, кто каким-то образом оказывает влияние на продукт. В качестве примера можно рассмотреть космический ракета-носитель РН. Одно из основных ключевых требований, предъявляемых стейкхолдерами это способность выводить требуемую полезную нагрузку. Требование, например, звучит так – космический ракета-носитель РН должен выводить на орбиту высотой h километров, и наклонением i градусов, полезную нагрузку m тонн. Для выполнения этого требования необходимо организовать работу всей кооперации участников, взаимодействующих с данной системой:

- разработчик маршевых ступеней и боковых блоков;
- разработчик маршевых двигателей;
- разработчик системы управления;
- разработчик разгонного блока;
- разработчик полезной нагрузки (космические аппараты или возвращаемый аппарат);
- разработчик систем стартового комплекса;
- разработчик систем энергообеспечения;
- разработчик заправочных системы и т.п.

Эти системы можно рассматривать только в комплексе поскольку хоть маршевые двигатели или космический аппарат очень важные системы РН, но сами по себе они это требование (вывод полезной нагрузки) выполнить не смогут, и удовлетворить это требование возможно только как результат поведения всех составляющих РН, а также систем, которые РН обслуживают. В итоге система – набор механизмов, людей, программного обеспечения, которые согласованно, коллективно, ритмично действуют для выполнения требований всех стейкхолдеров. Невыполнение какого-либо требования к одной из систем может привести к непредвиденным затратам/невыполнению проекта в целом. Основной целью становится разработка инновационной системы, впервые появившейся на рынке в возможно короткие

сроки, из которой все стейкхолдеры начиная от разработчиков до заказчиков и конечных потребителей/клиентов извлекли выгоду (деньги, технологии, новые возможности и т.п.).

Системы

Для определения границ проекта и определения объема работ, нужно выявить основные системы, влияющие на успешность проекта, что в этих системах необходимо создать\доработать\адаптировать не упуская из виду основную цель проекта. Различают следующие виды систем. **Целевая система** – система, при эксплуатации которой внешние стейкхолдеры (не разработчики) получают прибыль\удовлетворяют свои потребности. В качестве примера, рассмотрим в виде целевой системы РН (Рис. 1).

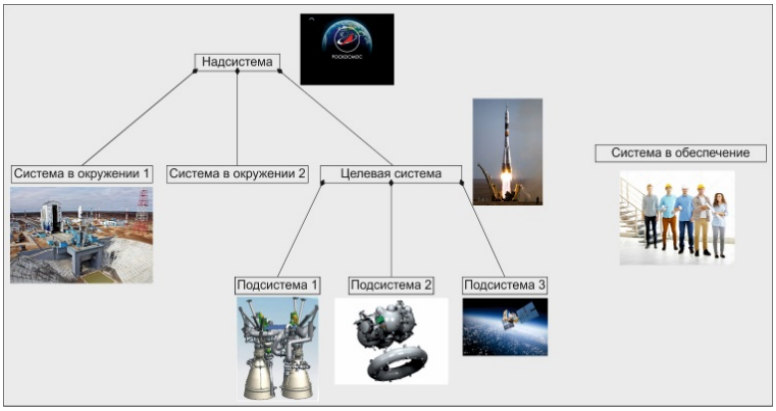


Рис. 1. Целевая система как составная часть в иерархии систем, участвующих в проекте.

Целевая система РН здесь скорее назначена, она должна выявляться стейкхолдерами в самом начале проекта, потому что целевой системой, удовлетворяющей требования по запуску космических аппаратов, может быть, например, сервис по предоставлению космических услуг, и РН в этот сервис будет входить как одна из подсистем. А для разработчиков составных частей РН таких как двигатели, космические аппараты (отдельные предприятия) целевыми системами будут двигатели и космические аппараты соответственно, но нельзя упускать из виду, что только при слаженном взаимодействии всех систем вместе может наступить системный эффект эмерджентность (emergent properties) из которого все стейкхолдеры извлекут максимальную пользу. У системы всегда имеется **надсистема** – система, которая эксплуатирует, использует в своих интересах целевую систему, извлекает из нее пользу. Надсистема формирует ключевые требования к системе. Если систему никто не эксплуатирует, она не требуется. В данном случае для РН надсистемой будет фирма, предоставляющая услуги по запуску спутников с помощью эксплуатации РН (использование целевой системы в своих интересах). На Рис. 1 не показаны уровни выше и ниже, но у каждой надсистемы есть над надсистема уровня выше, так и у подсистемы есть под подсистема, все зависит от точки зрения с которой мы смотрим на систему. Целевая система состоит из **подсистем** в данном случае это двигатели, разгонный блок, космический аппарат. **Система окружения** – стартовый комплекс, внешняя окружающая среда – внешние условия, влияющие на РН на стартовом комплексе (ветровые нагрузки, температурный диапазон), нагрузки при старте и во время полета. **Система в обеспечение** – не входит в структуру целевой системы, но она оказывает на нее воздействие переводя её от состояния к состоянию на всем протяжении жизненного цикла. Она задумывает целевую систему, проектирует, конструирует, производит, модернизирует, утилизирует. Только при слаженном взаимодействии всех без исключения систем можно создать высокотехнологичную конкурентоспособную продукцию, максимально учесть потребности (ожидания) всех стейкхолдеров.

Нормативная база

Высокотехнологичная продукция состоит из большого количества разнообразных систем/агрегатов/сборок, взаимодействующих между собой, для создания которых

необходима высокая компетенция в самых различных отраслях, в связи с этим каждый раз выбирается кооперация предприятий под определенный проект с учетом требуемых возможностей каждого из предприятий, также в кооперацию могут входить международные предприятия. Слаженная, ритмичная, коллективная работа над проектом уникальной кооперации в настоящее время не возможна без обмена информацией в единой информационной системе, содержащей всю информацию о проекте. С учетом того, что некоторые проекты по созданию высокотехнологичных систем продолжаются длительное время (более десяти лет), за этот период может измениться как кооперация, так и технологии. Для учета этих факторов используется практика обмена инженерными данными, с согласованием контрактов интерфейсов между модулями с опорой на международные стандарты в области инженерии. Преимущество такого подхода состоит в том, что международные стандарты с определенной периодичностью пересматри-ваются группами специалистов, учитываются всевозможные изменения технологий (практически все стандарты системной инженерии имеют соответствующие ГОСТ). Конечно полностью специфики ведения проекта данные стандарты не смогут заменить (при необходимости выпускаются регламен-тирующие документы и стандарты предприятий), но использовать их как основу ведения проекта является хорошей практикой [2].

Потребности, проблемные моменты

Для того чтоб заинтересовать внешних стейкхолдеров (если система не будет создаваться за счет собственной прибыли) и получить средства на создание системы, необходимо провести всесторонний анализ будущего проекта, выявить требования, предъявляемые к системе, оценить стоимости, как по созданию системы, так и по конечной стоимости на рынке, оценить ожидаемую прибыль. Чем детальнее и точнее будет проведена предварительная работа, тем проще будет доказать рентабельность проекта и принять Решение о начале работ.

Также необходимо учитывать такие практики как продажи и маркетинг, надежность, безопасность, экологическая безопасность, соответствие нормативным базам и законодательству и т. п. Если при проектировании не учесть (упустить) мнение одного из стейкхолдеров, это может привести к значительным затратам (временным и денежным) или повлиять на успешность создаваемой системы в целом. Затраты на устранение ошибок (упущений) на поздних стадиях проектирования значительно выше (по некоторым данным в 250 раз) по сравнению с ошибкой (упущением) на стадии формирования требований [3].

Одним из первоначальных шагов является детальное описание потребностей (need) стейкхолдеров или проблемных моментов существующей системы (прототип), параметры неудовлетворенности заказчика. Так же «двигателем» к открытию проекта может стать возникновение технологии, позволяющей, например, снизить затраты на производство и повлиять в конечном счете на конечную себестоимость продукции. Если проблем, недостатков в существующей системе не удалось выявить, текущие потребности стейкхолдеров удовлетворены, то такая система не будет востребована на рынке и проект в конечном счете будет убыточный.

На примере РН представим, как бы могло выглядеть описание пользовательских потребностей (проблемных моментов):

- имеется потребность в одновременном выведении 3х космических аппаратов, суммарной массой m кг, со следующими габаритами (приведены требования на габариты), на орбиту (приведены требования к орбитам), что может заинтересовать следующих потенциальных заказчиков (приведен круг заказчиков), в настоящий момент данные задачи решаются только РН тяжелого класса и имеют ограниченное число запусков и высокую стоимость выведения 1 кг полезной нагрузки (ПН) (не занятая ниша рынка);
- с 20XX года, компания (приведено название компании) планирует выпуск РН, позволяющего выполнять схожие задачи с РН, выпускаемой «нашей» компанией, но с ожидаемой стоимостью вывода 1 кг ПН ниже на 10%, что может привести к уменьшению текущих заказов (снижение эффективности текущей системы);
- с 20XX года ожидается введение ограничений по времени пребывания отработанных ступеней на орбите не менее t лет, а в последствии к

принудительному приводнению в территориях мирового океана, что может повлиять на схему выведения и сокращению предельной массы, выводимой ПН m кг. на орбиту (приведены характеристики орбиты) (введение дополнительных ограничений, ранее не существовавших).

Проблемные вопросы и потребности обсуждаются стейкхолдерами: как внешними будущими пользователями (заказчиками разрабатываемой системы, кто будет извлекать из нее пользу, зарабатывать с её использованием), так и внутренними (проектировщики, конструктора, расчетчики, технологи). Для удобства обсуждения проблемных моментов/потребностей, им, как правило, присваивается уникальный идентификатор id, краткий заголовок, описание и пояснение рис. 2. [4].

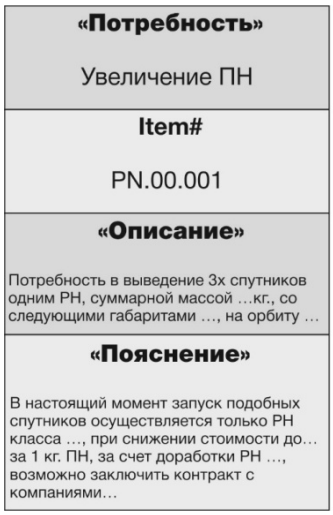


Рис. 2. Пример, вида объекта потребность/проблема в информационной системе.

Это одни из первых объектов, документируемые в информационной системе. Они должны содержать конкретную информацию описывающую потребность/проблему, которая однозначно бы её определяла (формулировка в общих чертах, можно неправильно выявить требование, которое появится в последствие в системе на основании этого объекта). Потребности/проблемы согласовываются со стейкхолдерами, и утверждаются в информационной системе. Описание проводится с использованием различных точек зрения, поскольку взгляды у разных стейкхолдеров на одну и ту же систему могут значительно отличаться. Ответственный за это направление работ **системный инженер по требованиям**. При назначении ответственных за направления работ вводится ролевое назначение в проекте, а не должностное. Одну роль может выполнять как один сотрудник, так и группа, также один сотрудник может выполнять несколько ролей. При описании проблем/потребностей с учетом разных точек зрения могут возникать коллизии, их обсуждают и решают на совещаниях.

В результате данных работ в информационной системе появляются описанные и признанные (документированные) утвержденные в информационной системе объекты определяющие потребности/проблемы, они понятные всем стейкхолдерам и не противоречат между собой.

Требования

На основании потребностей/проблем проводится выявление и описание первоначальных требований, предъявляемых к системе. Ключевое слово здесь **выявление** требований, а не назначение. Из-за сложности высокотехнологичных систем к ним предъявляются тысячи или десятки тысяч требований (масштабируется в процессе жизненного цикла), которые необходимо контролировать, верифицировать и валидировать, выполнять данные работы вне информационной системы достаточно проблематично. Требования должны быть сформулированы четкими и однозначными формулировками для последующего их размещения в информационной системе. Они вводятся в информационную систему в виде специальных объектов item, ассоциативно связанных с объектами потребность/проблема, для них вводится гораздо большее количество атрибутов, по которым требования в последствии классифицируются и структурируются: по определению, назначению, контролю статуса (просмотрено/согласовано/утверждено), контролю связей в структуре требований, анализу выполнения и т.п.

При формировании требований желательно пользоваться специальным языком и определенными шаблонами, что позволит системному инженеру по требованиям качественнее сформировать структуру/матрицу требований, меньше вводить изменений в процессе работ [5]. Также необходимо учитывать следующие отличия по уровню требований:

Ключевые требования (требования заказчика\пользователя) – то что заказ-

чик\пользователь ожидает получить от предлагаемой системы.

Системные требования – каким образом предлагаемая система будет отвечать ожиданиям заказчика/пользователя.

Архитектурные решения – как конкретная архитектура (из нескольких вариантов) отвечает системным требованиям.

Рассмотрим на примере пусковых услуг.

Ключевые требования: Космический аппарат (КА) должен функционировать в течении х лет, предоставлен перечень функций, выполняемых данным КА. Требования не должны в своем описании содержать технические решения с помощью какого РН данный КА должен быть выведен на орбиту (ограничения КА по максимальным перегрузкам, температурному диапазону и т.п. обязательно должны учитываться) в конкретное местоположение где он будет определенное время функционировать и приносить пользу. Такая формулировка позволит системным инженерам подобрать наилучшее решение имея гораздо меньше ограничений в свободе выбора.

Системные требования: РН с разгонным блоком должен иметь возможность выводить КА массой m на орбиту (приведено описание орбиты). Требования не должны содержать количество ступеней РН, типы двигателей, конкретизация разгонного блока.

Архитектурные решения будут основываться на ограничениях, исходящих от пользовательских, системных требований и различных ограничений, налагаемых законодательством, окружающей средой и т.п. Различные варианты РН с разными количествами ступеней и боковых блоков, различные типы топлива на ступенях, различные разгонные блоки и т.п.

Ключевые требования относятся к области проблем/потребностей, а системные и архитектурные к области решения этих проблем. Такое разделение очень важно для понимания ограничений (масштаба) предлагаемой системы, какие требования исходят от заказчика, а какие от исполнителей. Это позволит при выборе варианта не забыть главного (в первую очередь учесть требования заказчика\пользователя). Без выполнения ключевых требований система не интересна стейкхолдерам или данные требования не являются ключевыми.

Детализация первоначальной структуры требований индивидуальна для каждого проекта, но она в любом случае должна не иметь коллизий между требованиями и минимально представлять из себя некий «черный ящик», который понимаемо ведет себя в надсистеме (выполняет требования, предъявляемые к системе, которая его эксплуатирует, обслуживает и т.п.), как он устроен внутри (архитектурные решения) обсуждаются позднее.

Предварительный технический облик

На основании требований, предъявляемых к системе, проводится поиск технического облика/технических обликов/концептов систем, model base conceptual design (MBCD). Основная работа на этом этапе – раскрытие «черного ящика», какие функции он должен выполнять, возможно ли требуемую функцию закрыть одним модулем, или нужно дополнительное разбиение на подсистемы Рис 3. [6].

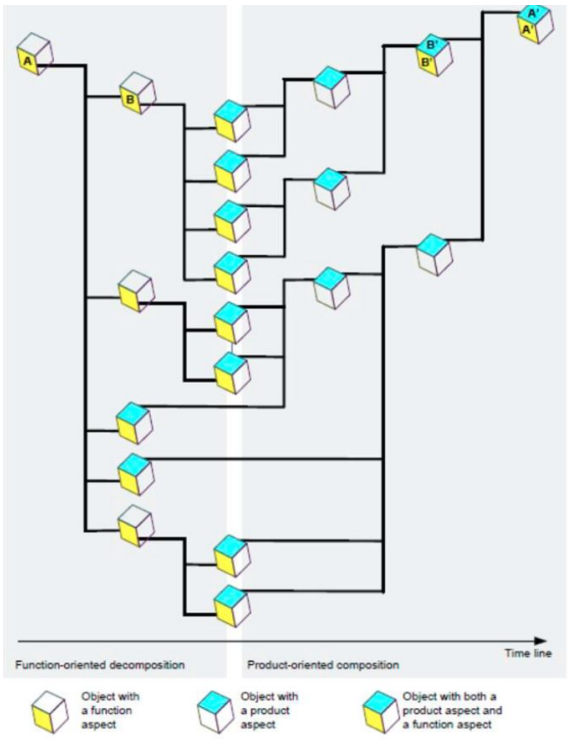


Рис. 3. Функционально – логическая схема укрупненного технического облика системы.

На рисунке показано, что в данной структуре системы каждому функциональному аспекту соответствует от одного до четырех модулей (product), которые полностью «покрывают» требуемый функционал, далее подсборки собираются в сборки и агрегаты, и в конечном счете головная сборка системы полностью

выполняет все необходимые функции для работы в надсистеме. Ответственная роль за это направление работ **архитектор проекта**. Как правило на данном этапе предлагаются несколько концептуальных дизайнов системы, из которых впоследствии при детальном анализе (окончание этапа эскизного проектирования) будет выбран вариант наиболее полно отвечающей потребностям стейкхолдеров.

При разработке концептуального дизайна необходимо рассматривать создаваемую систему на всех стадиях жизненного цикла от задумки, проектирования, производства, вывода системы на рынок, присутствия на рынке, модернизации, утилизации. Также для успешной системы очень важно попасть в так называемое «окно возможностей» – период времени, когда система будет востребована на рынке. Например, если вести разработку дольше потребного времени, «окно возможностей» может «закрыться» и разработанная система не в срок не принесет ожидаемой прибыли.

В процессе разработки системы она постоянно изменяется и уточняется по мере накопления знаний и совершенствования моделей, её описывающих, при этом требования, предъявляемые к системе тоже до определенного времени, корректируются Рис. 4. [7]

Рис. 4. Периодическая проверка/уточнение основных артефактов, по мере увеличения знаний о системе в течении проекта.

С каждой новой версией системы она уточняется и детализируется, а периодический пересмотр и повторение операций позволяет не упустить главного и не допустить серьезных упущений при разработке. Поскольку при данном подходе имеется прозрачная система управления требованиями в информационной системе, то в любой момент можно уточнить на каком основании изменялось то или иное требование, кто внес изменения, на что это изменение влияет и это не сдерживает, а наоборот способствует динамичному ведению проекта.

Принятие решение о начале работ по проекту

Имея перечень артефактов, полученных на предыдущих этапах работ, необходимо данные знания сформировать в виде документа (коммерческое предложение/отчет/презентация), по результатам рассмотрения которого принимается Решение, стоит ли этим направлением работ в принципе заниматься, какая и когда будет прибыль от разработанной системы, всевозможные риски. Для этого в документе должны максимально полно отражены все аспекты создаваемой системы, проведен детальный анализ (насколько это возможно на начальной стадии проектирования), и не упущено ничего важного.

Широко распространенный способ используемый в системной инженерии, представленный в ряде международных стандартов, в частности [8] это использование способа продвижения ALPHA Abstract-Level Progress Health Attribute, но неформально это просто «идеальный рабочий продукт», названный «альфой» для уменьшения путаницы с «реальными рабочими продуктами» и аббревиатура для него была подобрана задним числом [9]. Данный способ позволяет по состоянию каждой из альф оценить степень продвижения проекта, задавать «правильные» вопросы к планируемой системе и выстраивать последовательность действий, не упуская из вида ничего важного.

В итоге разработанный документ всесторонне обсужден стейкхолдерами (как внешними, так и внутренними), имеется понимание что, кто, когда, зачем, почему и как делает, какую пользу возможно извлечь из предлагаемой системы. Только после этого принимается решение о финансировании данного проекта и открытия работ, до этого все работы, как правило, носят инициативный характер. В ряде случаев для выявления успешности системы проводят работы, относящиеся к более поздним стадиям проектирования, такие как моделирование, многокритериальные оптимизации системы, расчеты

отдельных узлов численными методами. Универсального подхода нет, все зависит от специфики проекта. Если по результатам предпроектных работ выясняется, что проект убыточный и им не стоит заниматься, это так же хороший результат, позволяющий значительно сэкономить время, ресурсы, затраты и направить их на более перспективные направления.

Заключение

Применение данного подхода и технологий при создании высокотехнологичной конкурентной продукции на ранней стадии проектирования позволяет выявить перспективность предлагаемого проекта, выявить всевозможные риски и не упустить важного при создании системы, что бы это ни было. Это хорошая основа для использования в проекте полученных знаний, информации, исходных данных, предъявляемых требований в PLM, ERP впоследствии в MES системах на всем протяжении жизненного цикла изделия.

Работа выполнена в МСЦ РАН в рамках государственного задания по теме 0065-2019-0016 (рег. номер АААА-А19-119011590098-8).

Modern design technologies in the creation and modernization of advanced systems at the early stages of the life cycle

A.D. Chopornyak, B.M. Shabanov

Abstract. In this paper, the approaches and technologies that can be used by enterprises from different industries in the creation of modern high-tech competitive products at the earliest stages of the life cycle of the system (design, research, formation of TK for the advance project\technical proposal) are presented. These approaches allow us to identify the need for the development of this system, to determine its role and place in the market, to assess the potential profit from the project, to identify all kinds of risks.

Keywords: Systems engineering, successful system, requirements management, MBCD Model Base Conceptual Design

Литература

1. Guide to the systems engineering body of knowledge. // URL: https://www.sebokwiki.org/wiki/Stakeholder_(glossary) (дата обращения 23.10.2019)
2. Основные стандарты системной инженерии: ISO/IEC/IEEE 15288:2015 Systems and software engineering – System life cycle processes ГОСТ Р 57102-2016; ISO 42010 Systems and software engineering – Architecture description – ГОСТ Р 57100-2016; ISO 81346-1:2009 Industrial systems, installations and equipment and industrial products – Structuring principles and reference designations – Part 1: Basic rules ГОСТР МЭК 62023-2016.
3. P. Davies. Quantification of the Value of Systems Engineering. // Thales UK INCOSE UK ASEC November 2013, URL: https://www.bristol.ac.uk/media-library/sites/eng-systems-centre/migrated/documents/pdavies-blockley-lecture.pdf (дата обращения 23.10.2019).
4. J. Hummel. Model Based Conceptual Design (MBCD). // MBSE Solutions, LLC, URL: http://mbse.solutions/wp-content/uploads/2016/09/MBSESolutions_MBCD_V05.pdf (дата обращения 23.10.2019).
5. E. Hull, K. Jackson, J. Dick. Requirements Engineering. // Springer, 2011.
6. А. Левенчук. Системное мышление. // Издательские решения, 2019.
7. Модели жизненного цикла программного обеспечения // URL: http://libraryno.ru/4-5-modeli-zhiznennogo-cikla-programmnogo-obespecheniya-trpo (дата обращения 23.10.2019)
8. Essence – Kernel and Language for Software Engineering Methods Version 1.2 Standard document. // URL: https://www.omg.org/spec/Essence/ (дата обращения 23.10.2019).
9. Systems Engineering Thinking Wiki. // URL: http://sewiki.ru/Категория:Альфы (дата обращения 23.10.2019).

Вопросы совершенствования российского сегмента сервиса роуминга в беспроводных сетях eduroam в условиях интеграции научно-образовательных сетей RUNNet и RASNet

А.Г. Абрамов [1], И.В. Васильев [1], Ю.Н. Морин [2], А.П. Овсянников [2], В.А. Порхачёв [1]

[1] СПбО МСЦ РАН – филиала ФГУ ФНЦ НИИСИ РАН, Санкт-Петербург, Россия

[2] Межведомственный суперкомпьютерный центр РАН – филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия

E-mail's: [1] abramov@runnet.ru, [1] vasilyev@runnet.ru, [2] morin@rasnet.ru, [2] apo@jscc.ru, [1] porhachev@runnet.ru

Аннотация: В статье дается обзор технических решений и организационных аспектов, обеспечивающих многолетнее успешное функционирование международного сервиса роуминга в беспроводных сетях eduroam. Представляется действующая архитектура российского сегмента сервиса, обсуждаются мероприятия, нацеленные на ее совершенствование, реализацию сопутствующих eduroam прикладных сервисов в интересах организаций науки и образования России, на повышение уровня участия в международных проектах, базирующихся на технологиях федеративной аутентификации, в условиях происходящей интеграции отраслевых научно-образовательных сетей RUNNet и RASNet.

Ключевые слова: eduroam, федеративная аутентификация, удостоверяющая федерация, беспроводные сети, Wi-Fi, RADIUS, национальные научно-образовательные сети, НИКС, GÉANT, NREN, RUNNet, RASNet

Введение

Сегодня научно-исследовательская и образовательная деятельность все более интернационализируются, а глобальное международное сотрудничество проникает в большинство как фундаментальных, так и прикладных областей знаний.

Исследователи, преподаватели и учащиеся объединяются в распределенные коллаборации для эффективной и результативной реализации совместных проектов. Научная работа и образовательный процесс уже не ограничиваются рамками учебной аудитории или исследовательской лаборатории: участникам необходим постоянный и повсеместный доступ в Интернет, к сетевым ресурсам и сервисам там, где это необходимо в данный момент.

Ответственность за ИКТ-поддержку высокомобильной сферы образования и науки по всему миру исторически берут на себя национальные научно-образовательные телекоммуникационные сети (National Research and Education Networks, NREN) и межгосударственные сетевые консорциумы [1, 2].

Решение, казалось бы, простой задачи по обеспечению "гостей" организации – участников конференций, приглашенных преподавателей, иностранных студентов безопасным и повсеместным доступом в Интернет через локальный сервис Wi-Fi может потребовать привлечения значимых административных и технических ресурсов. Необходимый функционал прозрачного в использовании сервиса роуминга в беспроводных сетях для научно-образовательного сообщества реализуется в рамках международного проекта eduroam (education roaming) [3, 4]. Идея реализации проекта была инициирована общеевропейским сетевым консорциумом GÉANT в 2003 году, а в качестве полноценно функционирующего сервиса eduroam успешно и с постоянно расширяющейся географией охвата развивается с 2008 года.

Учащиеся, преподаватели и научные работники получают возможность бесплатного подключения к сети Интернет и ресурсам NREN в научных и образовательных организациях внутри страны и за рубежом, в кампусах участников проекта eduroam. Защищенный доступ к сервису с личных портативных и мобильных устройств осуществляется на основе единых учетных данных (логина и пароля), выданных пользователю "домашней"

организацией (в которой он работает или учится).

Сервис eduroam в настоящее время функционирует более чем в 100 странах по всему миру, оказывая поддержку процессам научного сотрудничества и образовательной мобильности тысячам научных и образовательных организаций [3]. Количество зон доступа к сервису уже превышает 10 тысяч, и ежегодно регистрируется более миллиарда сессий аутентификации пользователей.

В нашей стране сервис eduroam был изначально реализован в 2011 году Межведомственным суперкомпьютерным центром Российской академии наук (МСЦ РАН) [5, 6].

Новый импульс к развитию российского сегмента сервиса был дан в 2017 году после заключения МСЦ РАН и (тогда еще функционировавшим) ФГАУ ГНИИ ИТТ "Информика" Соглашения о сотрудничестве в области создания и развития интегрированной инфраструктуры авторизации и аутентификации в рамках международных проектов eduroam и eduGAIN на базе научно-образовательных сетей RUNNet и RASNet [7, 8].

В настоящей статье приводятся общие сведения о некоторых организационных и технических аспектах функционирования международного сервиса eduroam, представляется текущая архитектура российского сегмента проекта, описываются находящиеся в стадии проработки и/или реализации мероприятия, нацеленные на совершенствование архитектуры, а также создание новых и содержательное участие в существующих сопутствующих проекту специализированных прикладных сервисах.

1. Eduroam: общие сведения о функционировании сервиса

В терминологии eduroam организация, предоставляющая своим работникам или учащимся, а также гостям доступ в Интернет (обычно через беспроводную локальную сеть) называется провайдером сервиса (Service Provider, SP), а домашняя организация, куда происходит обращение для проверки подлинности учетных данных пользователя – провайдером идентификации (Identity Provider, IdP) [3, 7, 8].

Функционирование сервиса основано на IEEE стандарте сетевой безопасности в публичных местах 802.1X и иерархии взаимодействующих между собой RADIUS-серверов (см. рис. 1). Стандартом 802.1X определяется протокол контроля доступа и аутентификации для пользовательских устройств, которые желают подключиться к проводной или беспроводной локальной сети.

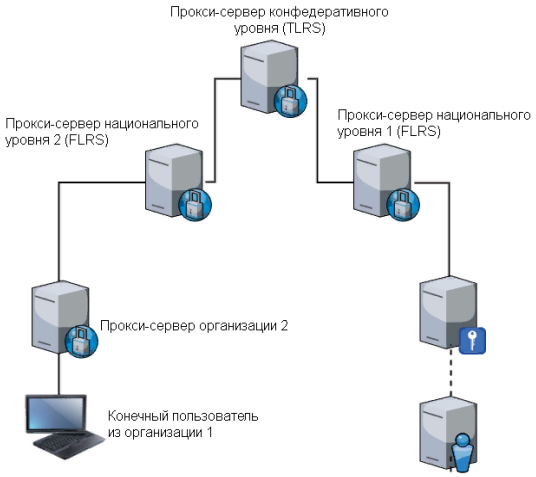


Рис. 1. Общая схема авторизации пользователя в сервисе eduroam

Аутентификация конечных устройств пользователей базируется на протоколе EAP (Extensible Authentication Protocol) – универсальном фреймворке аутентификации, реализующим ряд общих функций и обеспечивающим согласование используемых IdP методов проверки подлинности (методов EAP), среди которых можно упомянуть EAP-TLS, EAP-TTLS и EAP-PEAP.

Стандарт 802.1X описывает процесс инкапсуляции сообщений EAP, передаваемых между запрашивающими доступ устройствами (клиентами, например, смартфон, планшет, ноутбук), серверами контроля доступа к сети (Network Access Server, NAS, например, Wi-Fi точка доступа) и серверами аутентификации (Authentication Server, AS, например, RADIUS-сервер).

RADIUS (Remote Authentication Dial-In User Service) представляет собой протокол, разработанный для выполнения функций аутентификации, авторизации и сбора сведений об использованных ресурсах, в соответствии с которым происходит обмен данными между NAS и AS.

В процессе подключения к беспроводной сети с SSID (Service Set Identifier) eduroam клиентское устройство взаимодействует по протоколу EAP с NAS, передавая учетные данные по локальной сети (EAP over LAN, EAPoL). NAS инкапсулирует поступивший EAP-запрос и транспортирует его на RADIUS-сервер (AS), ожидая от него выполнения аутентификации с последующим приемом, либо отклонением запроса пользователя на доступ к сети.

С целью обеспечения конфиденциальности передачи данных по беспроводным сетям

применяются развитые технологии шифрования WEP (Wired Equivalent Privacy) и WPA/WPA2 (Wi-Fi Protected Access), поддерживающие шифрование в соответствии со стандартом AES (Advanced Encryption Standard).

Запрос к серверу аутентификации SP содержит в себе зашифрованную часть (логин и пароль пользователя), и открытую, адресную часть – доменное имя IdP его домашней организации.

Имя пользователя (логин) в соответствии с правилами eduroam должно иметь вид user@realm, где realm это доменное имя IdP (обычно institution.tld, tld – домен верхнего уровня). На основе realm и выполняется маршрутизация запросов в распределенной структуре RADIUS прокси-серверов.

На вершине связанной иерархической архитектуры eduroam расположены два головных RADIUS прокси-сервера конфедеративного уровня (Top-Level RADIUS Servers, TLRS), на каждом из которых ведется синхронизированный актуальный список подключенных к сервису верхнеуровневых доменов национальных федераций, традиционно обслуживаемых NREN соответствующих стран.

Следующая ступень иерархии отводится RADIUS прокси-серверам федеративного уровня (Federation-Level RADIUS Server, FLRS), находящимся под управлением национального оператора eduroam (National Roaming Operator, NRO) и содержащим конфигурационную информацию о подключенных к национальной удостоверяющей федерации (УФ) конкретной страны организациях и их ролях.

Поступившие от клиентского устройства "гостя" в SP учетные данные передаются через иерархию промежуточных TLRS/FLRS RADIUS прокси-серверов RADIUS-серверу домашней организации (определяемой по realm), который, взаимодействуя с локальным сервером учетных записей организации, осуществляет проверку их правильности и инициирует передачу обратно SP сообщения о разрешении (или отказе) авторизации пользователя в сервисе.

Домашняя организация, выступая в качестве IdP, отвечает за аутентификацию своих пользователей, а также за поддержание учетных данных в актуальном состоянии, включая и ситуации, когда учащийся или работник находится в гостевых кампусах в стране или за рубежом. Таким образом, учетные данные пользователей хранятся и обрабатываются локально и никогда не передаются в другие участвующие в проекте организации.

Техническое развертывание сервиса eduroam в конкретной организации предполагает наличие в ней функционирующей беспроводной сети, базы данных с учетными записями пользователей и выполнение трех последовательных шагов:

– настройка локального RADIUS-сервера с подключением его к серверу IdP организации;
– подключение Wi-Fi точек доступа к RADIUS-серверу организации;
– подключение локального RADIUS-сервера к RADIUS прокси-серверу (или серверам) федеративного уровня (FLRS) во взаимодействии с NRO.

В организационно-документальной части претендующие на присоединение к проекту организации должны подписать с NRO своей страны Регламент национальной УФ eduroam [9], который описывает роли и зоны ответственности оператора и ее участников, необходимые для развертывания и поддержания в рабочем состоянии базовых сервисов, особенности протоколирования сопутствующих функционированию сервиса событий и поддержки пользователей. Каждая организация должна, в частности, явно обозначить в Регламенте свою роль как участника проекта – SP, IdP или совмещающая обе эти функции.

Осуществляя деятельность по проекту eduroam в качестве IdP, в целях выполнения требований Федерального закона от 27.07.2006 №152-ФЗ "О персональных данных" и Постановления Правительства Российской Федерации от 01.11.2012 №1119 "Об утверждении требований к защите персональных данных при их обработке в информационных системах персональных данных" организация обязана руководствоваться локально утвержденным Положением о единой учетной записи. В этом документе должны быть отражены роль и ответственность служб организации по сопровождению жизненного цикла учетных данных пользователей, а также персональная ответственность пользователей за сохранность личных учетных данных.

Следует заметить здесь, что при координации специалистов консорциума GÉANT развивается и поддерживается целый ряд востребованных операционных и пользовательских сервисов eduroam, среди которых можно упомянуть:

– сайт-агрегатор операционных сервисов (monitor.eduroam.org);
– база данных организаций-участников проекта;

– интерактивная карта зон доступа eduroam (monitor.eduroam.org/map_service_loc.php, рис. 2);
– карты статистики сессий аутентификации пользователей, приезжающих в/из определенной страны из/в другие страны;
– сервис централизованного мониторинга состояния RADIUS прокси-серверов FLRS;
– сервис автоматической настройки устройств для доступа к eduroam (Configuration Assistant Tool, CAT).

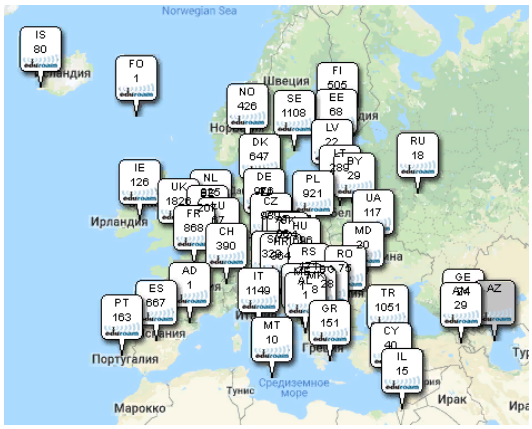


Рис. 2. Фрагмент интерактивной карты зон доступа eduroam в Европе

Поиск ближайших зон доступа eduroam с мобильных устройств может быть произведен также с помощью приложения *eduroam Companion*, разработанного NREN Великобритании Jisc.

Централизованно ведущаяся специалистами Operations Team (OT) eduroam единая база данных (БД) организаций-участников проекта используется для предоставления информации, необходимой для работы eduroam и перечисленных выше вспомогательных сервисов. БД реализует механизмы, позволяющие осуществлять автоматический сбор данных от NRO, включая общие сведения о нем самом, об организациях конкретной страны (SP и IdP), точках доступа к eduroam (местах обслуживания), мониторинг и статистику использования. NRO должен представлять актуальные данные в XML или JSON формате в соответствии с установленными спецификациями и правилами размещения информации (на локальном сайте eduroam.tld, где tld – домен верхнего уровня страны). OT предоставляет специалистам NRO веб-инструменты для валидации XML/JSON файлов и ведет онлайн-таблицу, в которой отражается доступность и правильность информации конкретных NRO.

На официальном wiki-сайте GÉANT (wiki.geant.org) в свободном доступе размещаются и систематически пополняются весьма полезные материалы, содержащие техническую документацию по развертыванию и поддержанию текущего функционирования сервиса eduroam для разных категорий участников, в том числе конечных пользователей, IT-служб организаций и NRO.

Важной функцией NRO является сбор и ведение журналов с информацией об использовании сервиса, а также регулярная отправка файлов статистики в заданном формате ("F-Ticks") в OT, который на их основе формирует визуальные карты со статистикой международного взаимодействия в рамках проекта.

F-Ticks (Federated Ticket System) представляет собой модульный продукт для сбора и визуализации данных о событиях аутентификации в eduroam, называемых *тиками* (ticks). Решение включает в себя специализированный программный пакет, инструкции по его настройке для участников проекта, SQL-схему БД, стандартизованный синтаксис сообщений для отчетов о событиях и образец веб-сайта на основе языка PHP для визуализации данных.

Базовый формат описания тика подразумевает включение в него только наиболее существенной информации о сессии аутентификации пользователя, а именно – идентификатор организации пользователя (REALM), ISO-код посещенной страны (VISCOUNTRY), MAC-адрес пользовательского устройства (CSI) и результат аутентификации (RESULT):

F-TICKS/eduroam/1.0#REALM=JSCC.RU
#VISCOUNTRY=DE#CSI=00-FF-62-37-8C-20
#RESULT=OK#

В тех случаях, когда на уровне национальной УФ требуется более подробная статистика об использовании сервиса, формат тиков может быть расширен с включением в него в качестве необязательного параметра аббревиатуры посещенной организации (VISINST):

F-TICKS/eduroam/1.0#REALM=JSCC.RU
#VISCOUNTRY=DE#VISINST=TUM.DE#CSI=00-FF-62-37-8C-20#RESULT=OK#

Сервис eduroam Managed IdP (hosted.eduroam.org) может оказаться полезным для небольших организаций, у которых отсутствуют навыки и/или ресурсы для обеспечения функционирования собственных IT-служб, поскольку перекла-

дывает функции технической настройки нового IdP на ОТ проекта.

Указанный сервис включает в себя следующие компоненты:

– веб-интерфейс для администраторов NRO по управлению IdP организаций с возможностью делегирования или отзыва прав доступа к eduroam;
– веб-интерфейс и техническую инфраструктуру для администраторов конкретных IdP, позволяющую управлять учетными данными конечных пользователей при доступе к eduroam;
– специализированную техническую инфраструктуру ("виртуальный RADIUS-сервер"), проверяющую учетные данные пользователей и предоставляющую доступ к eduroam.

Сервис eduPKI eduroam RA отвечает за создание и выдачу серверных сертификатов eduroam операторам для использования при развертывании собственной защищенной RADIUS/TLS (Transport Layer Security) инфраструктуры.

Существует два типа серверных сертификатов:

– профиль eduroam SP – для SP сервиса eduroam (операторов точек доступа eduroam или прокси-релеев для одной или нескольких точек доступа);
– профиль eduroam IdP – для нового IdP (операторов realm или прокси-релеев для одного или более IdP).

Заботясь об удобстве конечных пользователей, ОТ eduroam предоставляет возможность автоматической настройки устройств для доступа к сервису. На сайте cat.eduroam.org (Configuration Assistant Tool, CAT) пользователю предлагается выбрать из формируемого с учетом геолокации (рис. 3) списка домашнюю организацию (IdP), скачать и установить на устройство программное обеспечение (ПО) для конкретной операционной системы, после чего связать с настроенной сетью eduroam свои учетные данные.

На мобильных устройствах с операционной системой Android для этих целей можно использовать официальное приложение разработки GÉANT – eduroam CAT.

ПО CAT устанавливается в виде конфигурационного профиля, содержащего информацию о настройках канала до RADIUS-сервера домашней организации, при этом однократная регистрация в сети позволяет устройству в дальнейшем подключаться к eduroam в любом месте присутствия без необходимости в повторном вводе учетных данных.

Конфигуратор дает возможность загрузки в профиль X.509 сертификатов CA, определяющих "цепочку доверия" для серверного сертификата, пересылаемого RADIUS-сервером IdP перед началом сеанса передачи пользователем учетных данных.

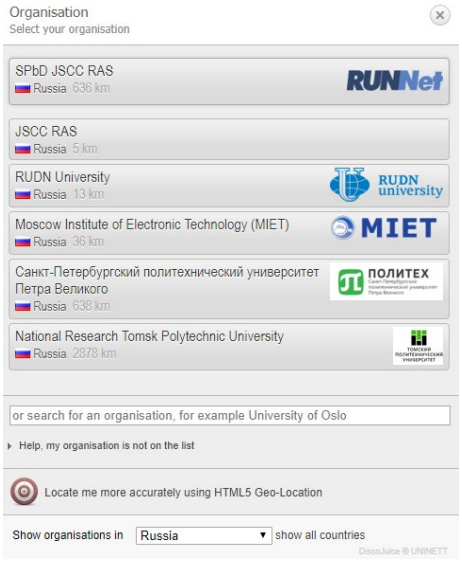


Рис. 3. Выдача списка организаций для загрузки ПО конфигуратора CAT eduroam

Веб-сервис CAT позволяет администраторам организаций самостоятельно производить настройки ПО конфигуратора. Для доступа в личный кабинет (ЛК) специалист NRO средствами сервиса по запросу направляет на e-mail администратора электронное "приглашение", в котором содержится гиперссылка и временный токен активации доступа.

Вход в ЛК может быть выполнен с помощью учетных данных пользователя в международном проекте eduGAIN [10], либо через популярные социальные сети и веб-сервисы (Google, Facebook, Twitter и др.).

В ЛК (см. рис. 4) администратору следует занести общие сведения об организации, контактные данные службы технической поддержки, свойства среды передачи данных (названия дополнительных SSID, наличие или отсутствие поддержки проводной сети и др.) и список поддерживаемых IdP организации методов EAP. После выполнения всех необходимых действий администратор получит возможность сформировать и сделать доступными своим пользователям набор программ-установщиков для настройки их конечных устройств (в виде комбинации операционной системы и метода EAP).

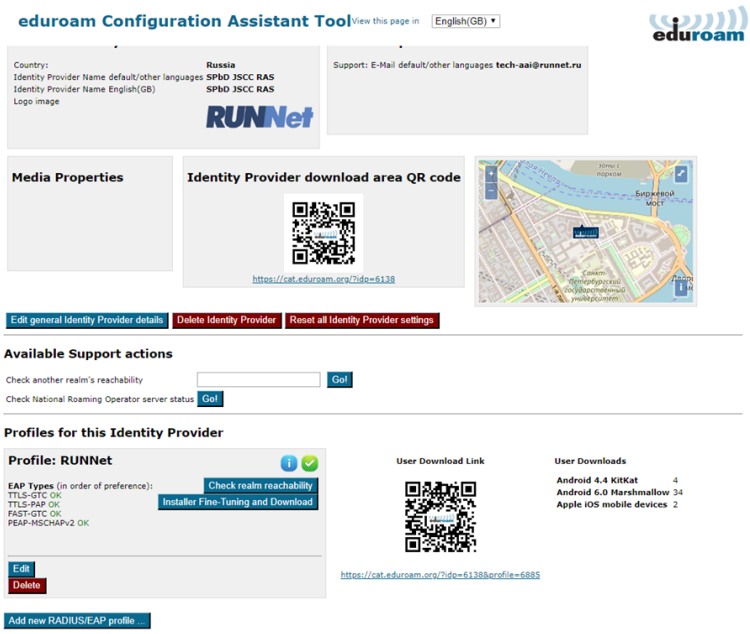


Рис. 4. Личный кабинет настройки администратором конфигуратора CAT eduroam

2. Текущее состояние российского сегмента сервиса eduroam

В течение 2017-2018 годов российский сегмент eduroam находился под совместным управлением МСЦ РАН и ФГАУ ГНИИ ИТТ "Информика" на инфраструктурной основе эксплуатируемых и систематически развиваемых ими научно-образовательных телекоммуникационных сетей – сети Российской академии наук RASNet (Russian Academy of Science Network) и федеральной университетской сети RUNNet (Russian UNiversity Network) [2, 11].

В рамках подписанного между организациями в 2017 году Соглашения о сотрудничестве было налажено оперативное и эффективное организационно-техническое взаимодействие команд, вовлеченных в выполнение проекта, и реализовано резервирование ключевых элементов инфраструктуры национального уровня (FLRS) [7, 8]. В этот период, фактически, функционировало два независимых регистратора национального уровня, уполномоченных выполнять организационные и технические мероприятия по подключению к eduroam новых организаций с формальным разделением на "зоны ответственности" в соответствии с исторически сложившимися целевыми аудиториями (научные институты РАН и организации высшего образования).

Функционировавшая в этот период архитектура российского сегмента сервиса eduroam показана на рис. 5 [8]. В принятой схеме маршрутизация запросов RADIUS-серверов аутентификации организаций-участников проекта может производиться либо напрямую через прокси-сервер национального уровня FLRS МСЦ РАН, либо через RADIUS прокси-сервер федеративного уровня RUNNet (при необходимости, с последующим транзитом через FLRS в направлении TLRS).

Узким местом такой архитектуры сервиса является единственная точка взаимодействия с TLRS, локализованная на одном сервере. Также, в силу различной ведомственной принадлежности, было недостаточно эффективно обеспечено взаимодействие корневых RADIUS прокси-серверов федеративного и национального уровней. Это впоследствии потребовало внесения изменений в общую архитектуру сервиса.

Следует отметить, что на сегодняшний день сервис eduroam уже применяется более чем в 20 научных и образовательных организациях страны, и имеется ряд заявок на подключение от новых организаций. Ежедневно более тысячи российских учащихся, преподавателей и исследователей используют возможности сервиса в поездках по стране и за рубежом, сопоставимое число зарубежных гостей пользуются eduroam в российских организациях-участниках проекта.

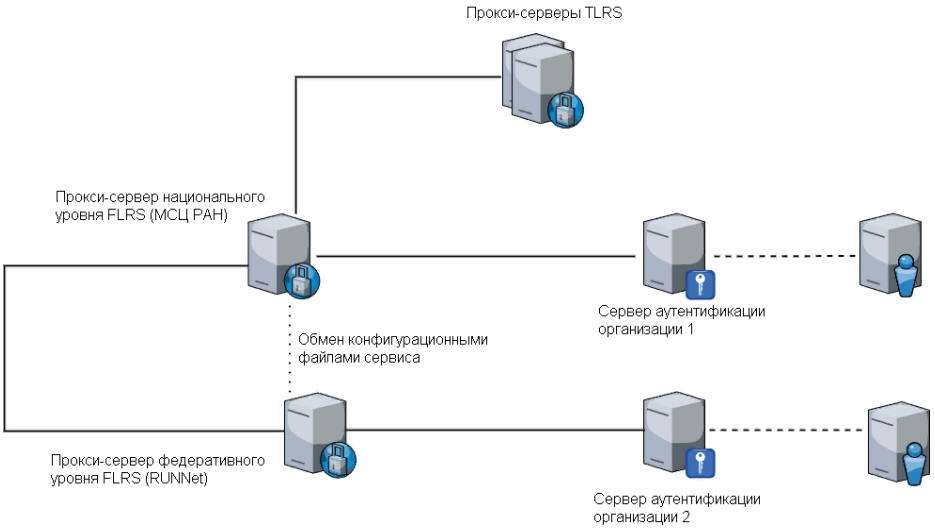


Рис. 5. Архитектура российского сегмента eduroam

3. Ключевые направления работ по развитию российского сегмента eduroam

В 2018 году по решению (тогда еще единого) Министерства образования и науки Российской Федерации ФГАУ ГНИИ ИТТ "Информика" было реорганизовано и впоследствии вновь созданными федеральными органами управления образованием наукой (Министерством науки и высшего образования Российской Федерации и Министерством просвещения Российской Федерации) было принято совместное решение о передаче функций управления сетью RUNNet в МСЦ РАН.

В 2019 г. Советом Министерства науки и высшего образования Российской Федерации по информационно-телекоммуникационной инфраструктуре, информационной безопасности и суперкомпьютерным технологиям была одобрена концепция создания Национальной исследовательской компьютерной сети нового поколения (НИКС) на основе интеграции сетей RUNNet и RASNet. НИКС должна выполнять функции национальной научно-образовательной сети (National Rresearch and Educational Network) России, в том числе на международной арене. МСЦ РАН как оператор НИКС по поручению Министерства науки и высшего образования Российской Федерации реализует комплекс мероприятий по обеспечению ее бесперебойного функционирования и развития [11, 12].

Развитие сервисов федеративной и межфедеративной аутентификации и авторизации, образовательной и научной мобильности, предполагающих, в частности, предоставление свободного и прозрачного авторизованного доступа пользователей к ресурсам NREN и собственным сетевым ресурсам и сервисам независимо от местоположения, является одной из традиционных и значимых функций NREN.

Создание НИКС на базе интеграции отраслевых сетей RASNet и RUNNet, объединение их кадрового и организационно-технического потенциала, обусловило повышение результативности работ по проекту eduroam.

В 2019 году создан новый сайт российского сегмента проекта eduroam.ru, значительно расширяющий предоставление локализованной методической и технической информации по применению и развитию eduroam в организациях науки и образования страны.

Актуализированы данные по российским организациям-участникам проекта и расположенным на их территории зонам доступа, реализован экспорт сведений в соответствии с принятыми спецификациями в OT eduroam для их отражения на глобальной мировой карте.

В стадии проработки/реализации находятся следующие основные мероприятия:

– переход российского сегмента eduroam на модернизированную отказоустойчивую архитектуру;
– объединение серверных конфигураций пользовательской базы регистраторов в общий источник обновления realm для прокси-серверов FLRS;
– сбор, резервированное хранение и обработка статистики прокси-серверов FLRS (F-Ticks) с отправкой данных на сервера OT eduroam для отражения на картах международного взаимодействия

и на собственных веб-ресурсах (в расширенном виде);
- локализация и адаптация информационных, методических и технических материалов проекта eduroam для размещения на сайте eduroam.ru;
- распространение сервиса CAT eduroam в российских организациях для упрощения настройки конечных устройств пользователей при доступе к сервису;
- разработка пользовательского интерфейса карты "eduroam Россия", обеспечивающего организациям возможности самостоятельного заполнения данных о геолокациях точек доступа;
- разработка инструментария для централизованной обработки и визуализации расширенных статистических данных об использовании сервиса организациями, в том числе для доступа к записям журналов прокси-серверов FLRS, относящихся к их зоне ответственности;
- разработка контейнера RADIUS-сервера провайдера идентификации и провайдера-сервиса для «быстрого» развертывания eduroam и распространения лучших практик внедрения и применения сервиса.

Разработанное проектное решение для обновленной архитектуры российского сегмента eduroam (рис. 6) предполагает упрощение общей схемы при одновременном повышении уровня отказоустойчивости сервиса за счет развертывания двух физически разнесенных равноправных RADIUS-серверов на базе ПО с открытым исходным кодом FreeRADIUS в качестве прокси-серверов национального уровня.

Настройка FreeRADIUS будет выполняться с использованием директивы include, позволяющей независимо хранить и обновлять конфигурационные файлы, относящиеся к описанию технических настроек сервиса для конкретной организации.

Конфигурационные файлы предлагается хранить на выделенном сервере настроек в широко применяемой на практике системе управления репозиториями gitlab, что позволит отслеживать версионность вносимых в конфигурацию изменений, вызывать методы проверки их корректности, и синхронизировать настройки серверов FLRS, например, с использованием утилиты rsync.

Балансировку нагрузки между серверами предполагается реализовать на основе программных решений HAProxy/Keepalived, а для защиты от DDoS-атак и деградации сервиса использовать решения, эксплуатируемые в НИКС.

В качестве основных мероприятий, нацеленных на дальнейшее развитие и популяризацию сервисов федеративной аутентификации и мобильности (в том числе, eduroam и eduGAIN) рассматриваются:

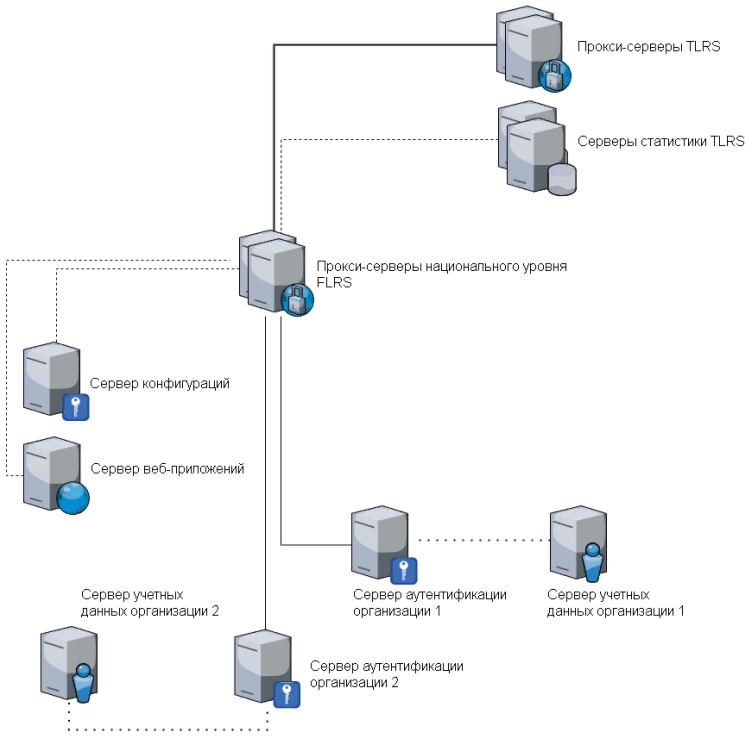


Рис. 6. Усовершенствованная архитектура российского сегмента eduroam

- расширение пользовательской базы сервисов в стране;
- содействие в продвижении сервисов удостоверяющих систем сети Интернет в научных и образовательных организациях России;
- проведение тематических конференций, семинаров, вебинаров с участием действующих и потенциальных пользователей;
- распространение лучших практик использования служб и сервисов удостоверяющих систем в научных и образовательных организациях;
- выработку методик и технических решений по максимально оперативному внедрению элементов удостоверяющих систем в научных и образовательных организациях;
- взаимодействие с общественными организациями и экспертным сообществом в целях улучшения качества предоставления сервисов.

В целом, накопленный объем компетенций позволяет вовлеченным в реализацию проекта специалистам МСЦ РАН осуществлять консультации и оказывать адресную организационную и техническую поддержку представителям организаций по самому широкому кругу вопросов, связанных с развертыванием на их площадках сервисов eduroam и eduGAIN.

В качестве отдельного, представляющегося перспективным направления развития проекта eduroam может рассматриваться возможность налаживания взаимодействия с "традиционными" операторами связи в части развертывания сервиса в публичных местах.

В связи с этим можно заметить, что на начальных этапах реализации проекта зоны доступа eduroam преимущественно размещались на территории научных и образовательных организаций, а с его развитием зоны стали массово разворачиваться и в других публичных местах: музеях, библиотеках, театрах, аэропортах и вокзалах, кафе.

Заключение

Одним из очевидных положительных последствий реализуемых в настоящее время процессов интеграции российских отраслевых научно-образовательных сетей RUNNet и RASNet стало объединение их организационного и технического потенциала, создавшее предпосылки для ускоренного развития и внедрения в сети специализированных прикладных ИКТ-сервисов для сферы науки и образования страны.

Примером успешной многолетней коллаборации мировых научно-образовательных сетей является создание, систематическое развитие и все более широкое распространение сервиса бесплатного роуминга в беспроводных сетях eduroam.

Описанные в статье мероприятия по поддержанию текущего функционирования и совершенствованию российского сегмента eduroam, как ожидается, позволят повысить степень участия российских организаций в проекте, расширить территорию охвата, предоставить организациям централизованное решение по статистическому учету и контролю использования сервиса в качестве одного из показателей, характеризующего уровень научной и образовательной мобильности.

Статья подготовлена в рамках государственного задания ФГУ ФНЦ НИИСИ РАН на 2019 год (тема №0065-2019-0014).

Issues of enhancement the Russian segment of roaming service in wireless networks eduroam in the context of the integration of research and education networks RUNNet and RASNet

A.G. Abramov, I.V. Vasilyev, Yu.N. Morin, A.P. Ovsyannikov, V.A. Porkhachev

Abstract: The paper provides an overview of technical solutions and organizational aspects that ensure the long-term successful functioning of the international roaming service in wireless networks called eduroam. The current architecture of the Russian segment of the service is presented, events aimed at enhancement it, implementation of applied services related to eduroam in the interests of Russian research and education organizations, increasing the level of participation in international projects based on federated authentication concepts in the context of ongoing integration of industry-specific R&E networks RUNNet and RASNet are being discussed.

Keywords: eduroam, federated authentication, Identity Federation, wireless networks, Wi-Fi, RADIUS, national research and education networks, GÉANT, NREN, RUNNet, RASNet

Литература

1. C. Allocchio, L. Balint, V. Berkhout, J. Bersee, Y. Izhvanov et al. A History of international research networking: the people who made it happen. N.Y.: Wiley-VCH, 2010. 317 p.

2. А.Г. Абрамов, А.В. Евсеев. RUNNet как национальная научно-образовательная сеть России: цели, основные задачи, телекоммуникационная инфраструктура и сервисы // Информатизация образования и науки. 2018. №4(40). С. 3–15.

3. Официальный сайт проекта eduroam [Электронный ресурс]. – Режим доступа: https://eduroam.org.

4. M. Milinović, S. Winter. eduroam Policy Service Definition. GÉANT. 2012 [Электронный ресурс]. – Режим доступа: https://www.eduroam.org/wp-content/uploads/2016/05/GN3-12-192_eduroam-policy-service-definition_ver28_26072012.pdf

5. А.П. Овсянников, Г.И. Савин, Б.М. Шабанов. Удостоверяющие федерации научно-образовательных сетей // Программные продукты и системы. 2012. № 4. С. 3–7.

6. А.П. Овсянников, Т.В. Овсянникова, С.А. Овчаренко. Механизм прав на основе групп пользователей в eduroam - федеративной системе управления доступом к сетевым ресурсам научно- образовательных сетей // Программные продукты и системы. – 2012, № 4, с. 10-18

7. А.Г. Абрамов, И.В. Васильев, В.А. Порхачёв. Развитие инфраструктуры аутентификации и авторизации для удостоверяющей федерации в рамках проектов eduGAIN и eduroam на базе сети RUNNet // ИТНОУ: Информационные технологии в науке, образовании и управлении. 2017. №4. С. 56–64.

8. Д.В. Вершинин, Ю.Н. Морин, А.П. Овсянников, Б.М. Шабанов. О реализации российского сегмента eduroam // Труды НИИСИ РАН. 2017. Т.7. №4. С. 162–167.

9. Официальный сайт проекта eduroam Россия [Электронный ресурс]. – Режим доступа: https://eduroam.ru.

10. Официальный сайт проекта eduGAIN [Электронный ресурс]. – Режим доступа: https://edugain.org.

11. А.Г. Абрамов, А.В. Евсеев. Концептуальные аспекты создания в Российской Федерации национальной исследовательской компьютерной сети нового поколения // Информационные технологии. 2019. Т. 25. №12 (принято к печати).

12. Г.И. Савин, Б.М. Шабанов, А.В. Баранов, А.А. Гончар, А.П. Овсянников. Об использовании федеральной научной телекоммуникационной инфраструктуры для суперкомпьютерных вычислений // Суперкомпьютерные дни в России: Труды международной конференции. 23-24 сентября 2019 г., г. Москва / Под. ред. Вл.В. Воеводина – М.: МАКС Пресс, 2019. С. 101-112.

Влияние неоднородности свойств пласта по толщине на эффективность разработки с применением внутрипластового горения

И.В. Афанаскин[1], М.Ю. Ахапкин[2], А.А. Глушаков[3], А.В. Королев[4], А.С. Кундин[5], В.А. Юдин[6]

[1]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, ivan@afanaskin.ru;
[2]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, m.y.ahapkin@niisi.ras.ru;
[3]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, stormwww@yandex.ru;
[4]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, alexandre.korolev@mail.ru;
[5]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, ask8@email.com;
[6]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, yudinval@yandex.ru

Аннотация. В работе кратко описана неизотермическая модель многокомпонентной фильтрации нефти, газа и воды с химическими реакциями. Приводятся результаты численных экспериментов по изучению влияния неоднородности свойств пласта по толщине на эффективность разработки с применением внутрипластового горения. Для случая однородного пласта рассмотрены варианты с закачкой в пласт холодной воды, горячей воды, горячего инертного газа и горячего обогащенного кислородом воздуха. Рассмотрено 10 вариантов с неоднородным строением пласта: расположение одного высокопроницаемого пропластка в кровле или в подошве или в средней части пласта, расположение двух высокопроницаемых пропластков в кровле и в подошве пласта, расположение трех высокопроницаемых пропластков в кровле, в подошве и в средней части пласта. Проницаемости высокопроницаемых пропластков увеличены в 10 и в 100 раз по сравнению с основным объёмом пласта. Общая проводимость пласта при этом не изменяется.

Ключевые слова: высоковязкие нефти, внутрипластовое горение, закачка воздуха, математическое моделирование.

1 Введение

Доля благоприятных для извлечения запасов составляет менее половины российских разведанных запасов нефти [3]. Однако, даже благоприятные запасы, к сожалению, характеризуются в России рядом отрицательных особенностей:

- высокой степенью выработки, которая на многих ныне разрабатываемых месторождениях превысила 50%,
- высокой обводненностью продукции скважин - 70% и более.

Обеспеченность отечественной нефтяной промышленности запасами достаточно низка, а качество остаточных запасов ухудшается [3]. Большинство разведанных запасов уже введено в разработку, многие выработаны в значительной степени [3].

Одним из потенциальных источников нефтедобычи являются вязкие нефти. Суммарные запасы тяжелой нефти и природных битумов составляют порядка 13,1% от общего объема разведанных в России запасов нефти (14% всех запасов приходится на тяжёлые нефти, 12% - на высокосернистые, 11% - на высоковязкие) [3].

Несмотря на различие в применяемых классификациях нефтей и их запасов и связанное с этим различие опубликованных оценок, можно утверждать, что добыча вязких нефтей может стать весьма значимым источником в поддержании нефтедобычи России [3]. Объём запасов вязких нефтей, с учётом применения разных классификаций нефтей и разных систем классификации запасов, составляет от 6 до 10 млрд.т [3].

Детальная разведка месторождений вязких нефтей и применение тепловых методов добычи позволяют ожидать добычи дополнительно 3-4 млрд.т [3]. При этом степень выработанности запасов вязких нефтей в России в настоящее время весьма низка, то есть эти запасы могут стать оперативным резервом нефтедобычи на ближайшие 10-20 лет [3].

Основной проблемой для разработки месторождений вязких нефтей является низкая скорость фильтрации по причине высокой вязкости. Использование заводнения не позволяет достичь больших значений

нефтеотдачи. Для всех таких нефтей вязкость существенно снижается с ростом температуры, поэтому основными методами разработки таких месторождений являются тепловые.

Традиционные технологии разработки (заводнение) обеспечивают нефтеотдачу месторождений с вязкими нефтями 0,20-0,25 д.ед. Применение тепловых методов разработки во многих случаях позволяет достичь нефтеотдачи 0,4-0,45 д.ед. [3].

Существует множество разнообразных тепловых методов разработки месторождений высоковязкой нефти. Наиболее эффективными среди них являются способы, связанные с закачкой в пласт воздуха, поскольку они совмещают в себе гидродинамическое, тепловое, химическое и газовое воздействие. Однако эти способы очень сложны в реализации и весьма дороги. Они требуют тщательного контроля процесса разработки и осознанного его регулирования. Данная работа посвящена изучению с помощью численного моделирования влияния неоднородности фильтрационно-емкостных свойств пласта по толщине на эффективность разработки нефтяных месторождений с трудноизвлекаемыми запасами с применением закачки воздуха для организации внутрипластового горения.

2 Математическая модель

Рассмотрим неизотермическую модель многокомпонентной фильтрации нефти, газа и воды в пористой среде с химическими реакциями [1, 6, 7, 9-11]. Поскольку скорость обмена между фазами сильно превосходит скорость фильтрации, примем гипотезу локального термодинамического равновесия.

Выделим четыре фазы:

1. Твердая фаза, например кокс или парафин.
2. Вода – состоит из одного компонента и не смешивается с углеводородными фазами.
3. Нефть - жидкая углеводородная фаза.
4. Газ – газообразная условно углеводородная фаза (кроме углеводородов может содержать кислород, азот, углекислый газ, пары воды и пр.).

Запишем отдельно для каждого компонента закон сохранения массы в молях:

$$\frac{\partial}{\partial t}\left[\phi(1-S_s)N_c\right] = -\nabla\left[\sum_{\alpha=w,o,g}\left(x_{c,\alpha}\xi_\alpha \vec{W}_\alpha\right)\right] - q_c + \sum_{r=1}^{n_r}\left(S_{Prc}-S_{Rrc}\right)R_r,\ c=1,\ldots,n_c^{/}, \quad (1)$$

$$\frac{\partial}{\partial t}\left(\phi S_s N_c\right) = q_{Ri,c},\ c=n_c^{/}+1,\ldots,n_c, \quad (2)$$

где ϕ – открытая пористость, S_s – насыщенность твердой фазой, N_c – молярная плотность, нижний индекс «c» - номер компонента, нижний индекс «α» - номер фазы, $x_{c,\alpha}$ – доля компонента в фазе, ξ_α – молярная плотность фазы, W_α – скорость фильтрации фазы, q_c – плотность стока компонента в скважину, нижний индекс «r» - номер химической реакции, R_r – скорость химической реакции, S_{Prc} – стехиометрические коэффициенты продуктов реакции, S_{Rrc} – стехиометрические коэффициенты реагентов, n_c – количество компонентов, $n_c^{/}$ - количество подвижных (текучих) компонентов.

Запишем обобщенный закон Дарси в качестве закона сохранения количества движения:

$$\vec{W}_\alpha = -\frac{kk_{r\alpha}}{\mu_\alpha}\left(\nabla P_\alpha - \rho_\alpha g \nabla D\right),\ \alpha = w,o,g, \quad (3)$$

где g - гравитационная постоянная; k - абсолютная проницаемость пласта; $k_{r\alpha}$ - относительная проницаемость для фазы α; μ_α и ρ_α - динамическая вязкость и массовая плотность фазы; D - превышение точки пласта над некоторой горизонтальной плоскостью; P_α - давление в фазе α.

Благодаря предположению о локальном термодинамическом равновесии мы можем записать закон сохранения энергии для всей системы. При этом будем учитывать изменение внутренней энергии флюидов и породы за счет конвективного переноса, теплопроводности, химических реакций, скважин, теплообмена с вмещающими пласт породами:

$$\frac{\partial}{\partial t}\left(U_b\right) = -\nabla\left[\sum_{\alpha=w,o,g}\left(H_\alpha \xi_\alpha \vec{W}_\alpha\right)\right] + \nabla\left(K_b \nabla T\right) + \sum_{r=1}^{n_r} H_r R_r - \sum_{\alpha=w,o,g}\left(H_\alpha \xi_\alpha q_\alpha\right) - q_L \quad (4)$$

где U_b – внутренняя энергия элементарного объема, H_α – энтальпия фазы, K_b – теплопроводность элементарного объема, T – температура, H_r – энтальпия реакции, q_α – плотность источника (стока) фазы, q_L – плотность оттока энергии в окружающие продуктивный пласт породы.

Внутренняя энергия элементарного объема вычисляется как взвешенная по доле занимаемого объема сумма внутренних энергий подвижных фаз, твердой фазы и породы. Фазовое состояние определяется с помощью аппарата констант равновесия. Теплопроводность элементарного объема

определяется как взвешенная по доле занимаемого объема сумма теплопроводности подвижных фаз, твердой фазы и породы. В общем случае зависимости коэффициента теплопроводности от пористости, насыщенности, давления и температуры должны быть установлены экспериментально [5]. Скорость химических реакций определяется с помощью формулы Аррениуса. Энтальпия компонентов в твёрдой и жидких фазах определяется через их удельную теплоёмкость и изменение температуры. Аналогично определяется энтальпия породы. Для определения энтальпии компонентов в газовой фазе добавляется ещё теплота парообразования. Энтальпия фазы определяется через концентрацию компонентов в фазе, энтальпию компонентов и их молекулярный вес. Энтальпия воды как функция давления и температуры берётся из стандартных таблиц. Для расчета молярной и массовой плотности фаз используется уравнение Редлиха-Квонга с нулевыми коэффициентами попарного взаимодействия. Капиллярные давления и относительные фазовые проницаемости задаются как функции насыщенности соответствующих фаз. Для расчета фазовой проницаемости для нефти в трехфазном потоке используется вторая модель Стоуна. Начальные условия задаются в явном виде. Зависимости вязкости компонентов в нефтяной фазе от давления заданы с помощью формулы Андраде. Зависимости вязкости компонентов в газовой фазе от давления заданы с помощью степенной функции. Для моделирования скважин применен стандартный аналитический подход с использованием формулы Писмана [12]. Во всех расчетах, проведенных в рамках данной работы, рассматриваются отдельные единичные элементы системы разработки и считается, что линии тока не пересекают границы этих элементов и, соответственно, границы моделей. Перетоки флюидов через горизонтальные и вертикальные границы модели отсутствуют. Считается, что рассматриваемый пласт окружен бесконечными по простиранию и однородными по тепловым свойствам породами. Потери энергии за счет обмена с внешней средой через кровлю и подошву пласта рассчитываются по стандартной аналитической схеме Vinsome-Westerveld [13]. Ввиду симметрии элементов разработки также считается, что перетоки энергии через вертикальные границы модели отсутствуют.

Систему уравнений (1)-(4) аппроксимирована методом конечных разностей полностью неявным способом. Линеаризация проведена методом Ньютона. Значения относительных фазовых проницаемостей и энтальпии фаз снесены вверх по потоку.

При выполнении работы использовалось ПО компании CMG.

3. Численный эксперимент

Изучение с помощью численного моделирования влияния неоднородности фильтрационно-емкостных свойств пласта по его толщине на эффективность разработки нефтяных месторождений с трудноизвлекаемыми запасами путем закачки воздуха на примере внутрипластового горения проводилось для 1/4 части элемента пятиточечной системы разработки. Расчетный элемент пласта в плане представляет квадрат со сторонами 300х300 м. В противоположных углах квадрата расположены две вертикальные скважины, вскрывающие всю толщину пласта, - добывающая и нагнетательная. Толщина пласта 11 м. Количество ячеек по осям XYZ – 15х15х11 шт., размеры ячеек – 20х20х1 м. Глубина залегания пласта 2850 м. Пористость – 0,2 д.ед. Общая проводимость (произведение толщины на абсолютную проницаемость) – 7700 мД*м и остается постоянной, а проницаемость пласта меняется от варианта к варианту. Рассмотрено 11 вариантов строения с разным количеством и расположением высокопроницаемых пропластков. Коэффициент вертикальной анизотропии проницаемости - 0,1 д.ед. Теплофизические параметры горных пород приняты по результатам анализа литературных данных [5]. Объемная теплоемкость пород пласта и вмещающих пород – 2200 кДж/м^3/ºС. Теплопроводность пород пласта и вмещающих пород – 10 кДж/м/час/ºС. Начальная пластовая температура 80 ºС. Начальное пластовое давление 100 бар.

В процессе расчетов использованы следующие компоненты: кислород, углекислый газ, азот, вода и псевдокомпоненты: тяжёлая нефть, лёгкая нефть, кокс. Растворенным в нефти природным газом пренебрегается, поскольку высоковязкие нефти, как правило, содержат мало растворенного газа. На начальный момент времени углеводородный компонент в пласте представлен только фракцией тяжёлой нефти. Использованные в расчётах относительные фазовые проницаемости приведены на рис. 1 и 2. Газ становится подвижным при газонасыщенности более 0,05 д.ед., но в интервале газонасыщенности 0,0-0,05 д.ед. относительная фазовая проницаемость по нефти падает.

Капиллярное давление не учитывается. На рис. 3 показана зависимость вязкости фракции тяжёлой нефти в жидкой фазе от температуры.

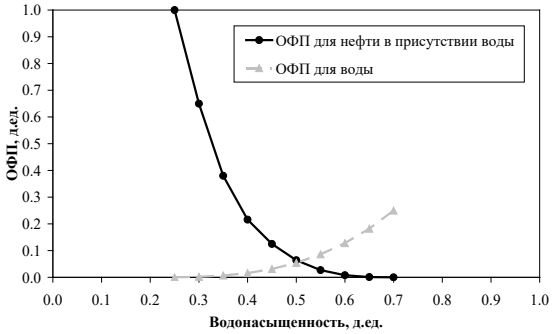


Рис. 1. Кривые ОФП (относительных фазовых проницаемостей) в системе «вода-нефть»

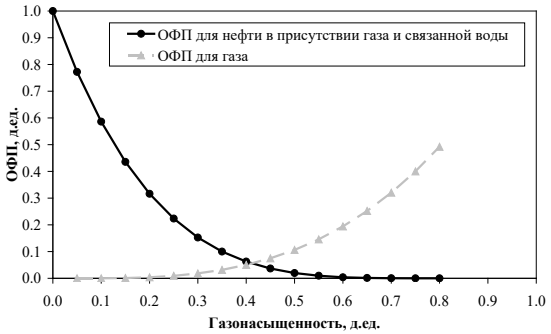


Рис. 2. Кривые ОФП (относительных фазовых проницаемостей) в системе «нефть-газ»

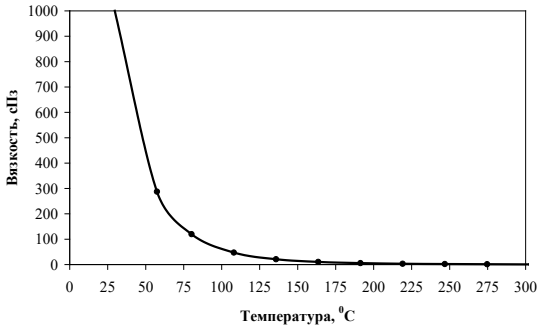


Рис. 3. Зависимость вязкости тяжёлой нефти в жидком состоянии от температуры

После анализа литературных данных [4] модель химических реакций была взята из работы [9] со следующими реакциями:

1. горение тяжёлой нефтяной фракции $HEAVY+18{,}5O_2 \rightarrow 12CO_2+13H_2O$,
2. горение лёгкой нефтяной фракции $LIGHT+5O_2 \rightarrow 3CO_2+4H_2O$,
3. пиролиз тяжёлой нефтяной фракции $HEAVY \rightarrow 2LIGHT+4{,}67COKE$,
4. горение кокса $COKE+1{,}25O_2 \rightarrow CO_2+0{,}5H_2O$,

где HEAVY – фракция тяжёлой нефти, LIGHT – фракция лёгкой нефти, COKE – кокс.

Параметры химических реакций указаны в табл. 1.

Свойства флюидов взяты из работы [9]. Свойства фракций нефти представлены в табл. 2.

Молекулярная масса кокса принята 13 а.е.м. Плотность кокса при температуре 15 ^{0}C и давлении 1 атм. равна 1200 кг/м^3.

Таблица 1. Параметры химических реакций

	Скорость реакции, кг-моль/сут	Константа энергии активации, кДж/кг-моль	Энтальпия реакции, кДж/кг-моль
Реакция 1	$0{,}45 \cdot 10^6$	$33{,}3 \cdot 10^3$	$8{,}18 \cdot 10^6$
Реакция 2	$0{,}45 \cdot 10^6$	$33{,}3 \cdot 10^3$	$2{,}22 \cdot 10^6$
Реакция 3	$0{,}135 \cdot 10^6$	$28{,}8 \cdot 10^3$	$4{,}69 \cdot 10^4$
Реакция 4	$0{,}45 \cdot 10^6$	$23{,}4 \cdot 10^3$	$0{,}54 \cdot 10^6$

Таблица 2. Свойства фракций нефти

	Фракция тяжелой нефти	Фракция легкой нефти
Молекулярная масса, а.е.м.	170	44
Критическое давление, атм.	18,5	43,1
Критическая температура, ^{0}C	897	735
Плотность, кг/м^3	при 1 атм. и 0 ^{0}C равна 853	при 70 атм. и 0 ^{0}C равна 747
Сжимаемость, 1/атм.	$1{,}0*10^{-5}$	$2{,}2*10^{-4}$
Коэффициент температурного расширения, 1/К	$3{,}0*10^{-4}$	$6{,}2*10^{-4}$
Теплота парообразования, кДж/кг	615	213

Моделировалось сухое внутрипластовое горение. Рассмотрены варианты с закачкой воды, подогретой воды, инертного (не растворяющегося и не вступающего в реакции) газа и воздуха. В случае воздуха в нагнетательную скважину закачивается обогащённый кислородом до 40 массовых % воздух, поскольку это существенно повышает эффективность работ [2]. Температура закачиваемого воздуха на забое нагнетательной скважины - 100 ^{0}C. Нагнетательная скважина работает с постоянным расходом газа 28 тыс.нм^3/сут и максимальным забойным давлением 150 бар. Закачка воды идет при постоянном забойном давлении 150 бар. Забойное давление в добывающей скважине - 50 бар. В качестве технологического ограничения из соображения взрывобезопасности принято ограничение мольной доли кислорода 5 % в продукции добывающей скважины (ограничение рассчитано по [8]). При достижении ограничения участок перфорации с наихудшим значением закрывался и счет продолжался дальше. В качестве экономических ограничений на добывающей скважине принято: обводненность 98 %, газонефтяной фактор более 5 тыс.нм^3/м^3, дебит нефти 1 м^3/сут. Максимальный срок разработки принят равным 30 годам. Результаты расчетов приведены в табл. 3, 4 и показаны на рис. 4-13. В табл. 3 и 4 полужирным подчеркнутым выделен вариант с максимальной нефтеотдачей, а полужирным курсивом - с минимальной. Серой заливкой показана причина остановки скважин для вариантов с учетом экономических ограничений.

Худшим вариантом (дающим наименьшую нефтеотдачу) является вариант 3 с закачкой инертного газа при температуре 100 °C в однородном пласте - 0,093 д.ед. за 30 лет без экономических ограничений и 0,032 д.ед. за 1,2 года с экономическими ограничениями.

Затем для однородного пласта следуют варианты 1 и 2 с закачкой «холодной» воды при температуре 30 °C (нефтеотдача 0,236 д.ед. за 30 лет независимо от экономических ограничений) и с закачкой «горячей» воды при температуре 100 °C (нефтеотдача 0,241 д.ед. за 30 лет независимо от экономических ограничений).

В однородном пласте закачка обогащенного кислородом воздуха (вариант 4) дает повышение нефтеотдачи по сравнению с заводнением до 0,504 д.ед. за 25,2 года без экономических ограничений и до 0,300 д.ед. за 12,4 года с учетом этих ограничений.

Необходимо отметить, что с учетом экономических ограничений при максимальном сроке разработки 30 лет во всех вариантах, где разработка была закончена ранее, добывающие скважины были остановлены из-за достижения максимального газонефтяного фактора 5 тыс.нм^3/м^3.

Без экономических ограничений все варианты с закачкой воздуха были в разработке менее 30 лет, добывающие скважины были остановлены по причине достижения ограничения мольной доли кислорода 5 % в продукции по всем участкам перфорации.

Вариант 5 - закачка обогащенного кислородом воздуха при температуре 100 °C на забое нагнетательной скважины в пласт, в середине которого существует один пропласток толщиной 1 м с проницаемостью в 10 раз выше проницаемости остального пласта, дает нефтеотдачу 0,492 д.ед. за 16,2 года без учета экономических ограничений и 0,420 д.ед. за 11,7 лет с учетом этих ограничений.

Вариант 6 - закачка обогащенного кислородом воздуха при температуре 100 °C на забое нагнетательной скважины в пласт, в кровле которого существует один пропласток толщиной 1 м с проницаемостью в 10 раз выше проницаемости остального пласта, дает нефтеотдачу 0,268 д.ед. за 13,9 года без учета экономических ограничений и 0,125 д.ед. за 4,7 года с учетом этих ограничений.

Вариант 7 - закачка обогащенного кислородом воздуха при температуре 100 °C на забое нагнетательной скважины в пласт, в подошве которого существует один пропласток толщиной 1 м с проницаемостью в 10 раз выше проницаемости остального пласта дает, нефтеотдачу 0,530 д.ед. за 22,0 года без учета экономических ограничений и 0,453 д.ед. за 16,8 года с учетом этих ограничений. Это вариант с максимальной нефтеотдачей без экономических ограничений.

Вариант 8 - закачка обогащенного кислородом воздуха при температуре 100 °C на забое нагнетательной скважины в пласт, в кровле и подошве которого существуют пропластки толщиной 1 м с проницаемостью в 10 раз выше проницаемости остального пласта, дает нефтеотдачу 0,464 д.ед. за 26,2 года без учета экономических ограничений и 0,149 д.ед. за 6,8 года с учетом этих ограничений.

Таблица 3. Влияние неоднородности свойств пласта по толщине
на эффективность внутрипластового горения, варианты 1-7

Вариант	1	2	3	4	5	6	7
Закачиваемый агент	Вода при 30 ºС	Вода при 100 ºС	Азот при 100 ºС	Обогащенный воздух при 100 ºС			
Расположение высокопроницаемого пропластка (пропластков)	Одно-родный пласт	Одно-родный пласт	Одно-родный пласт	Одно-родный пласт	Один пропласток в середине пласта	Один пропласток в кровле пласта	Один пропласток в подошве пласта
Соотношение проницаемости высокопроницаемого пропластка (пропластков) и остального пласта	-	-	-	-	10	10	10
Показатели разработки, рассчитанные без учета экономических ограничений							
Дебит нефти на конец срока разработки, $м^3$/сут	1,7	1,8	0,5	1,4	1,0	2,7	**1,4**
Обводненность на конец срока разработки, %	96,2	96,5	-	85,6	84,0	46,6	**80,3**
Газонефтяной фактор на конец срока разработки, тыс.$нм^3/м^3$	-	-	59	19	29	10	**19**
Температура на забое добывающей скважины на конец срока разработки, ºС	-	-	-	165,1	174,9	251,4	**179,9**
Отношение накопленной добычи легкой и тяжелой нефти, т/т	0	0	0	2,2	2,6	2,4	**2,5**
Срок разработки, годы	30,0	30,0	30,0	25,2	16,2	13,9	**22,0**
Нефтеотдача, д.ед.	0,236	0,241	0,093	0,504	0,492	0,268	**0,530**
Показатели разработки, рассчитанные с учетом экономических ограничений							
Дебит нефти на конец срока разработки, $м^3$/сут	1,7	1,8	5,8	5,6	5,3	5,1	5,2
Обводненность на конец срока разработки, %	96,2	96,5	-	26,8	47,2	47,5	61,9
Газонефтяной фактор на конец срока разработки, тыс.$нм^3/м^3$	-	-	5	5	5	5	5
Температура на забое добывающей скважины на конец срока разработки, ºС	-	-	-	80,0	171,8	80,0	150,9
Отношение накопленной добычи легкой и тяжелой нефти, т/т	0	0	0	0,9	2,0	0,8	2,1
Срок разработки, годы	30,0	30,0	1,2	12,4	11,7	4,7	16,8
Нефтеотдача, д.ед.	0,236	0,241	0,032	0,300	0,420	0,125	0,453

Таблица 4. Влияние неоднородности свойств пласта по толщине на эффективность внутрипластового горения, варианты 8-14

Вариант	8	9	10	11	12	13	14
Закачиваемый агент	Обогащенный воздух при 100 ºС						
Расположение высокопроницаемого пропластка (пропластков)	Два пропл. в кровле и подошве пласта	Три пропл. в кровле, в серед. и в подошве пласта	Один пропласток в середине пласта	Один пропласток в кровле пласта	Один пропласток в подошве пласта	Два пропл. в кровле и подошве пласта	Три пропл. в кровле, в серед. и в подошве пласта
Соотношение проницаемости высокопроницаемого пропластка (пропластков) и остального пласта	10	10	100	100	100	100	100
Показатели разработки, рассчитанные без учета экономических ограничений							
Дебит нефти на конец срока разработки, $м^3$/сут	1,6	1,3	1,0	***19,9***	0,8	0,6	1,0
Обводненность на конец срока разработки, %	87,0	72,2	75,5	***11,9***	80,7	89,4	84,9
Газонефтяной фактор на конец срока разработки, тыс.$нм^3/м^3$	17	20	25	***2***	35	47	39
Температура на забое доб. скв. на конец срока разработки, ºС	201,0	191,5	172,0	***261,2***	228,0	231,7	214,2
Отношение накопленной добычи легкой и тяжелой нефти, т/т	3,6	2,4	2,3	***0,9***	3,4	4,3	4,0
Срок разработки, годы	26,2	16,7	8,3	***5,3***	14,5	21,4	21,9
Нефтеотдача, д.ед.	0,464	0,509	0,224	***0,098***	0,246	0,430	0,440
Показатели разработки, рассчитанные с учетом экономических ограничений							
Дебит нефти на конец срока разработки, $м^3$/сут	5,6	**5,3**	5,5	***4,6***	5,4	5,6	5,9
Обводненность на конец срока разработки, %	52,7	**51,0**	54,7	***14,8***	56,6	55,1	60,2
Газонефтяной фактор на конец срока разработки, тыс.$нм^3/м^3$	5	**5**	5	***5***	5	5	5
Температура на забое доб. скв. на конец срока разработки, ºС	80,0	**178,8**	80,0	***80,0***	80,0	80,0	80,0
Отношение накопленной добычи легкой и тяжелой нефти, т/т	0,7	**2,2**	1,7	***0,01***	0,8	1,1	1,3
Срок разработки, годы	6,8	**13,9**	5,1	***2,0***	5,4	7,9	8,8
Нефтеотдача, д.ед.	0,149	**0,460**	0,172	***0,028***	0,113	0,189	0,184

Вариант 9 - закачка обогащенного кислородом воздуха при температуре 100 °С на забое нагнетательной скважины в пласт, в кровле, подошве и середине которого существуют пропластки толщиной 1 м с проницаемостью в 10 раз выше проницаемости остального пласта, дает нефтеотдачу 0,509 д.ед. за 16,7 года без учета экономических ограничений и 0,460 д.ед. за 13,9 лет с учетом этих ограничений. Это вариант с максимальной нефтеотдачей с учетом экономических ограничений.

Вариант 10 - закачка обогащенного кислородом воздуха при температуре 100 °С на забое нагнетательной скважины в пласт, в середине которого существует один пропласток толщиной 1 м с проницаемостью в 100 раз выше проницаемости остального пласта, дает нефтеотдачу 0,224 д.ед. за 8,3 года без учета экономических ограничений и 0,172 д.ед. за 5,1 года с учетом этих ограничений.

Вариант 11 - закачка обогащенного кислородом воздуха при температуре 100 °С на забое нагнетательной скважины в пласт, в кровле которого существует один пропласток толщиной 1 м с проницаемостью в 100 раз выше проницаемости остального пласта, дает нефтеотдачу 0,098 д.ед. за 5,3 года без учета экономических ограничений и 0,028 д.ед. за 2,0 года с учетом этих ограничений. Это вариант с минимальной нефтеотдачей как с учетом экономических ограничений, так и без их учета.

Вариант 12 - закачка обогащенного кислородом воздуха при температуре 100 °С на забое нагнетательной скважины в пласт, в подошве которого существует один пропласток толщиной 1 м с проницаемостью в 100 раз выше проницаемости остального пласта, дает нефтеотдачу 0,246 д.ед. за 14,5 года без учета экономических ограничений и 0,113 д.ед. за 5,4 года с учетом этих ограничений.

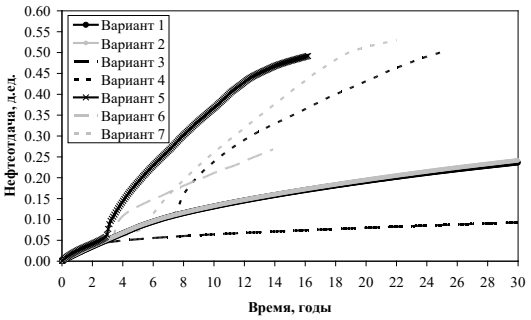


Рис. 4. Зависимость нефтеотдачи от времени, варианты 1-7

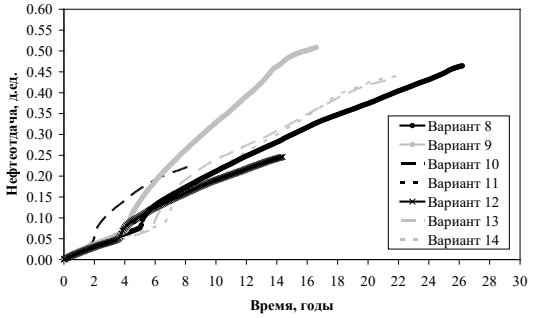


Рис. 5. Зависимость нефтеотдачи от времени, варианты 8-14

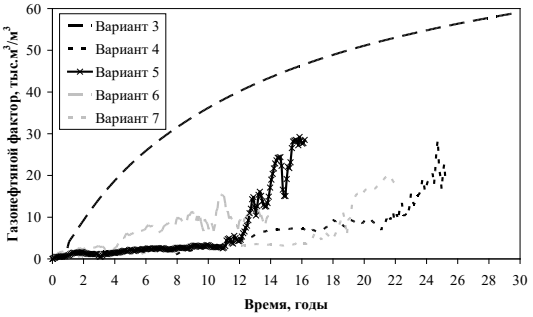


Рис. 6. Зависимость газонефтяного фактора от времени, варианты 3-7

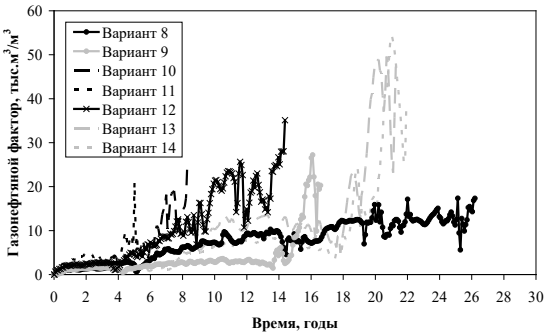


Рис. 7. Зависимость газонефтяного фактора от времени, варианты 8-14

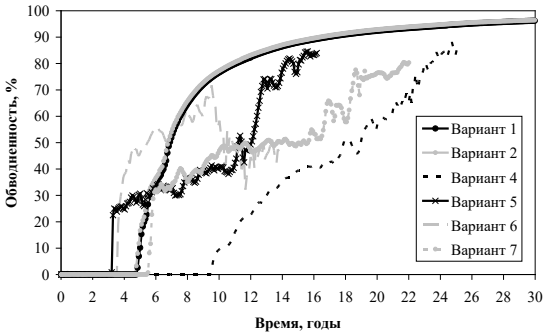


Рис. 8. Зависимость обводненности от времени, варианты 1-2, 4-7

Вариант 13 - закачка обогащенного кислородом воздуха при температуре 100 °С на

забое нагнетательной скважины в пласт, в кровле и подошве которого существуют пропластки толщиной 1 м с проницаемостью в 100 раз выше проницаемости остального пласта, дает нефтеотдачу 0,430 д.ед. за 21,4 года без учета экономических ограничений и 0,189 д.ед. за 7,9 года с учетом этих ограничений.

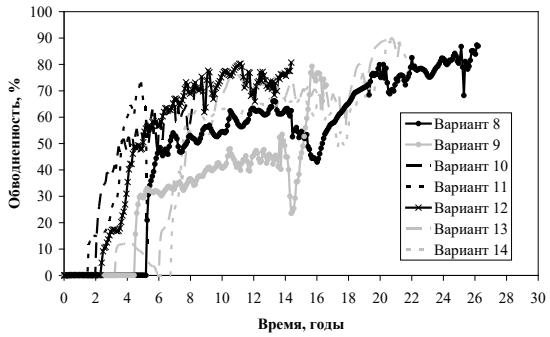


Рис. 9. Зависимость обводненности от времени, варианты 8-14

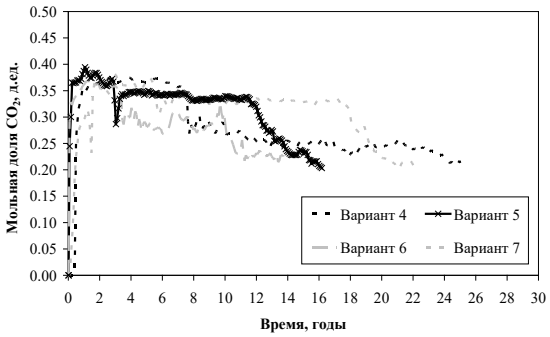


Рис. 10. Зависимость мольной доли CO_2 от времени, варианты 4-7

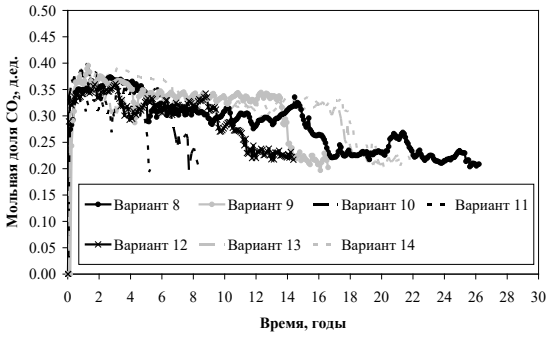


Рис. 11. Зависимость мольной доли CO_2 от времени, варианты 8-14

Вариант 14 - закачка обогащенного кислородом воздуха при температуре 100 °С на забое нагнетательной скважины в пласт, в кровле, подошве и середине которого существуют пропластки толщиной 1 м с проницаемостью в 100 раз выше проницаемости остального пласта, дает нефтеотдачу 0,440 д.ед. за 21,9 года без учета экономических ограничений и 0,184 д.ед. за 8,8 года с учетом этих ограничений.

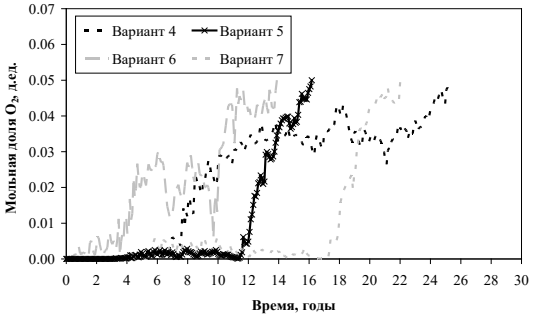


Рис. 12. Зависимость мольной доли O_2 от времени, варианты 4-7

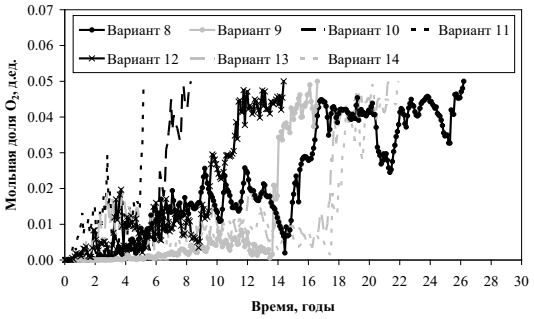


Рис. 13. Зависимость мольной доли O_2 от времени, варианты 8-14

4 Заключение

С помощью численного моделирования изучено влияния неоднородности фильтрационно-емкостных свойств пласта по толщине на эффективность разработки нефтяных месторождений с трудноизвлекаемыми запасами с применением внутрипластового горения.

Описана неизотермическая модель многокомпонентной фильтрации нефти, газа и воды с химическими реакциями.

Рассмотрены варианты с закачкой холодной воды, горячей воды, горячего инертного газа и горячего обогащенного кислородом воздуха (внутрипластовое горение) в однородный пласт с высоковязкой нефтью. Наименее эффективным вариантом оказалась закачка инертного газа, а наиболее эффективным - внутрипластовое горение. Оно позволяет увеличить нефтеотдачу на 0,263 д.ед. по сравнению с заводнением и имеет меньший срок разработки.

Рассмотрено 10 вариантов внутрипластового горения в неоднородном по толщине пласте.

При наличии в кровле пласта пропластка с

проницаемостью в 10 раз выше проницаемости окружающих пород нефтеотдача падает в 1,88 раза, а в середине пласта - только в 1,02 раза. При наличии такого пропластка в подошве пласта нефтеотдача возрастает в 1,05 раза. Два пропластка в кровле и подошве немного снижают нефтеотдачу (в 1,09 раза), а три пропластка в кровле, подошве и в середине пласта увеличивают нефтеотдачу в 1,01 раза.

При наличии в кровле пласта пропластка с проницаемостью в 100 раз выше проницаемости окружающих пород нефтеотдача падает в 5,14 раза, а в середине пласта - в 2,25 раза. При наличии такого пропластка в подошве пласта нефтеотдача падает в 2,05 раза. Два пропластка в кровле и подошве несколько снижают нефтеотдачу (в 1,17 раза), а три пропластка в кровле, подошве и в середине пласта снижают нефтеотдачу в 1,15 раза.

То есть, появление высокопроницаемых пропластков всегда снижает срок разработки. Такое влияние неоднородности по толщине на нефтеотдачу связано с гравитационным разделением фаз, выражающемся в активном всплывании в пласте закачиваемого воздуха и газов горения, что вызывает преждевременный прорыв кислорода в добывающую скважину и ее остановку из соображений безопасности работ.

Следует понимать, что даже небольшое изменение нефтеотдачи в масштабах месторождения приводит к существенному увеличению добычи нефти и продлению жизни месторождения.

Данные результаты получены без учета экономических ограничений. В действительности показатели привязаны к цене нефти и особенностям эксплуатации скважин в конкретном регионе. В работе приведены также расчеты с учетом средних по стране экономических ограничений - обводненность 98 %, газонефтяной фактор более 5 тыс.нм^3/м^3, дебит нефти 1 м^3/сут.

Работа выполнена при поддержке программы Президиума РАН № 2 (I27), тема (проект) «Изучение с помощью математического моделирования некоторых особенностей фильтрации флюидов в нефтяных коллекторах с трудноизвлекаемыми запасами (на примере баженовской свиты)» № 0065-2019-0023.

Efficiency of Viscous Oil Production Using In-Situ Combustion. Reservoir Heterogeneity Influence

I.V. Afanaskin, M.Yu. Ahapkin, A.A. Glushakov, A.V. Korolev, A.S. Kundin, V.A. Yudin

Abstract. Non-isothermal multiphase compositional flow model with chemical reactions is described. Efficiency of viscous oil production using in-situ combustion is studied. Numerical calculations evaluate results of reservoir heterogeneity influence. For the homogeneous reservoir some cases are described for various production technologies with injection of cold water, hot water, hot inert gas and oxygen enriched hot air. Ten cases for heterogeneous reservoir are studied for different high permeability layer location: top of formation, bottom of formation, in the middle of formation; two high permeability layers located at formation top and formation bottom; three high permeability layers located at formation top, formation bottom and in the middle of formation. Permeability of individual layers was increased ten and hundred times compared with the main formation permeability values but total formation transmissibility was not changed.

Key words: high viscosity oils, in-situ combustion, air injection, mathematical modeling.

Литература

1. Алишаев М.Г., Розенберг М.Д., Теслюк Е.В. Неизотермическая фильтрация при разработке нефтяных месторождений. М.: Недра, 1985. 271 с.

2. Афанаскин И.В. Повышение технологической эффективности метода направленной закачки воздуха в нефтяные пласты на основе численного моделирования и результатов гидродинамических исследований скважин // Дисс. на соиск. уч. степ. к.т.н. М.: ОАО «ВНИИнефть», 2013. – 273 с.

3. Бетелин В.Б., Юдин В.А., Афанаскин И.В., Вольпин С.Г., Кац Р.М., Королёв А.В. Создание отечественного термогидросимулятора – необходимый этап освоения нетрадиционных залежей углеводородов России. – М.: ФГУ ФНЦ НИИСИ РАН, 2015. – 206 с.

4. Бетелин В.Б., Юдин В.А., Королёв А.В., Афанаскин И.В., Вольпин С.Г. Моделирование химических реакций окисления и горения углеводородов при добыче нефти с закачкой в пласт воздуха. – М.: ФГУ ФНЦ НИИСИ РАН, 2015. – 161 с.

5. Бетелин В.Б., Юдин В.А., Афанаскин И.В., Вольпин С.Г., Королёв А.В. Термические преобразования скелета пород при добыче нефти с закачкой в пласт воздуха. – М.: ФГУ ФНЦ НИИСИ РАН, 2015. – 204 с.

6. Бурже Ж., Сурио П., Комбарну М. Термические методы повышения нефтеотдачи пластов. М.: Недра, 1988. 424 с.

7. Методические указания по созданию постоянно действующих геолого-технологических моделей нефтяных и газонефтяных месторождений (Часть 2. Фильтрационные модели). М.: ОАО «ВНИИОЭНГ», 2003. - 228 с.

8. РД 39-9-191-79. Методическое руководство по проектированию и применению внутрипластового горения в разработке нефтяных месторождений. Дополнение. Инструкция по инициированию внутрипластового процесса горения для различных геолого-физических условий.

9. Coats K.H. In-situ combustion model // SPE of AIM, 1980. № 8394. P. 533-554.

10. Gottfried B.S. A Mathematical Model of Thermal Oil Recovery in Linear Systems // SPE 1117, 1965. - 15 P.

11. In-Situ Combustion Handbook - Principles and Practices. Oklahoma: BDM Petroleum Technology, 1999. - 424 P.

12. Peaceman D.W. Interpretation of Well-Block Pressures in Numerical Reservoir Simulation with Nonsquare Grid Blocks and Anisotropic Permeability // SPE Journal. – 1983. – v. 23. - № 3. – PP. 531-543.

13. Vinsome P. K. W. A Simple Method for Predicting Cap and Base Rock Heat Losses in Thermal Reservoir Simulators // The Journal of Canadian Petroleum Technology (JCPT), Montreal. – 1980. - July-Sept. - v. 19. - № 3. – PP. 87-90.

Оценка точности определения положения водонефтяного контакта

Ю.Б. Чен-лен-сон[1]

[1]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, JChenlenson@niisi.ras.ru

Аннотация: В статье анализируется понятие водонефтяного контакта (ВНК), особенности его строения, определяемого свойствами пород, насыщающих флюидов, капиллярных процессов. Точность определения положения ВНК зависит от достоверности информации о продуктивном пласте, а также от точности используемых геофизических методов. С целью повышения точности определения ВНК необходимо комплексировать данные геофизических, гидродинамических и трассерных видов исследования пластов и скважин.

Ключевые слова: водонефтяной контакт, переходная зона, инклинометрия

Водонефтяным контактом (ВНК) в нефтяной залежи, подстилаемой водой, называется условная граница, разделяющая нефте и водонасыщенные зоны пласта. Условной эта граница является, потому, что не является явно выраженной линией, а представляет собой некоторую зону пласта с постепенным переходом от водонасыщенной к нефтенасыщенной зоне. Положение ВНК зависит от пористости и проницаемости коллектора, свойств нефти и воды, фазовых проницаемостей для нефти и воды, капиллярного давления, высоты залежи, степени литологической неоднородности коллектора по разрезу и площади и ряда других факторов.

Точность определения положения ВНК зависит от достоверности информации о продуктивном пласте, а также от точности используемых геофизических методов. Достоверность определения начального положения ВНК и анализа передвижения ВНК по площади повышается при комплексировании данных по капилляриметрическим исследованиям керна (или фазовым проницаемостям), испытаниям скважин и результатам обработки материалов ГИС. При оценке текущего ВНК необходимо комплексировать данные геофизических, гидродинамических и трассерных видов исследования скважин, основанных на закачке в пласт-коллектор радиоактивных изотопов или жидкостей различного химического состава.

В российской нефтяной практике определение положения ВНК довольно вольное, и зависит от нефтяных традиций того, или иного региона, так в Поволжье границей считают глубину с которой начинают получать безводную нефть, а в Западной Сибири появление первых капель нефти. Таким образом, за ВНК принимается, где нижняя, а где верхняя граница переходной зоны. Определяется это, главным образом, по результатам опробования разведочных скважин. Толщина переходной зоны, табл. 1, очень сильно варьирует от месторождения к месторождению, и может меняться в разных частях одного месторождения. Зависит это от физических свойств пород коллектора и флюида. В коллекторах с малой проницаемостью переходная зона, как правило, больше. Это связано, в основном, с величиной поверхностного натяжения и размерами проницаемых каналов. Величина поверхностного натяжения зависит от типа смачиваемости горных пород (гидрофильная или гидрофобная породы). Всё это влияет на величину капиллярных сил. В зарубежной нефтяной практике за нижнюю границу переходной зоны принимают глубину, где капиллярные силы равны нулю. Колебания высоты переходной зоны, от первых метров, до первых десятков метров. Связано это с геологическими условиями формирования залежи. Если общее нефтесодержание невысокое, то переходная зона может охватывать всю толщину пласта. Переходная зона газонефтяного контакта (ГВК), как правило, значительно меньше переходной зоны нефть-вода, потому что разница плотностей газа и нефти выше, чем нефти и воды, а разница величины поверхностного натяжения ниже.

Размеры переходной зоны в вертикальном направлении изменяются в широких пределах. Для высокопроницаемых терригенных кварцевых коллекторов толщина переходной

зоны составляет всего лишь доли метра. Такие отложения характерны, например, для Волго-Уральской нефтегазоносной провинции. А для полимиктовых коллекторов Западной Сибири с большим содержанием глины толщина переходной зоны достигает 20-30 метров.

Определение положения ВНК производится, чаще всего на основании данных геофизических исследований скважин (ГИС), в большинстве случаев используют данные электрометрии, рис.1 [1]. В некоторых случаях, из-за особенностей литологии определение положения ВНК с помощью электрометрии ГИС вызывает затруднения. Связано это, в основном, с присутствием в разрезе железистых минералов (сидерит, пирит) обладающих повышенной проводимостью и понижающих удельное сопротивление пласта.

В таких случаях рекомендуется воспользоваться замерами градиента давления и капиллярных сил при помощи пластоиспытателей. В силу небольшой высоты переходной зоны необходим ряд селективных замеров, которые могут быть обеспечены только с помощью пластоиспытателей на кабеле (MDT, RFT и т.п.). Вообще-то это единственное разумное применение пластоиспытателей в нефтяной практике, попытки получения других параметров пласта (пластовое давление, нефтенасыщенность, проницаемость) не выдерживают никакой критики, в силу специфичного влияния скважинных условий во время замеров, сводящих замеры пластоиспытателями к опробованию.

Для определения нефтенасыщенных зон в заколонном пространстве скважин наиболее часто применяется углеродно-кислородный каротаж (С/О-каротаж). Этот каротаж основывается на определении отношения количества углерода «С» и кислорода «О» в породе пласта. Это отношение характеризует содержание в пласте углеводородов [2].

Определение глубины положения ВНК, границ пластов и положения этих границ в пространстве, производится в процессе бурения скважины при помощи инклинометрии [3].

Инклинометрией скважины называется метод определения ее пространственного положения, ориентируясь на ось. Целью инклинометрических исследований является определение пространственного положения ствола буровой скважины. Результаты инклинометрии применяют для глубины расположения геологических объектов по вертикали (истинная глубина). Такие глубины в том числе используются при построении всевозможных картографических материалов совместно с различными данными геофизических исследований.

Таблица 1. Характеристика ВНК и переходной зоны [1]

Название зоны	**Подзоны**	**Характеристика зоны**
Нефтяная зона	Подзона предельной нефтенасыщенности	Содержит нефть и прочно связанную остаточную воду. Нефтенасыщенность не зависит от гипсометрической отметки пласта.
	Подзона нефтенасыщения	Из этой зоны получают притоки безводной нефти. Нефтенасыщенность возрастает снизу вверх. Она зависит от гипсометрической отметки пласта, что объясняется наличием рыхлосвязанной воды, количество которой убывает сверху вниз.
Переходная зона	Подзона двухфазного притока	Характеризуется наличием подвижной нефти. При опробовании отсюда получают нефть с водой. По нижней части этой подзоны проводят водонефтяной контакт, который должен быть подтвержден притоками нефти (даже с водой).
	Подзона остаточного нефтенасыщения	Эта зона содержит остаточную нефть и воду. При испытании получают воду с пленкой нефти.

Водяная зона		Эта зона характеризуется низкими гипсометрическими отметками. Эта зона насыщена свободной водой. Под воздействием гравитационных сил и градиента давления зона может передвигаться. Так же в зоне может присутствовать остаточная вода (локально сохранившаяся в порах коллектора после того, как он был заполнен нефтью) и кристаллизационная вода (находится в составе молекул минералов).

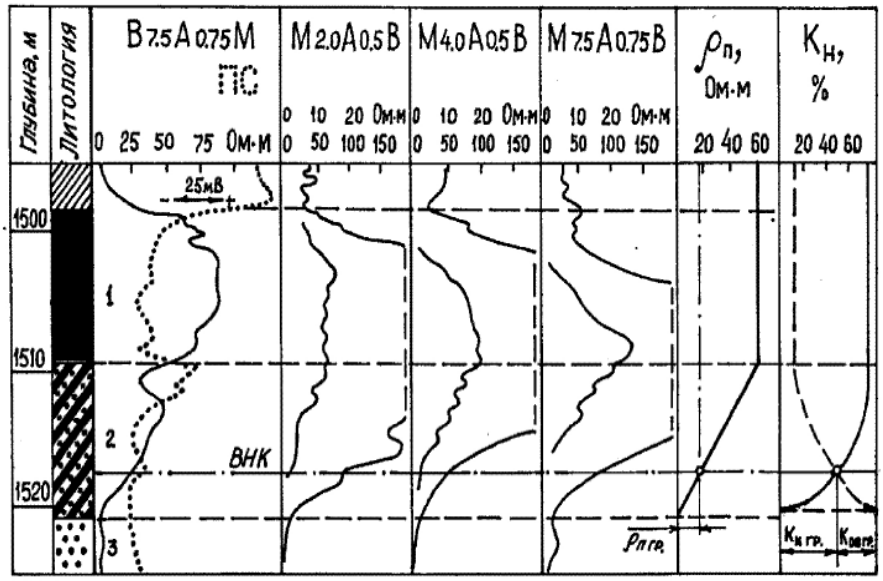


Рис. 1. Определение ВНК при наличии зоны предельной нефтенасыщенности (1), переходной зоны (2) и водоносной части коллектора (3) [1]

При геофизических исследованиях горизонтальных скважин и боковых стволов значимость и информативность инклинометрических измерений существенно возрастает по сравнению с обычными вертикальными скважинами. В основе инклинометрических измерений лежит определение зенитного угла и магнитного азимута. Под понятием зенитный угол понимают угол отклонения оси определенной скважины от вертикали (т.е. угла 90°). Для определения этого угла инклинометрический преобразователь должен быть оснащен гравитационными датчиками, которые позволяют найти направление вектора силы тяжести. Магнитный азимут - это проекция оси скважины на плоскости. При инклинометрии поле получения необходимых данных замеров зенитного угла и магнитного азимута, строится так называемая инклинограмма, при этом принимается во внимание глубина исследуемого ствола скважины. Инклинограмма представляет собой проекцию оси ствола на горизонтальную плоскость и вертикальную проекцию на широтную плоскость, магнитного меридиана и любую другую. При наличии фактических координат скважин, которые находятся в процессе бурения, можно максимально точно установить точки пересечения скважины и различных слоев геологического разреза, что означает правильно установить заданное направление для бурения. Точность определения точек пересечения зависит от точности прибора (инклинометра), интервала между точками записи и методов расчёта инклинограммы. Погрешность инклинометров КИТ, самых распространённых в советское время, составляет до 0.5° по вертикали и 3-4° при определении магнитного азимута, и всё это при отсутствии каких-либо следов металла в скважине. В настоящее время используются более совершенные гироскопические инклинометры, обладающие гораздо меньшей погрешностью, но замеры, проведённые в давно пробуренных скважинах, остаются как есть. На точность определения глубины и пространственного положения оказывают влияние и методы обсчёта инклинометрических данных.

Применяется несколько методов интерпретации инклинометрии.

Тангенциальный метод

В тангенциальном методе скважина представляется прямой линией по всему охваченному замером интервалу. Для расчетов используют азимут ствола скважины в нижней части замеряемого интервала и зенитный угол, рис. 2. Таким образом определяют прямую линию, проходящую через подошву замеряемого интервала и отображающую ствол изучаемой скважины. Тангенциальный метод является наименее точным из всех рассматриваемых в работе. Его можно использовать только в тех случаях, когда участок измерений имеет размеры не более длины самого прибора.

Метод усредненного угла

В методе усредненного угла для расчетов используются соответственно средние значения азимута и зенитных углов, которые были замерены в кровле и подошве интервала измерений, рис. 3. Средние значения по двум точкам приписываются всему интервалу замера. Траектория скважины рассчитывается с использованием тригонометрических формул. Все расчеты можно проделать с помощью обычного калькулятора, поэтому метод хорошо подходит для полевых работ. Метод усредненного угла является существенно более точным по сравнению с тангенциальным методом. Однако он немного уступает в точности методам радиуса кривизны и минимальной кривизны.

Метод радиуса кривизны

Этот метод предполагает построение траектории ствола скважины в виде дуги в вертикальной и горизонтальной проекциях, рис. 4. Для этого используются азимут ствола скважины и зенитный угол, которые были определены в кровле и подошве интервала измерений.

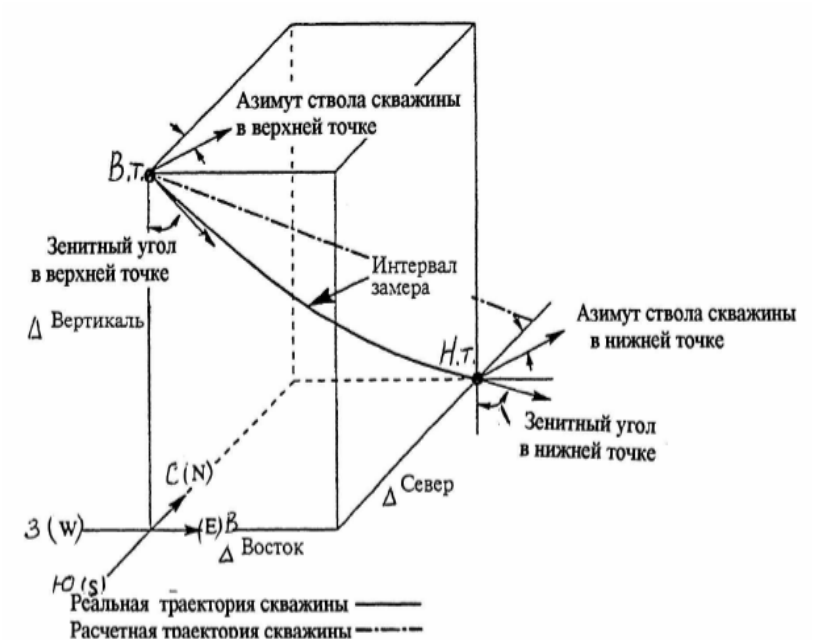


Рис. 2. Схема обработки инклинометрии тангенциальным методом [1]

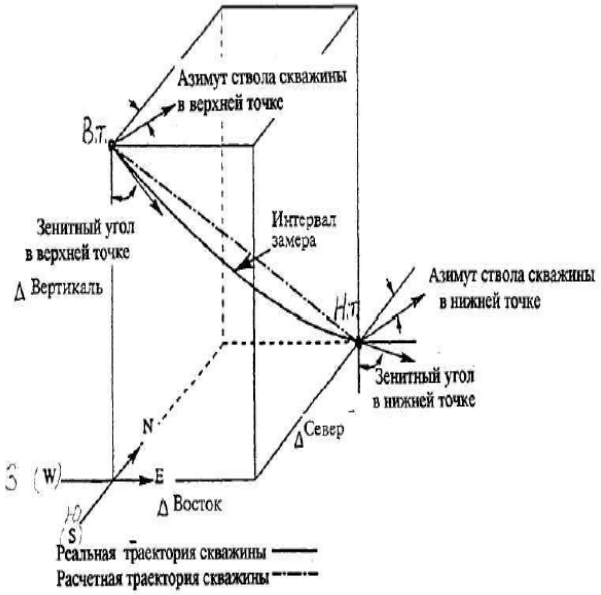


Рис. 3. Схема обработки инклинометрии методом усредненного угла [1]

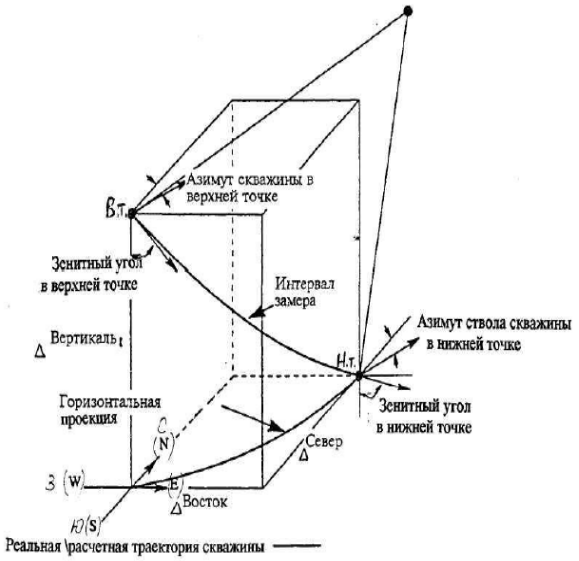


Рис. 4. Схема обработки инклинометрии методом радиуса кривизны [1]

Всё вышеизложенное приведено для того, чтобы показать, какая большая вероятность ошибки в определении глубины того или иного пропластка, той или иной границы, особенно границ контактов (ВНК, ГНК).

Но при этом многие геологи, используют разницу глубины ВНК для обоснования наличия, так называемых, малоамплитудных разломов, которые разбивают месторождение на блоки.

При этом, разломы не создают никаких непроницаемых гидродинамических границ и вообще являются вещью в себе, видимые только по сейсмическим исследованиям и по разнице в глубине контактов. По всей видимости, выделение таких разломов, это просто ошибки в определении глубин этих контактов и неуверенная корреляция сейсмики.

Работа выполнена при поддержке гранта РФФИ 18-07-00676 А.

Oil-water contact location justification accuracy

Yu.B. Chen-len-son[1]

Abstract: Article contains oil-water contact term meaning analysis, its location features, which are determined by rock properties, saturation fluids composition and capillary effects properties. Oil-water contact location determination precision depends on productive formation properties data accuracy, which includes geophysical survey data. Oil-water contact justification accuracy Improvement requires complexion of geophysical, hydrodynamic and tracer well survey data.

Keywords: oil -water contact, transition zone, inclinometry

Литература

1. В.Н. Косков, Б.В. Косков. Геофизические исследования скважин и интерпретация данных ГИС. Пермь, Изд-во Перм. гос. техн. ун-та, 2007.

2. Е.С. Кучурин, В.Л. Глухов, А.Н. Огнев, В.П. Метелев. Состояние, эффективность применения и перспективы развития углеродно-кислородного каротажа для оценки нефтенасыщенности пластов разрабатываемых месторождений. НТВ «Каротажник», 2004, Вып. 1, 114.

3. В.К. Хмелевской, В.И. Костицын. Основы геофизических методов. Пермь, Перм. ун-т, 2010.

Оценка вовлечения в разработку недренируемых керогенсодержащих зон баженовской свиты при моделировании термогазового воздействии

Миронов Д.Т.[1], Ялов П.В. [2]

[1] ФГУ ФНЦ НИИСИ РАН, Москва, Россия, mdt1958@yandex.ru;

[2] ФГУ ФНЦ НИИСИ РАН, Москва, Россия, petryalov@gmail.com

Аннотация: Рассматриваются вопросы математического моделирования трехфазной фильтрации в условиях применения термогазового метода увеличения нефтеотдачи (ТГ МУН). Приведена краткая характеристика отложений баженовской свиты и дано описание основных механизмов воздействия на пласт при реализации ТГ МУН. Подготовлена 2Д модель, характеризующая типовые параметры пласта. Проведены расчеты по оценке вовлечения в разработку недренируемых интервалов баженовской свиты. Показана необходимость использовать модификации технологии с закачкой воздушной смеси для реализации влажного внутрипластового горения.

Ключевые слова: Термогазовое воздействие, МУН, баженовская свита, закачка кислородсодержащей смеси (воздуха, водовоздушной смеси), кероген - органическое вещество, дренируемые и недренируемые породы, влажное внутрипластовое горение.

Для эффективной разработки запасов и ресурсов, находящихся в нефтеносных отложениях баженовской свиты (БС) Западно-Сибирского региона, необходимо применение методов повышения нефтеотдачи и технологий воздействия, направленных на создание теплового воздействия на керогенсодержащую породу БС. Оценка планируемых и применяемых сегодня технологий воздействия на данные пласты приведены в работах [1,2]. Опытные работы по применению термогазового метода увеличения нефтеотдачи пластов (ТГВ) показали свою эффективность и сегодня метод уже применяется при проведении работ по воздействию на отложения БС [3].

Говоря о повышении эффективности воздействия на пласты БС надо отметить, что для искусственного ускорения процесса катагенеза требуется постоянный и значительный подвод тепла для большего охвата пласта. Поэтому подключение к разработке недренируемых толщ БС является одной из важных задач и может реализовываться с использованием технологических решений, например, при реализации закачки термогазовых смесей с определенным водовоздушным отношением и также с различными комбинациями термо-газохимического воздействия.

Прежде чем рассмотреть результаты проведенных расчетов на линейной гидродинамической модели, необходимо выделить ряд особенностей пород БС и основные свойства данных отложений.

Для оценки возможностей фильтрации в породах баженовской свиты условно можно выделить два комплекса пород. Первый комплекс представлен дренируемыми интервалами, сложенными из преимущественно кремнистых и карбонатно-кремнистых породам, имеющих биогенное происхождение, с высоким содержанием органического вещества, расположенными в аргилито-алевритовой толще, чередующимися с дренируемыми карбонатизированными прослоями радиолярий и вторичных доломитов [4]. Второй комплекс пород, характерный для недренируемых частей разреза, представлен керогенсодержащей аргилито-глинистой толщей с закрытыми порами (общая пустотность по ряду исследований достигать 8–11%), в которых сформировались прочные битумо-подобные пленки на границах зерен, что позволяет значительно увеличить прочность образованного межзернового каркаса породы, но качество содержащихся углеводородов невысокое.

В процессе катагенеза при повышенных температурах и давлении происходило постепенное преобразование твердого органического вещества (керогена) в жидкую нефть и газ. В условиях быстрого накопления осадочного материала при наличии тонкого смешения различных литотипов пород и большого количества

глинистых минералов, в ряде случаев сформировались практически непроницаемые структуры в кровле БС и условия, препятствующие дальнейшей фильтрации нефти и газа.

Недренируемая часть пласта – матрица или «баженит» (согласно А.Э. Конторовича) представляет собой упруго-пластичную гетерогеннную среду, в которой при термодинамических условиях зоны катагенеза в результате крекинга керогена происходит химическое превращение его значительной части в жидкие и газообразные флюиды. Каркас преобразованного баженита хрупкий и сохраняет вторично образованную пористость только если поровое пространство заполнено флюидами [5].

Результаты многочисленных исследований кернового материала показал, что в замкнутых порах породы БС, составляющих в исследованных образцах до 20% от общего объема пор, находится значительное количество углеводородов и углеводородных гетероатомных соединений [6]. Эти ресурсы (кроме твердого ОВ) можно условно отнести к остаточным (трудноизвлекаемым) запасам и разделить на нефти средней и высокой плотности [7].

При формировании отложений БС одновременно по мере роста горного давления продолжался естественный пиролиз кероген-содержащих пород и формирование новых зон пустотности в местах преобразования керогеновых зерен породы. Выделяющиеся в процессе катагенеза углеводороды и газовые агенты резко увеличивали давление, что приводило к дальнейшему растрескиванию породы и формированию новых зон трещиноватости в направлениях, имеющих минимальные эффективные напряжения в породе, которое определяется и контролируется как структурным положением, так и литологическими особенностями слагающих данные зоны пород.

Часть уже сформированного углеводородного сырья, содержащего преимущественно газовые и легкие компоненты генерировалось и фильтровалось в зоны с развитой микротрещиноватостью и с повышенными ФЕС, где и сформировались зоны максимальной плотности запасов. Оставшаяся часть частично дегазированных более тяжелых углеводородных компонентов с высоким содержанием смол и асфальтенов оставалась запечатанными в закрытых порах.

Многочисленные исследования по скважинам на месторождениях баженовской свиты проводились в ОАО «Сургутнефтегаз» [8] и показали значительную зависимость фильтрационно-емкостных свойств (ФЕС) от литотипов породы. Согласно обобщенным результатам исследований литолого-физических характеристик пород в БС получены зависимости величины пустотности для различных литотипов породы от температуры. Согласно обобщенным результатам исследований (рисунок 1) средние значения пустотности дренируемых пород БС в пластовых условиях находятся в пределах от 4% до 8%. По результатам исследований также отмечено, что общая пустотность пород увеличивается с ростом пластовой температуры.

Проведенные исследования показали необходимость искусственного повышения температуры пласта для возможности эффективной разработки данных отложений. При этом можно будет происходить не только рост общей пустотности, но и увеличение охвата недренируемых интервалов пласта.

Данная особенность влияния температуры на изменение свойств породы баженовской свиты является основной предпосылкой использования термогазового воздействия как перспективного метода разработки залежей нефтематеринских пород. Детальному рассмотрению этого вопроса посвящены работы Боксермана А.А. с соавторами [9,10,11].

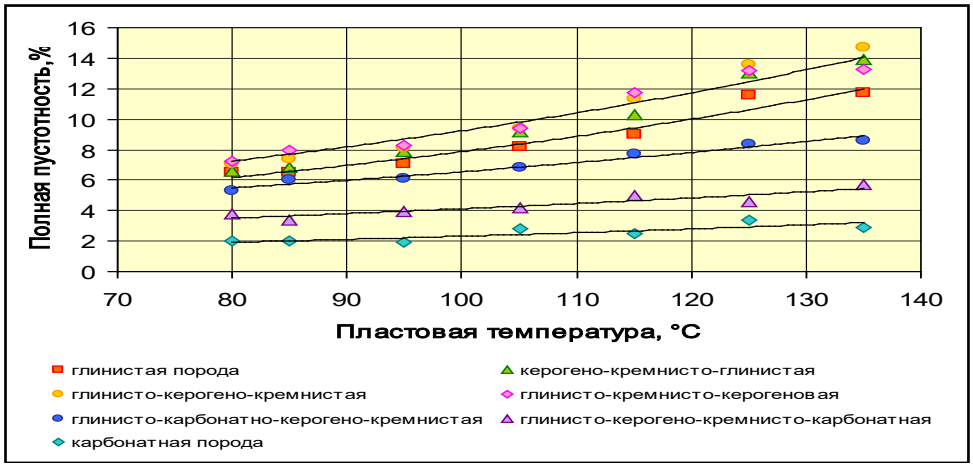


Рис.1. Зависимость средних значений полной пустотности литотипов пород БС от температуры пласта

Сложный характер строения и вещественного состава пород Баженовской свиты, высокое содержание керогена осложняют изучение их фильтрационно-емкостных характеристик. Тем не менее, определение фильтрационных свойств пород Баженовской свиты проводятся в значительном количестве [8,12]. Это позволяет определить основные подходы по характеру динамического изменения фильтрационно-емкостных свойств пород, необходимые для оценки рассматриваемого способа воздействия на пласты Баженовской свиты.

Надо отметить, что результаты исследований фильтрационных свойств пород образцов керна Баженовской свиты Салымского месторождения изучались в основном на плотных кернах [8]. В результате фильтрационных исследований на кернах значения проницаемости изменялись в пределах от 10-9 мкм2 до 10-3 мкм2. При этом для всех литотипов пород наблюдалось увеличение проницаемости с увеличением температуры проведения исследований. Повышение температуры опыта с 60 0С до 120 0С приводило к увеличению проницаемости в 2-6 раз. Изменение проницаемости от температуры достаточно явно представлено для коллекторов дренируемых зон и практически не определяется для глинистых пород, что связано прежде всего с разрушением породы в условиях эксперимента. К сожалению, диапазон исследования температур соответствует только начальным пластовым условиям, и не описывает характер изменения проницаемости для условий проведения термогазового воздействия при изменениях температуры вплоть до 400-600оС.

Для построения обоснованной температурной зависимости ФЕС керна баженовской свиты от температуры экспериментальных данных пока не достаточно, поэтому при моделировании была использована предварительная оценка, отражающая связь изменения (увеличения) пустотности, соответственно приводящее к росту проницаемости, при росте температуры в пласте.

Основные этапы создания 2Д модели

Для численной реализации процесса ТГВ, включая вовлечение изначально недренируемых зон (матрицы) баженовской свиты в процессе термогазового воздействия был использован гидродинамический симулятор STARS (CMG Канада) в неизотермической постановке.

Моделирование эффекта появления проницаемости возможно проводить используя различные подходы. В работе [13] рассмотрены ряд подходов, включая модель Бартена-Бэндиса, которая позволяет моделировать динамику изменения напряженного состояния пласта с сформированием соответствующей проницаемостью трещины, и метод, с использование модуля PermShale, описывающий появление проницаемости в результате воздействия температур в недренируемой зоне. При использовании данного модуля, в модели непосредственно задаётся зависимость проницаемости как функции от температуры.

Данный подход, с использование модуля PermShale. реализованный в симуляторе CMG и позволяющий, с учетом сегодняшнего уровня изученности процесса, получить корректные результаты, и был

использован при построении рассматриваемой модели.

Процессы окисления и пиролиза нефти и керогена в пласте в процессе закачки различных окислителей (в частности воздуха) подробно изложены в работах [14,15]. В данных исследованиях была выполнена серия экспериментов, которая позволила определить кинетические параметры реакций, и по исследованиям получены экспериментальные оценки выхода УВ и получения синтетической нефти из керогена БС баженовских отложений.

В процессе работы была создана линейная модель по схеме «нагнетательная – добывающая скважины» с недренируемыми слоями (интервалами матрицы), с учетом теплопотерь для численного воспроизведения основных процессов, происходящих в зонах при термогазовом воздействии на пласт. В модель закладывались средние фильтрационно-емкостные и геохимические параметры, характерные для данных отложений и была принята усредненная схема строения залежи, характерная для структуры пласта баженовской свиты (рисунок 2).

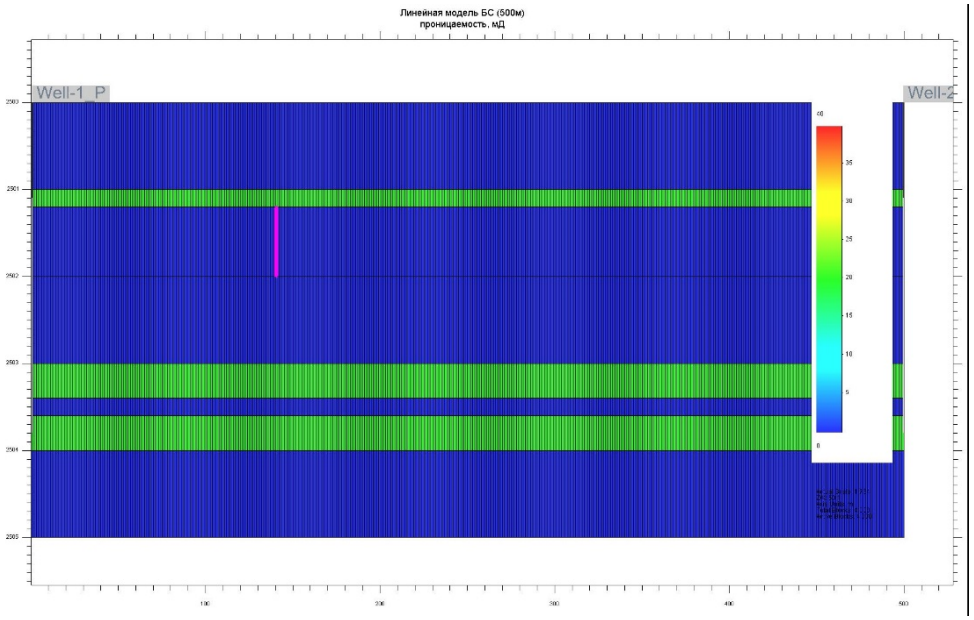


Рис.2. Изменение проницаемости по разрезу линейной модели БС (до начала закачки)

Пласт условно разделен на восемь интервалов (пропластков), включая пять недренируемых глинистых и керогенсодержащих непроницаемых пропластков, разделенных тремя маломощными (толщиной до 0,4м) дренируемыми слоями, имеющими средние для БС фильтрационно-емкостные свойства (ФЕС), с высоким содержанием карбонатного и кремнистого материала. Глинистые пропластки содержат основную часть органического вещества нефтематеринской породы - керогена. Основной приток флюидов в скважину и их закачка в пласт возможен из дренируемых пропластков.

Оценка эффективности воздействия на пласт и моделирование процессов проведена с применением различных технологических решений при закачке как сухого воздуха, так и водовоздушной смеси (различная циклическая закачка).

Для моделирования ТГВ использована композиционная модель пластовой нефти, которая содержит шесть углеводородных компонентов, включая фракции от метановой до фракции С18+, два не углеводородных компонента - азот и углекислый газ, также учтена вода и кислород, содержащийся в закачиваемом воздухе. Кроме этого, добавлены еще три компонента - кероген (Kerogen), его твердый остаток (Coke) и синтетическую нефть (Synt_Oil), возникающие при термодеструкции (пиролизе) керогена. Таким образом, всего в модели принято 16 компонентов.

Для описания окислительных и пиролитических процессов, протекающих при термогазовом воздействии, подготовлены химические реакции, которые учитывают окисление нефти и керогена, содержащихся в породе, пиролиз керогена с образованием из него синтетической нефти и

твердого остатка (кокса), а также горение указанного твердого остатка (кокса).

При процессе ТГВ возникает неизотермическая фильтрация ряда компонентов совместно с экзотермическими окислительными реакциями и связанными с ними термодинамическими изменениями компонентного состава, насыщающих пласт флюидов.

Кроме этого, модель позволяет моделировать механизм вытеснения нефти термогазовым методом с учетом таких процессов, как:

– экзотермическая химическая реакция окисления органических веществ кислородом воздуха с образованием в результате химической реакции продуктов окисления (CO, CO_2, кокса);

– испарение и конденсация легких углеводородных фракций;

– растворение легких углеводородных фракций, CO и CO_2 в нефти и формирование частичной смесимости в пласте;

– перенос тепла за счет теплопроводности и конвекции;

– теплопотери в кровлю и подошву пласта за счет теплопроводности;

– изменение коэффициента проницаемости глинистых пропластков, содержащих основную часть керогена за счет образования системы микротрещин, в ходе химических превращений.

Описанная выше композиционная модель учитывает компонентный состав жидкой и газовой фаз углеводородов и воды, фильтрующихся в пласте, что позволяет более точно рассчитывать фазовые переходы и изменение температуры в пласте.

Описание и кинетические параметры реакций

В модели применена следующая схема химических реакций, представленная ниже.

Первые две реакции описывают низкотемпературное окисление нефти, третья – пиролиз керогена, четвертая и пятая - сгорание керогена и кокса, соответственно.

1. C5toC7 + O2 --> WATER + CO2 Низко-температурное окисление
2. C18toC30 + O2 --> WATER + CO2 Низко-температурное окисление
3. Kerogen --> Sint oil + Sint gas +Coke Термолиз керогена
4. Coke + O2 --> WATER + CO2 Высокотемпературное окисление
5. Kerogen + O2 --> WATER + CO2 Высокотемпературное окисление

Для описания процесса, описывающего зависимость скорости реакции от температуры (модель Арениуса) необходима информация по параметрам формулы – энергия активации (Ea) и предэкспоненциальный коэффициент (A). Энергия активации характеризует необходимое количество энергии на моль вещества для начала реакции, предэкспоненциальный коэффициент характеризует частоту соударений молекул реагирующих веществ в единичном объёме. Скорость реакции прямо пропорционально значению предэкспоненциального коэффициента.

Для расчёта стехиометрических коэффициентов необходимо знать формулу компонента, т.е. количество атомов элементов, входящих в молекулу вещества. Поскольку в принятой схеме реакций (за исключением реакций термолиза) в реагентах и продуктах содержаться только углерод, водород и кислород, то для написания упрощённой формулы вещества достаточно задаться углеродно/водородным соотношением (УВО) и использовать молекулярную массу компонентов

Параметры, использованные при построении неизотермической линейной модели отложений БС

Исходя из рассмотренных выше исследований, в модели было принято, что происходит рост проницаемости до мах 5 мД в недренируемой зоне, в случае нагрева элемента пласта выше 300-350 °C.

К недостаткам данного подхода можно отнести тот факт, что при этом предполагается непосредственная связь между температурой и проницаемостью, тогда как в реальности, температура является лишь одной из причин для начала процесса, вызывающего рост давления внутри замкнутого порового пространства. Как пример, можно отметить, что если в зоне прогрева содержание керогена будет

недостаточно для того чтобы продукты пиролиза вызвали необходимое для разрыва породы давление, то трещиноватость формироваться не будет.

Для моделирования процесса подготовлена линейная модель размерностью 8м*1м*500м. Размер ячейки по осям X,Y составил 1 м и по вертикали - от 0,2 до 1 м. Расстояние между скважинами соответствует средним величинам, применяемым при разработке данных отложений. Темп закачки воздуха для линейной модели составляет от 5-10м3/сут, начальная нефтенасыщенность составила – 0,85, доля (от общей толщины) дренируемой зоны в общем разрезе пласта составила 20%, проницаемость дренируемых зон – 10 мД, проницаемость недренируемых зон после температурного воздействия – 5 мД.

Для моделирования процесса ТГВ проводится закачка сухого воздуха и анализ подключения недренируемых зон пласта. На рисунках 3-6 показано изменение температуры, эффективной пористости и проницаемости в процессе закачки и после 18 лет проведения закачки воздуха.

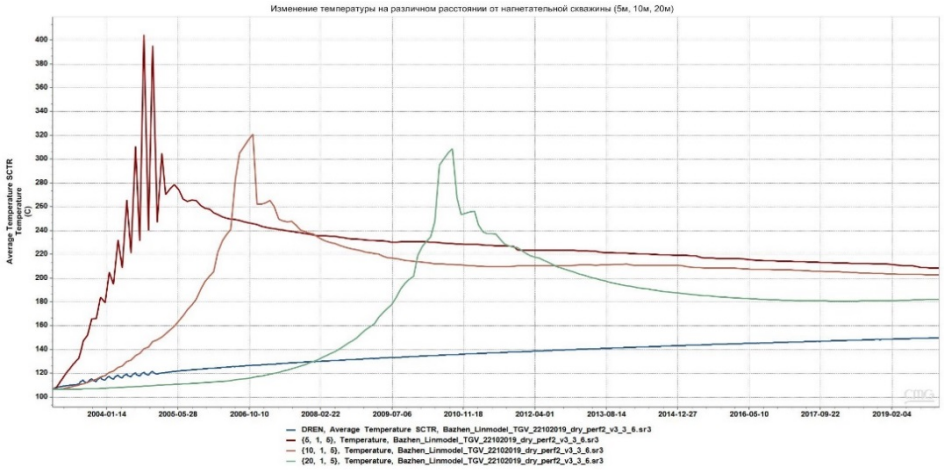


Рис.3. Изменение температуры во времени на различном расстоянии от нагнетательной скважины

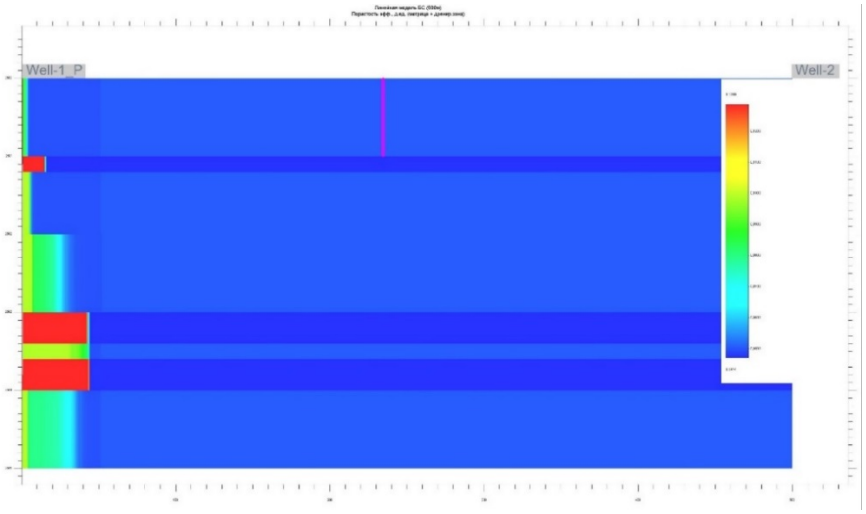


Рис.4. Изменение эффективной пористости по разрезу линейной модели БС (через 18 лет с начала закачки)

Как видно из рисунков 4 и 5 в модели реализован процесс динамического вовлечения матрицы в процесс фильтрации за счет возникновения и дальнейшего изменения эффективной проницаемости и пористости. На рисунке 3 показано некоторое снижение температуры со временем, что связано в том числе и с тем, что закачиваемый в пласт одинаковый объем воздуха поступает и уже расходуется не только в дренируемой, но и в изначально недренируемой зоне. Уменьшение расхода воздуха для дренируемой зоны и приводит к снижению средней температуры.

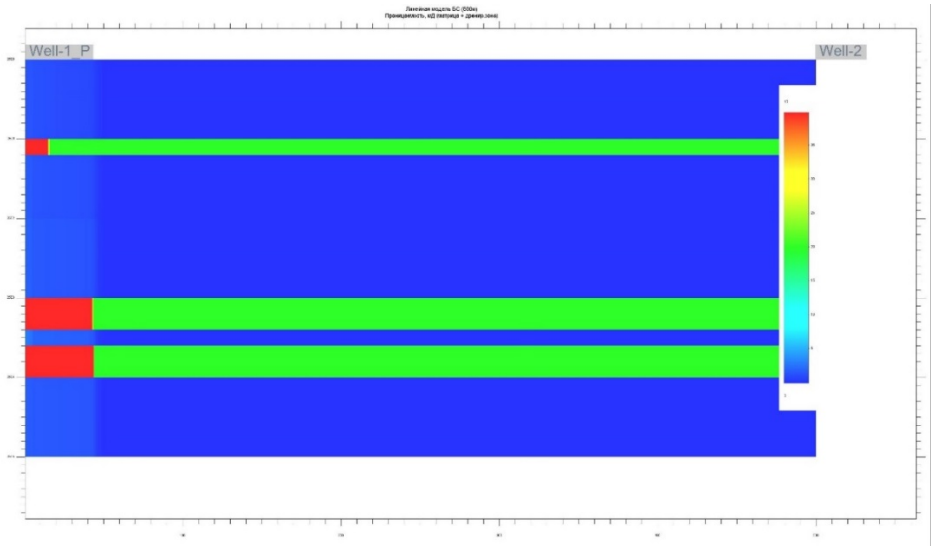


Рис.5. Изменение проницаемости по разрезу линейной модели БС (через 18 лет с начала закачки)

На рисунке 6 показан рост средней температуры в разрезе пласта по дренируемым и недренируемым интервалам в разные периоды времени. Видно, что поступление кислорода в матрицу активизировало процесс горения, а теплообмен между выше- и нижележащими слоями модели привел к выравниванию средних температур в дренируемых и недренируемых интервалах. На рисунке 7 приведен разрез пласта по температуре на момент окончания закачки.

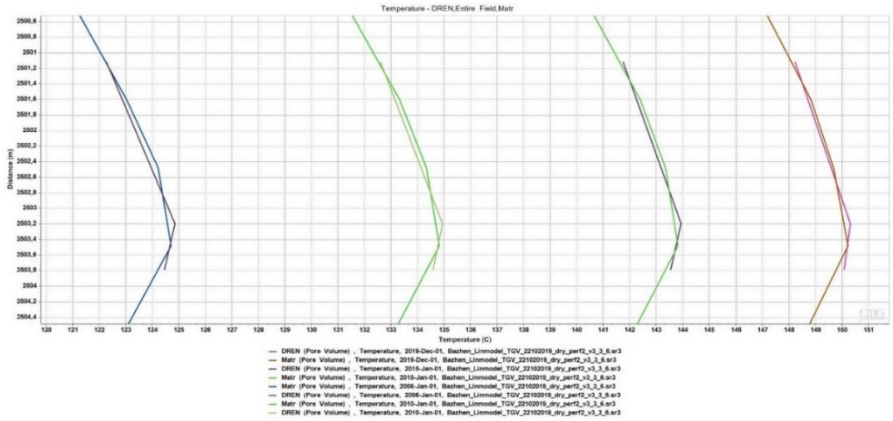


Рис.6. Изменение температуры по разрезу пласта модели БС (через 5, 10, 15и 18 лет с начала закачки)

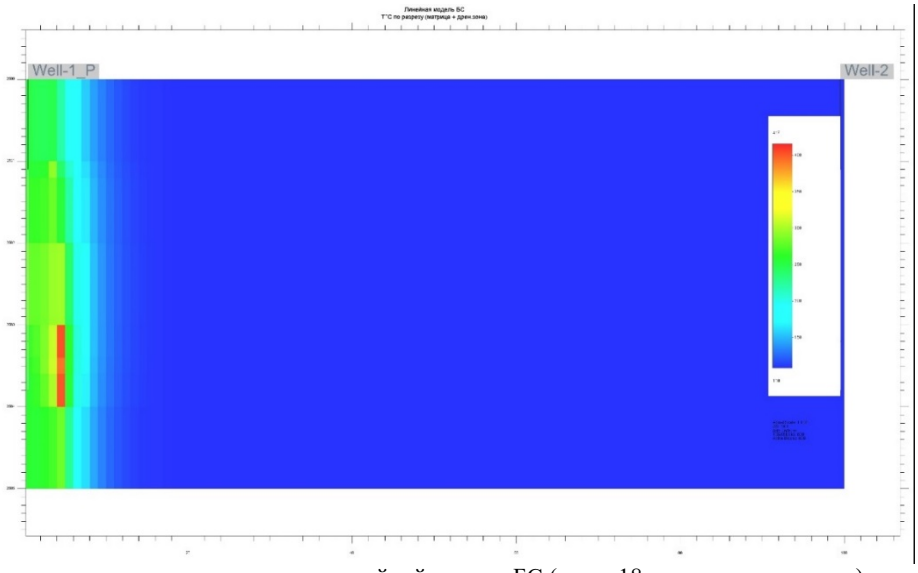


Рис.7. Изменение температуры по разрезу линейной модели БС (через 18 лет с начала закачки)

Как видно из рисунка 7 реализация сухого внутрипластового горения позволяет подключить матрицу, но объем охвата матрицы, дополнительно подвергнутый тепловому воздействию, за рассматриваемый период незначительный. Это обуславливается прежде всего особенностью сухого горения, связанной с незначительным продвижением тепла перед фронтом горения. Для увеличения охвата пласта тепловым воздействием возможно проведение закачки термогазовых смесей с определенным водовоздушным отношением (ВВО). При закачке водовоздушных смесей к эффекту вовлечения матрицы по всей толще, добавляется и перенос тепла водяным паром, что позволяет создать увеличенную зону прогрева всего пласта. Результаты таких расчетов будут рассмотрены в следующих статьях.

Таким образом результаты расчетов на линейной модели показали, что одна из важных задач при разработке керогенсодержащих отложений БС заключается в максимально эффективном использовании сгенерированного на фронте горения и при пиролизе керогена тепла и перенос его впереди фронта горения. Это приведет к расширению зон протекание процесса по площади и увеличению охвата воздействием. Данное увеличение зон прогрева пласта можно реализовать при применении влажного внутрипластового горения (ВВГ), когда реализуется закачка термогазовых смесей с определенным водовоздушным отношением.

Выводы и заключение

Результаты расчетов, полученные на математической модели позволяют сделать следующие выводы и заключения:

- Показана возможность реализации на гидродинамических моделях процессов при термогазовом воздействии с учетом подключения недренируемых зон в керогенсодержащих пластах;
- При реализации процесса динамического вовлечения матрицы в процесс фильтрации за счет возникновения и дальнейшего изменения эффективной проницаемости значительно растет эффективность процесса;
- Дальнейшее повышение эффективности воздействия на отложения БС можно реализовать при применении влажного внутрипластового горения. В случае закачки в пласт водовоздушной смеси за счет конвективного переноса тепла впереди фронта горения может происходить увеличение нагрева недренируемых интервалов пласта баженовской свиты с керогенсодержащими породами.
- Использование керогена в качестве топлива для самопроизвольных окислительных процессов позволяет поддерживать высокую температуру на фронте горения и задачи моделирования заключаются в том числе и в определении режимов, направленных на максимальный и преимущественно одновременный охват как дренируемых, так и

подключаемых к разработке непроницаемых зон баженовской свиты.

Работа выполнена при поддержке гранта РФФИ № 18–07-00504_А.

Estimation of matrix zone drainage process involvement in 2D model simulation of thermal-gas method for kerogen-content bazhenov suite

Mironov D.T., Yalov P.V. (NIISI RAS)
Scientific Research Institute of System Development of the Russian Academy of Sciences, mdt1958@yandex.ru

Abstract: Problems of mathematical modeling of three-phase flow while using the thermo-gas method for enhanced oil recovery - are considered. Brief characteristic of bazhenov kerogen content rock types and some parameters of termo-gas method for enhanced oil recovery are applied. The main features of the applied technology discussed. 2D simulation model with typical reservoir features are prepared. Simulation of matrix zone involvement in drainage process are discussed. Necessity of usage the modification with injection of water-air mixture in the reservoir in wet in-situ combustion process are proposed.

Key words: thermal-gas treatment, EOR methods, bazhenov formation, oxygen-containing mixture injection, kerogen, miscible drive, wet in-situ combustion, enhanced oil recovery

Литература

1. О.В. Коломийченко, А.А. Чернов, Технология для освоения баженовской свиты. Концепция внутрипластового каталитического ретортинга. «Oil & Gas Journal Russia», Октябрь 2015. 58–66.
2. Д.Т. Миронов, С.Г. Вольпин, В.А. Юдин. Технологические подходы при эксплуатации скважин баженовской свиты и оценка возможности подключения в разработку ресурсов недренируемых зон. «Вестник Кибернетики». 2018. № 3, 233–246.
3. В.Ю. Алекперов, В.И. Грайфер, Н.М. Николаев, В.Б. Карпов, В.И. Кокорев, Р.Г. Нургалиев, А.П. Палий, А.А. Боксерман, В.А. Клинчев, А.В. Фомкин, Новый отечественный способ разработки месторождений баженовской свиты (часть 1). «Нефтяное хозяйство» № 12. 2013, 100–105.
4. В.С. Славкин, А.Д. Алексеев, В.Н. Колосков. Некоторые аспекты геологического строения и перспектив нефтеносности баженовской свиты на западе Широтного Приобья, «Нефтяное хозяйство» 2007, № 8, 100-104.
5. А.Э. Конторович. Феномен Баженовской свиты: литология, органическая геохимия, палеогеография, пост-седиментационная эволюция, потенциал генерации аккумуляции нефти и газа. Доклад на VIII Всероссийском литологическом совещании. ИНГГ, Новосибирск. URL: https://www.youtube.com/watch?v=nPoVWKZmeVw от 30 окт. 2015 (дата обращения: 06.07.2018).
6. Г.А. Калмыков, Н.С. Балушкина, В.С. Белонин, С.И. Билибин, Т.Ф. Дьяконова, Т.Г. Исакова. Пустотное пространство пород баженовской свиты и насыщающие его флюиды. «Недропользование XXI век», 2015. № 1, 34–46
7. Н.С. Балушкина, Г. А. Калмыков, В.С Белохин, Р.А. Хамидуллин, Д.В. Корост. Кремнистые коллекторы баженовского горизонта Средне-Назымского месторождения и структура их пустотного пространства. «Вестник Московского университета», Серия 4. Геология. 2014. Том 4. № 2, 35–43.
8. Ю.Е. Батурин, В.П. Сонич, А.Г. Малышев, О.Г. Зарипов, В.Г. Шеметилло. Оценка перспектив применения гидротеровоздействия в пласте Ю0 месторождений ОАО «Сургутнефтегаз». «Интервал» № 1 (36). 2002, 17–36.

9. А.А. Боксерман. Результаты и перспективы применения тепловых методов воздействия на пласт. «Тепловые методы воздействия на пласт» (Материалы межотраслевого семинара, г. Ухта, 5-8 октября 1971 г.), ВНИИОЭНГ, Москва, 10-16.
10. С.В.Антонов, А.М. Полищук, А.А. Боксерман, Развитие термогазового метода повышения нефтеотдачи. Кинетические закономерности автоокисления керогена и нефти. В сб. «Теория и практика применения методов повышения нефтеотдачи пластов» Материалы 2-го Международного симпозиума 15 сентября 2009 г. - Т. 1, 183.
11. А.А.Боксерман, С.Г. Вольпин, Д.Т. Миронов. «Особенности моделирования МУН для условий применения термогазового метода увеличения нефтеотдачи в различных геолого–физических условиях». Материалы II Всероссийской науч.-практической конференции в г.Сургут, 27 мая 2016, «Север России: стратегии и перспективы развития», 8-19.
12. М.Ю. Зубков. Литолого-петрографическая характеристика отложений баженовской и абалакской свит центральной части Красноленинского свода. «Геология и геофизика». 1999. т.40, №12. 821–1836.
13. А.М. Шахмаев, Я.О. Симаков, П.В. Пятибратов, А.А. Мосесян. Численная реализация механизма термогазового воздействия на двумерной модели. «Экспозиция нефть газ», Февраль 1(61) 2018.
14. В.И. Кокорев, Н.Г. Судобин, А.М. Полищук, Е.Г. Горлов. Термодеструкция керогена битуминозных пород тутлейской (баженовской) свиты месторождений Красноленинского района. В сб. «Теория и практика применения методов повышения нефтеотдачи пластов». Материалы 2-го Международного симпозиума 15 сентября 2009 г. - Т. 1, 45.-50
15. С.А. Власов, Н.Г. Судобин, А.М. Полищук. Исследование процесса термического воздействия на образцы пород баженовской свиты. «Нефтепромысловое дело», 2010.- №3, 45-54.

Влияние закачки водовоздушной смеси на эффективность термогазового воздействия для баженовской свиты при 2D моделировании

Д.Т. Миронов[1], А.А. Глушаков[2], П.В. Ялов[3]

[1]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, mdt1958@yandex.ru;

[2]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, stormwww@yandex.ru;

[3]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, petryalov@gmail.com;

Аннотация. Рассматриваются вопросы математического моделирования трехфазной фильтрации в условиях применения термогазового метода увеличения нефтеотдачи (ТГ МУН). Описаны процессы при закачке водо-воздушной смеси для реализации влажного внутрипластового горения. Приведены описания основных механизмов воздействия на пласт при реализации ТГ МУН. Показаны характерные особенности данной технологии при закачке водо-воздушной смеси.

Ключевые слова: Термогазовое воздействие, МУН, баженовская свита, закачка кислородсодержащей смеси, водовоздушное отношение, кероген - органическое вещество, дренируемые и недренируемые породы, влажное внутрипластовое горение, смешивающееся вытеснение.

1. Введение

С целью поиска путей и способов эффективной разработки запасов и ресурсов нефтеносных отложений баженовской свиты (БС) Западно-Сибирского региона в настоящее время проводятся опытные работы по разработке и испытанию различных методов воздействия, прежде всего направленных на создание теплового воздействия на керогенсодержащую породу БС [1,2].

Работы по применению отечественного метода увеличения нефтеотдачи пластов - термогазового воздействия (ТГВ) показали свою эффективность и сегодня уже получены первые результаты применения технологии на Среде-Назымском месторождении для отложений БС [1].

В настоящей статье приводятся результаты моделирования термогазового воздействия на пласт при закачке водовоздушных смесей с различными соотношениями вытесняющих агентов водовоздушной смеси.

Описание основных параметров разработанной 2Д линейной модели с учетом подключения недренируемых интервалов (матрицы) в керогенсодержащих отложениях баженовской свиты и характеристики основных физических процессов при ТГВ приведено в предыдущих статьях [3,4] и настоящая статья является продолжением данной тематики по повышению эффективности разработки трудноизвлекаемых запасов из керогенсодержащих отложениях баженовской свиты (БС).

В статье [4] показано, что применение технологии ТГВ, когда проводится только закачка воздуха, позволяет задействовать и подключить к выработке недренируемые зоны керогенсодержащих интервалов матрицы породы, но объем охвата данных зон незначительный. Это объясняется медленным продвижением тепла перед фронтом горения и тем, что значительное содержание керогена в породе приводит к торможению теплового фронта (очага горения) от источника (воздухонагнетательной скважины). С целью увеличения охвата пласта тепловым воздействием предложено проведение закачки термогазовых смесей с определенным водовоздушным отношением (ВВО) [5]. При этом к эффекту подключения недренируемых зон (матрицы породы) к процессу фильтрации, добавляется и перенос тепла водяным паром и, соответственно, создание и расширение зоны прогрева по всей толщине пласта.

2. Математическая модель

Прежде чем рассмотреть результаты расчетов, полученных на 2D термогидродинамической модели и анализ технологических решений при применении данной модификации ТГВ с закачкой водовоздушной смеси, необходимо отметить некоторые особенности, характерные для процесса влажного внутрипластового горения (ВВГ).

Влажное горение применяется прежде всего для использования высокой теплоемкости воды и ее способности перераспределять тепло, сгенерированное за фронтом горения вперед, повышая эффективность процесса. При этом формируются протяженные зоны «парового плато» с повышенными температурами и увеличиваются зоны пара и воды (рисунок 1) и снижается количество сгорающего топлива.

Рис.1. Характерный профиль изменения температуры пласта при закачке водовоздушной смеси

Как показано на рисунке 1 при перемещении фронта горения по пласту в нем выделяется ряд характерных температурных зон, что отражено в ряде работ по внутрипластовому горению, созданных А.А. Боксерманом, В.П. Степановым, Ю.П. Желтовым и другими исследователями [5].

При увеличении содержания воды в смеси температура на фронте горения падает и эффективность резко снижается. При высоких ВВО (более 0,005 м3/нм3(соответствует сверхвлажному горению), закачиваемая вода с определенного момента уже в жидком виде внедряется в зону горения, снижая температуру и фактически останавливая процесс горения. В условиях керогенсодержащего коллектора Баженовской свиты это приведет к снижению температуры ниже температуры пиролиза и полное прекращение термодеструкции керогена. Также из-за ухода тепла в окружающие недренируемые интервалы пласта происходит дополнительное снижение температуры.

Лабораторные исследования для проекта на месторождении Суплако-де-Барко [6] показали, что оптимальное водовоздушное отношение составляет около 0,0014 м3/нм3. Оптимальное ВВО для каждой залежи свое, и определяется прежде всего фактической неоднородностью пласта [7].

Для численной реализации процесса ТГВ на гидродинамическом симуляторе STARS (CMG Канада) была подготовлена 2D модель, включающая возможность вовлечение углеводородного сырья (УВС) из изначально недренируемых зон (матрицы) баженовской свиты в процессе термогазового воздействия. Подробно данная модель описана в работе [4].

В симуляторе CMGSTARS была создана линейная модель по схеме «нагнетательная – добывающая скважины» с недренируемыми слоями (интервалами матрицы), со средними фильтрационно-емкостными и геохимическими параметрами, характерными для данных отложений. Схема строения залежи, характерная структуре пласта баженовской свиты приведена на рисунке 2.

Пласт был разделен на восемь интервалов (пропластков), включая пять недренируемых кероген-содержащих непроницаемых пропластков, и три проницаемых интервала (толщиной до 0,4м). Недренируемые пропластки содержат основную часть органического вещества нефтематеринской породы – керогена, а приток флюидов из скважины и закачка агентов в пласт проводится из дренируемых интервалов.

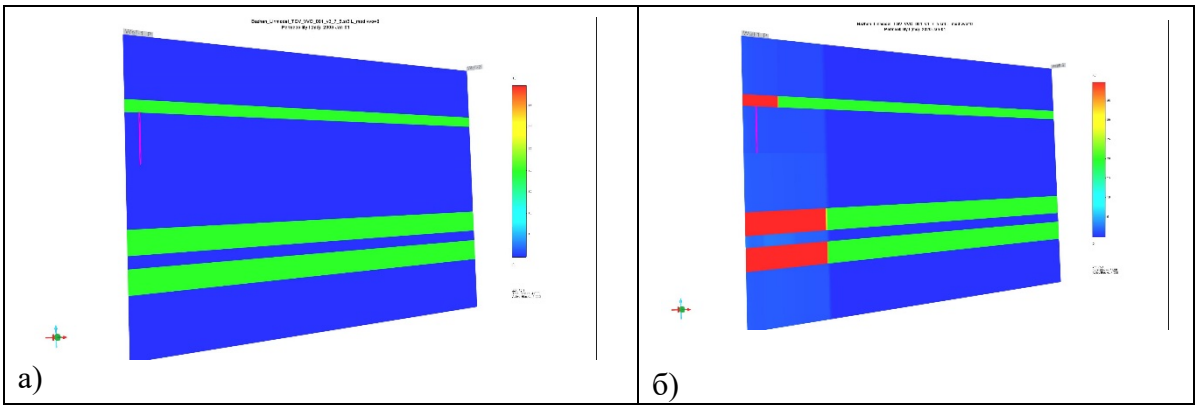


Рис.2. Принципиальная схема строения линейной модели БС и изменение проницаемости по разрезу – а) до начала закачки, б) по окончании закачки воздуха

Для моделирования ТГВ использована композиционная модель пластовой нефти, которая содержит шесть углеводородных компонентов, от метановой до фракции С18+, азот, углекислый газ, вода и кислород, содержащийся в закачиваемом воздухе. Также добавлены три компонента - кероген, его твердый остаток (кокс) и синтетическая нефть, возникающие при термодеструкции (пиролизе) керогена.

В модели описаны химические реакции, которые учитывают окисление нефти и керогена, пиролиз керогена с образованием из него синтетической нефти и твердого остатка (кокса) и также горение указанного твердого остатка.

При моделировании термогазового воздействия на пласты БС учтены следующие особенностей процесса:

- низко и высокотемпературное окисление (горение) УВ компонентов;
- термолиз (термодеструкции) керогена при температурах свыше 350ºС, приводящий к выходу жидкой фазы («синтетическая» нефти), попутного газа и тяжелого остатка (кокса). Реакция термолиза происходит впереди фронта горения, без доступа кислорода (весь кислород поглощается в зоне горения) с разложением керогена на углеводородные фракции и кокс, который составляет примерно 40% (по массе) продуктов реакции термолиза керогена;
- процесс горения кокса, как основного источника энергии, генерирующейся в пласте;
- изменения пористости и проницаемости в недренируемой части породы при изменении термобарических условий с использованием модуля PermShale, реализованного в симуляторе CMG.

3. Результаты расчетов

Как отмечено выше, одной из важных задач при разработке кероген-содержащих отложений БС является расширение зон протекание процесса по площади и увеличение охвата воздействием, которое возможно при закачке термогазовых смесей с определенным водовоздушным отношением. Данную технологию можно отнести к тепловым процессам характерным при применении влажного внутрипластового горения (ВВГ). Другая особенность ВВГ заключается в способности пара аккумулировать большое количество тепла и переносить его в область перед фронтом горения и далее.

В линейной модели осуществлялась закачка водовоздушной смеси с различной объемной долей воды в смеси (0.0015 м3/нм3, 0.001, 0.0005). Компонентный состав задавался в виде объемных долей каждого компонента смеси. При присутствии воды подвижность водовоздушной смеси ниже, чем воздуха, поэтому приемистость скважины может уменьшиться. Рассчитанные варианты сравнивались с вариантом с сухой закачкой воздуха в одинаковых условиях.

Изменение температуры по вариантам в процессе ТГВ при закачке сухого воздуха – сухое горение (СГ) и при закачке водовоздушной смеси с ВВО 0.001, полученные по результатам 2D моделирования на момент окончания расчета (через 18 лет с начала закачки) представлено на рисунке 3.

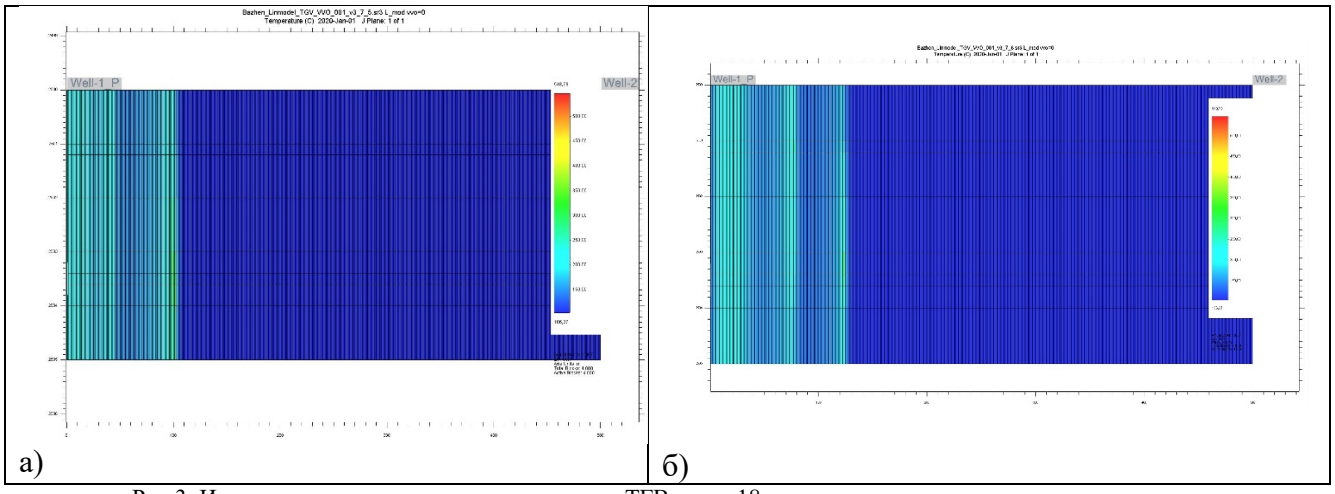


Рис.3. Изменение температуры при проведении ТГВ через 18 лет с начала закачки по вариантам
а) закачка СГ б) закачка водовоздушной смеси с ВВО 0.001

Сравнивая динамику продвижения температурного фронта по простиранию пласта видно, что переход на закачку водовоздушной смеси с ВВО 0.001 приводит к ускорению продвижения фронта горения и увеличению охвата пласта, включая недренируемые зоны изначально непроницаемой матрицы породы. В результате ускоренного продвижения фронта происходит и увеличение охвата по пласту, которое составляет до 20% по сравнению с сухим горением.

Также меняется и характер продвижения теплового потока по разрезу. На рисунке 4 показано изменение охвата тепловым воздействием по разрезу пласта, которое отражено в изменении параметра эффективной пористости. Данный параметр, наиболее наглядно характеризует рост эффективной пористости в недренируемых зонах пласта за счет пиролиза керогена и подключению к разработке ранее недренируемых толщин (интервалов).

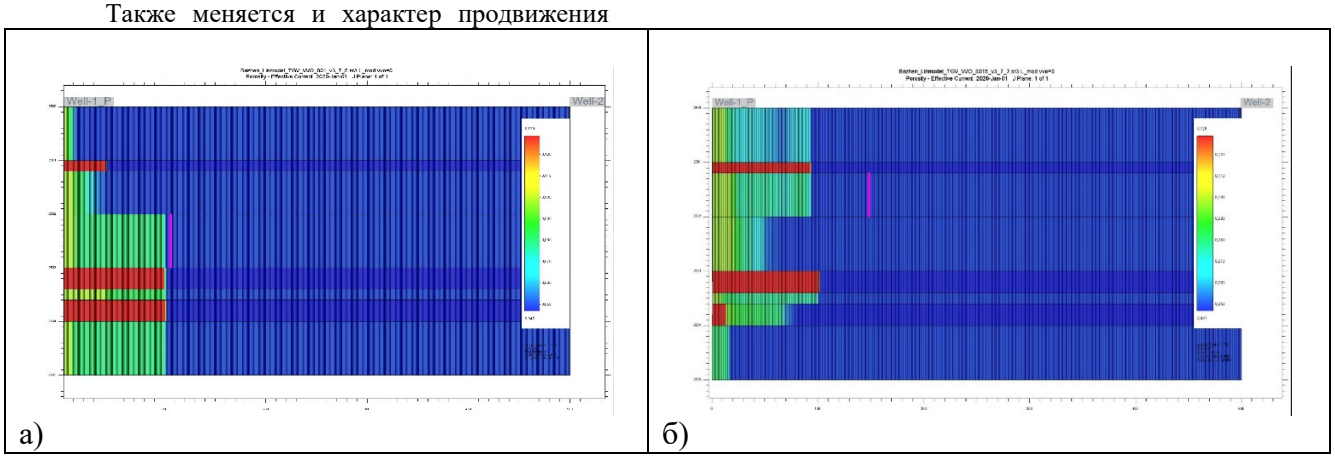


Рис.4. Изменение эффективной пористости при проведении ТГВ через 18 лет с начала закачки по вариантам
а) закачка СГ б) закачка водовоздушной смеси с ВВО 0.0015

Как видно на рисунке 4 при переходе на закачку водовоздушной смеси происходит значительное перераспределение поступающих агентов закачки – воздуха и воды по пласту и подключение преимущественно кровельных зон разреза.

Наличие воды в закачиваемой смеси, позволяет не только продвинуть фронт, но и изменить и при необходимости регулировать охват воздействием по разрезу.

На линейной модели, включающей матрицу и дренируемую зону были проведены расчеты по ВВГ с водовоздушными отношениями 0.0005, 0.001 и 0.0015. Расчеты показали, что с повышением водовоздушного отношения температура по направлению продвижения фронта горения несколько снижается. Это происходит за счет увеличения переносимого

водяным паром тепла из зоны за фронтом горения в зону перед фронтом горения и некоторым снижением общей температуры. В случае если температура в матрице падает ниже температуры начала пиролиза керогена (около 350°С) интенсивность данной реакции снижается, вплоть до ее прекращения. В данных условиях для выравнивания и поддержания фронта горения и равномерного охвата недренируемых зон необходимо предусматривать проведение мероприятий по регулированию процесса и изменению технологических режимов работы, в том числе изменяя соотношения и циклы закачиваемых агентов.

Сравнение температурных режимов во времени для различных зон пласта (находящихся на расстоянии 50 и 100м от нагнетательной скважины) при сухом горении и при закачке водовоздушной смеси с ВВО 0.001, полученные завесь период проведения ТГВ представлено на рисунке 5.

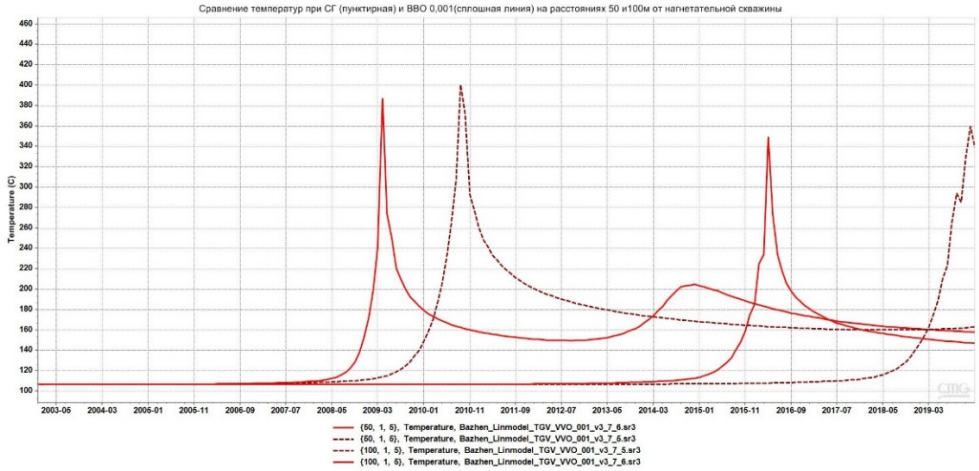


Рис.5. Изменение температуры во времени для различных зон пласта (на расстоянии 50 и 100м от нагнетательной скважины) при СГ (пунктирная линия коричневого цвета) и при ВВО 0.001 (красного цвета)

Анализ результатов расчета показывает, что время (для варианта с ВВО 0.001) прохождение теплового фронта через зоны пласта, находящиеся на расстоянии 50 м от нагнетательной скважины опережает на 1,5 года аналогичное время для варианта с СГ (коричневая линия). Соответствующее отставание теплового фронта по времени для СГ на расстоянии 100м от нагнетательной скважины, уже составляет более 4,5 лет.

На рисунке 5 также видно характерное для рассматриваемой технологии ТГВ с ВВО формирование второго температурного пика через 5 лет, которое характеризуется, в основном, притоком углеводородного сырья, сгенерированного и поступившего из изначально недренируемых зон матрицы. Данные УВ частично расходуются на низкотемпературное окисление, вызывают дальнейший рост температуры и подключаются к дальнейшей фильтрации к добывающей скважине.

Дальнейшее увеличение водовоздушного отношения до 0,0015 также приводит к замедлению перемещения температурного фронта и снижению температуры.

На рисунке 6 представлена динамика изменения нефтеотдачи для КИН по дренируемым интервалам (сплошная линия) и для залежи в целом (пунктирная линия) для вариантов с СГ, ВВО 0.001 и 0,0015 (через 18 лет с начала закачки).

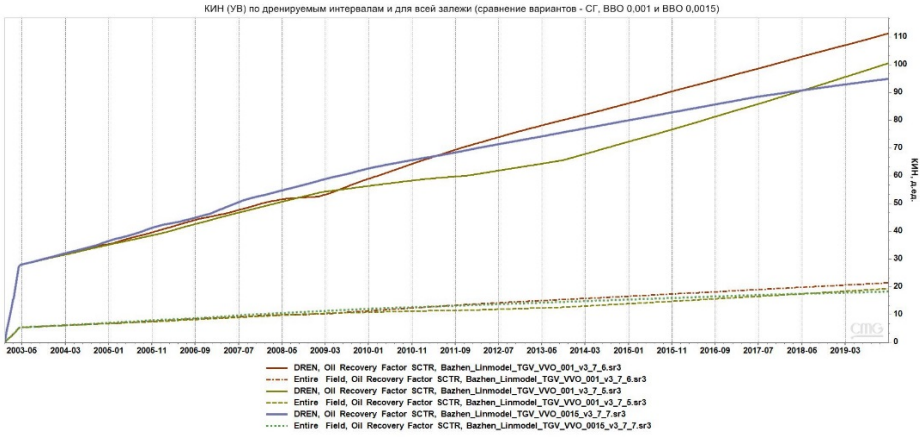


Рис. 6. Изменение КИН по дренируемым интервалам (сплошная линия) и для залежи в целом (пунктирная линия) для вариантов с СГ, ВВО 0.001 и 0,0015 (через 18 лет с начала закачки)

Согласно приведенному сопоставлению дополнительный эффект от реализации влажного горения составил около 10% в добыче жидких УВ. При этом варианты с влажным горением отличаются между собой незначительно. Наиболее предпочтительным выглядит вариант с ВВО 0.001, так как при его реализации более равномерно поддерживается процесс горения в матрице. Это позволяет наиболее полно реализовать потенциал термогазового метода и получить дополнительную добычу в виде синтетической нефти при пиролизе керогена.

4. Заключение

Таким образом расчеты показали, что при проведении ТГВ с закачкой водовоздушной смеси формируется различный характер теплопереноса в областях позади и впереди фронта горения. В области впереди фронта горения скорость конвективного теплопереноса будет выше скорости перемещения фронта горения. При осуществлении таких процессов при ВВГ в области впереди фронта горения может развиваться зона прогрева пласта, состоящая из зон насыщенного пара и горячего конденсата, имеющих высокие вытесняющие способности.

Проведенный выше анализ позволяет сделать следующие выводы и заключения:

- При закачке в пласт водовоздушной смеси впереди фронта горения развиваются зоны насыщенного пара, благодаря которым происходит значительный нагрев недренируемых интервалов пласта баженовской свиты с керогенсодержащими породами.
- Для определения необходимых технологических режимов работы нагнетательных скважин требуется предварительное проведение расчетов на гидродинамической модели с изменением состава вытесняющего агента и других параметров.
- В дальнейшем будет проведена оценка оптимальных режимов закачки агентов с ВВО для обеспечение эффективного извлечения нефти из дренируемых зон за счет формирования смешивающегося вытеснения.
- Расчеты показали, что использование керогена в качестве топлива для самопроизвольных окислительных процессов позволяет поддерживать высокую температуру на фронте горения и необходимо определить оптимальные режимы закачки вытесняющих агентов для увеличения охвата как дренируемых, так и подключаемых к разработке непроницаемых зон баженовской свиты.
- При реализации МУН необходимо проведение постоянного контроля за процессом закачки воздуха методами промысловых геофизических и гидродинамических исследований, одновременно с расширением экспериментальных и численных исследований при разработке отложений баженовской свиты, имеющих большой ресурсный потенциал.

Работа выполнена при поддержке гранта РФФИ № 18–07-00504_А.

Water-air mixture influence on thermal-gas method efficiency for bazhenov suite oil recovery in 2D modelling

D.T. Mironov, A.A. Glushakov, P.V. Yalov (NIISI RAS)
Scientific Research Institute of System Development of the Russian Academy of Sciences

Abstract. Problems of mathematical modeling of three-phase flow while using the thermo-gas method for enhanced oil recovery - are considered. Brief characteristic of bazhenov kerogen content rock types and some parameters of thermo-gas method for enhanced oil recovery are applied. The main features of the applied technology discussed when water-air mixture injected in the reservoir in wet in-situ combustion process.

Key words: thermal-gas treatment, EOR methods, bazhenov formation, oxygen-containing mixture injection, kerogen, miscible drive, wet in-situ combustion, enhanced oil recovery.

Литература

16. В.Ю. Алекперов, В.И. Грайфер, Н.М. Николаев, В.Б. Карпов, В.И. Кокорев, Р.Г. Нургалиев, А.П. Палий, А.А. Боксерман, В.А. Клинчев, А.В. Фомкин, Новый отечественный способ разработки месторождений баженовской свиты (часть 1). «Нефтяное хозяйство» № 12 (2013), 100–105.
17. Д.Т. Миронов, С.Г. Вольпин, В.А. Юдин. Технологические подходы при эксплуатации скважин баженовской свиты и оценка возможности подключения в разработку ресурсов недренируемых зон. «Вестник Кибернетики», № 3 (2018), 233–246.
18. Д.Т. Миронов, К.Д. Ашмян, А.В. Гореликов. Учет особенностей строения пластов баженовской свиты при построении гидродинамических моделей и моделировании термогазового воздействия. «Вестник Кибернетики», № 2 (2018), 109–119.
19. Д.Т. Миронов, П.В. Ялов. Оценка вовлечения в разработку недренируемых керогенсодержащих зон баженовской свиты при моделировании термогазового воздействии. «Труды НИИСИ РАН» том 9 (2019), № 6.
20. А.А. Боксерман, С.А. Жданов, А.А. Кочешков и др. Внутрипластовое горение с заводнением при разработке нефтяных месторождений. «Тр. ВНИИ» вып.LVIII, (1974), 168-182.
21. Joseph C. Pusch W.H. A Field comparison of wet and dry combustion, «JCPT» N32 1980, pp 1523-1528.
22. P. S. Sarathi. In-situ combustion handbook - principles and practices, 1999 BDM Petroleum Technologies.

Моделирование видимого движения Земли вдоль участков суточной трассы МКС в космических видеотренажерах

П.Ю. Тимохин[1]

[1]ФГУ ФНЦ НИИСИ РАН, Москва, Россия, webpismo@yahoo.de

Аннотация. В статье рассматривается задача реализации орбитальных наблюдений трехмерной виртуальной модели Земли в космических видеотренажерах. Предложена модифицированная технология моделирования видимого движения («бега местности») Земли вдоль участков суточной трассы Международной космической станции (МКС). В предложенной технологии мировая система координат связана с моделью МКС, а модель Земли перемещается и изменяет ориентацию таким образом, чтобы смоделировать полет вдоль заданного участка суточной трассы. Созданы эффективные методы и алгоритмы задания границ произвольного участка суточной трассы, вычисления моментов времени начала и конца процесса моделирования, а также интерактивного отслеживания корректности задаваемых параметров орбитального наблюдателя. На основе предложенных технологии, методов и алгоритмов разработан интерфейс управления орбитальным наблюдателем, позволяющий инструктору задавать произвольные допустимые участки суточной трассы МКС, а также оперативно управлять сеансом орбитального наблюдения с помощью диалогового окна с «умным» контролем вводимых параметров и горячих клавиш. Предложенное решение было реализовано в программном комплексе моделирования и визуализации Земли, созданном в ФГУ ФНЦ НИИСИ РАН. Апробация комплекса подтвердила адекватность полученных результатов поставленной задаче. Полученные научные и практические результаты могут быть использованы при разработке космических видеотренажерных комплексов, построении имитационных стендов, при создании систем виртуального окружения, образовательных приложений и др.

Ключевые слова: моделирование динамики, суточные витки, мониторинг Земли, космический видеотренажер, графический интерфейс, контроль ввода данных

1 Введение

В настоящее время одним из важных направлений научно-технических исследований на борту российского сегмента Международной космической станции (РС МКС) является «Дистанционное зондирование Земли» [1]. Оно включает в себя большое число экспериментов по развитию средств и методов наблюдения Земли из космоса, по оценке зон экологических и стихийных бедствий и др. Примером является космический эксперимент «ЭКОН-М» [2], в котором космонавты выполняют экологическое обследование различных районов и объектов России и зарубежных государств, проводя сеансы визуально-приборных наблюдений (с помощью ручных оптических, инфракрасных и иных приборов). Объектами являются акватории рек, морей и океанов, инфраструктуры крупных городов, пожары, наводнения, извержения вулканов, аномальные явления в атмосфере и т.д.

Обследование объектов земной поверхности выполняется вдоль подспутниковой трассы МКС – совокупности подспутниковых точек, полученной в результате наложения витков МКС по орбите и вращения Земли вокруг своей оси. Для удобства рассматривают суточную трассу, состоящую из трасс 16 суточных витков (орбитальный период МКС ~1,5 часа), которые начинаются и заканчиваются на экваторе. На рисунке 1 показан пример участка трассы МКС для 1-3 суточных витков. В зависимости от текущего суточного витка космонавты выполняют обследование тех наземных объектов, которые находятся ближе к трассе этого витка. На борту станции имеется программное обеспечение «Сигма» баллистико-навигационного отображения полетной обстановки [2, 3], которое формирует оперативный прогноз времени прохождения и условий видимости заданных объектов наблюдения. Однако, даже при благоприятных условиях временное окно уверенного наблюдения заданного участка земной поверхности является крайне малым (50-80 с), ввиду чего космонавту необходимо тренировать профессиональные навыки выполнения таких операций еще на этапе наземной подготовки.

Эффективной является тренировка орбитальных наблюдений на трехмерной виртуальной модели Земли в космических видеотренажерах [4]. Для реализации такой тренировки необходимо, чтобы видеотренажер моделировал видимое движение («бег местности») Земли вдоль задаваемых инструктором участков

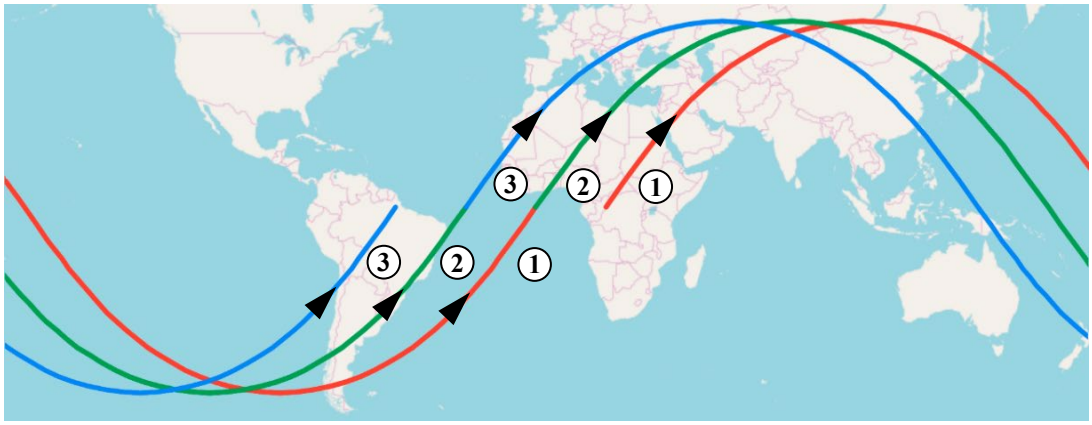


Рис. 1. Пример участка суточной трассы МКС (в изображении использована часть стандартного слоя карты © участники проекта OpenStreetMap, CC-BY-SA, www.openstreetmap.org)

суточной трассы МКС. В этой связи возникает задача реализации в вычислительной системе видеотренажера периодического (например, каждые 40 мс) расчета соответсвующих положений и ориентаций динамических виртуальных объектов в некоторой системе координат. Можно выделить два основных подхода к решению этой задачи.

Один подход состоит в вычислении в каждый момент времени положения и ориентации модели МКС на околоземной орбите в геоцентрической системе координат. Данный подход удобен для построения и анализа под-спутниковых трасс МКС, оценки границ зон обзора земной поверхности и других парамет-ров наблюдения [2, 3]. Однако, если таким способом выполнять моделирование «бега местности» в космическом видеотренажере, то погрешности вычисления положения модели МКС (и связанной с ней виртуальной камеры) становятся визуально заметными, что проявляется в виде дрожания синтезируемых изображений Земли.

Другой подход состоит в использовании для расчетов системы координат, связанной с МКС. Такой подход используется, например, при моделировании стыковки космического корабля (КК) и МКС, где необходима высокая точность вычисления позиций объектов (в миллиметрах). В рамках данного подхода авторами была предложена технология моделирования взаимного расположения орбитального наблюдателя и Земли [5], при котором модель МКС является неподвижной, а модель Земли перемещается и изменяет ориентацию таким образом, чтобы смоделировать полет вдоль подспутниковой трассы. Данная работа развивает предложенную технологию в части добавления нового этапа вычисления параметров начала и окончания пролета заданного произвольного участка суточной трассы МКС, а также создания интерфейса управления орбитальным наблюдателем с «умным» контролем вводимых параметров.

2 Технология моделирования «бега местности» Земли

Пусть имеется виртуальная сцена, включающая в себя трехмерные модели Земли, МКС и средства наблюдения (см. рис. 2), где:

- объектная система координат модели Земли (**Earth Object Coordinate System, EOCS**) совпадает с геоцентрической системой координат Земли;

- у объектной системы координат модели МКС (**ISS Object Coordinate System, IOCS**) центр совпадает с центром масс МКС, а оси направлены в соответствии с расположением строительных осей у реальной МКС;

- центр мировой системы координат (**World Coordinate System, WCS** [6]) совпадает с центром системы IOCS, а оси исходно совпадают с осями IOCS (модель МКС может менять свою ориентацию относительно системы WCS);

- позиция виртуальной камеры зафиксирована в некоторой точке модели МКС, а вектор взгляда камеры исходно направлен в центр модели Земли (камера может поворачиваться вокруг своих осей);

- объектная система координат модели средства наблюдения (**Camera Object Coordinate System, COCS**) жестко связана с виртуальной камерой.

Также заданы параметры, определяющие суточную трассу МКС: шесть параметров орбиты МКС (долгота Ω восходящего узла, наклонение i плоскости орбиты к плоскости экватора, аргумент ω перигея, эксцентриситет e, длина a большой полуоси и время t_π прохождения через перигей), угловая скорость ω_e вращения Земли вокруг своей оси и геоцентрическая долгота L_0 начала первого суточного витка (см. [5,7,8]).

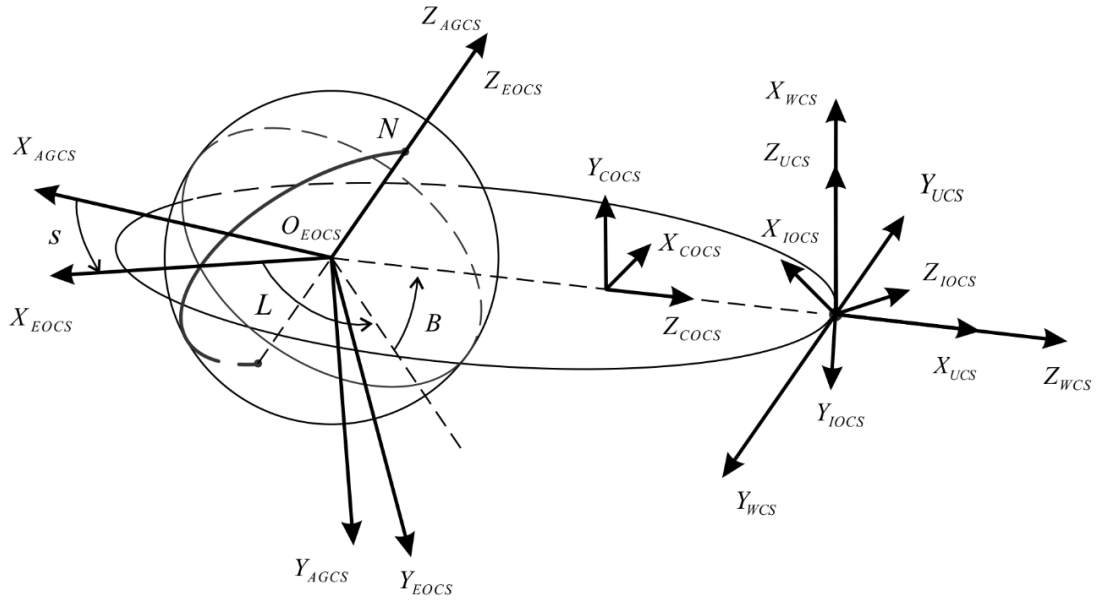


Рис. 2. Виртуальная сцена и системы координат

Рассмотрим задачу моделирования «бега местности» Земли вдоль некоторого произвольного участка суточной трассы МКС.

Из рисунка 1 видно, что любую точку суточной трассы МКС можно однозначно задать парой (k, L), где $k \in [1, 16]$ - номер суточного витка, на трассе которого лежит точка, а $L \in [-\pi, \pi]$ - ее геоцентрическая долгота (одной долготы L будет недостаточно, так как ей соответствует несколько точек суточной трассы). Обозначим через (k_s, L_s) и (k_e, L_e) начало и конец рассматриваемого участка суточной трассы ($k_s \geq k_e$), а через T_{orb} – орбитальный период [7] МКС:

$$T_{orb} = 2\pi\sqrt{a^3/b_0},$$

где $b_0 = 3{,}986 \cdot 10^{14}$ м^3/с^2 - гравитационный параметр Земли, равный произведению гравитационной постоянной на массу Земли. Введем ось Q времени полета орбитального наблюдателя вдоль суточной трассы МКС, на которой 0 соответствует подспутниковой точке (1, L_0) (начало 1-го витка), а отметка $16T_{orb}$ – концу суточной трассы (см. рис. 3). На оси Q парам (k_s, L_s) и (k_e, L_e) будут соответствовать некоторые моменты времени t_s и t_e начала и конца процесса моделирования «бега местности».

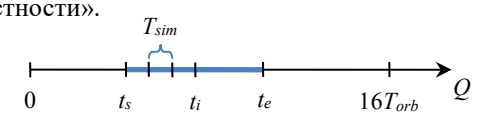


Рис. 3. Ось времени полета вдоль суточной трассы МКС

Рассмотрим некоторый i-ый шаг моделирования. Этому шагу на оси Q будет соответствовать момент времени t_i из отрезка $[t_s, t_e]$, такой что:

$$t_i = t_s + min(iT_{sim}, t_s - t_e), \quad (1)$$

где T_{sim} - периодичность расчета положений и ориентаций динамических объектов в видеотренажере (в миллисекундах), а $t_0 = t_s$. В момент времени t_i модель МКС будет располагаться в некоторой точке орбиты, иметь какую-то ориентацию и виртуальная камера будет направлена на модель Земли. Для получения изображения Земли в виртуальной камере необходимо вычислить координаты любой вершины P модели Земли в системе COCS средства наблюдения. Обозначим через P_{EOCS} и P_{COCS} координаты P в системах EOCS и COCS. Тогда рассматриваемая задача сводится к вычислению матрицы $M_{E \to C}$ перехода от системы координат EOCS к системе координат COCS, так что будем иметь:

$$P_{COCS} = M_{E \to C} P_{EOCS}.$$

Предлагаемая технология решения этой задачи состоит из следующих двух этапов.

На *первом этапе* (стадия задания параметров сеанса наблюдения) вычисляются моменты времени t_s и t_e начала и конца полета над участком суточной трассы, заданным парами (k_s, L_s) и (k_e, L_e).

На *втором этапе* (стадия визуализации) в каждый t_i момент времени (см. выражение (1)) вычисляется матрица $M_{E \to C}$. На данном этапе матрица $M_{E \to C}$ формируется на основе подхода, описанного в работе [5], при котором переход из системы EOCS Земли в систему COCS средства наблюдения выполняется через ряд промежуточных систем координат:

$$M_{E \to C} = M_{I \to C} M_{W \to I} M_{U \to W} M_{A \to U} M_{E \to A}, \quad (2)$$

где $M_{E \to A}$ - матрица перехода от системы EOCS

к абсолютной геоцентрической системе координат (AGCS), $M_{A\to U}$ - от системы AGCS к унифицированной системе координат (UCS), $M_{U\to W}$ - от системы UCS к системе WCS, $M_{W\to I}$ - от системы WCS к системе IOCS, $M_{I\to C}$ - от системы IOCS к системе COCS. Описание приведенных матриц можно найти в работе [5].

Далее расмотрим более подробно первый этап предложенной технологии.

3 Вычисление моментов времени начала и конца полета над участком суточной трассы

Сформулируем задачу вычисления t_s и t_e в общем виде, как задачу вычисления момента времени t_P пролета над подспутниковой точкой P, заданной парой (k_P, L_P). Она является частным случаем обратной задачи к задаче вычисления геоцентрических широты и долготы подспутниковой точки P в заданный момент времени t_P, основанной на численном решении уравнения Кеплера [8].

Обозначим через $L(t)$ функцию вычисления геоцентрической долготы подспутниковой точки в некоторый момент времени t из отрезка $[0, 16T_{orb}]$, а через $t_{s,k_P} = (k_P - 1)T_{orb}$ и $t_{e,k_P} = k_P T_{orb}$ - моменты времени начала и конца k_P-го суточного витка. Далее найдем функцию $L(t)$ и вычислим момент времени t_P численным приближением $L(t)$ к значению L_P на отрезке $[t_{s,k_P}, t_{e,k_P}]$.

Нахождение функции $L(t)$. Обозначим через $M_{E\to U}$ произведение матриц $M_{A\to U}M_{E\to A}$ из выражения (2). Согласно исследованию [5] для матрицы $M_{E\to U}$ можно записать следующее выражение:

$$M_{E\to U} = T_{\boldsymbol{d}} R_{Z,u}^T R_{X,i}^T R_{Z,\Omega}^T R_{Z,s}, \qquad (3)$$

где $R_{Z,s}$ - матрица поворота пространства [6] вокруг оси $O_{EOCS}Z_{EOCS}$ на угол $s(t)=\Omega - L_0+\omega_e(t+T_{orb}-t_в)$ ($t_в$ - время первого прохождения МКС восходящего узла орбиты, см. [5]); $R_{Z,\Omega}$ и $R_{X,i}$ - матрицы поворота пространства вокруг осей $O_{UCS}Z_{UCS}$ и $O_{UCS}X_{UCS}$ на углы Ω и i соответственно; $R_{Z,u}$ - матрица поворота пространства вокруг оси $O_{UCS}Z_{UCS}$ на угол (аргумент широты) $u(t)=\omega+\vartheta(t)$ (истинная аномалия $\vartheta(t)$ находится на основе численного решения уравнения Кеплера, см. [7, 8]); $T_{\boldsymbol{d}}$ - матрица переноса пространства [6] на вектор $\boldsymbol{d}$ $=(-r(t), 0, 0, 0)$ ($r(t) = a\,(1 - e^2) / (1 + e\cos\vartheta(t)$, см. [7, 8]).

Согласно формуле (3) матрицу $M_{E\to U}$ можно представить в виде произведения двух матриц $T_{\boldsymbol{d}}R$, где матрица $R = R_{Z,u}^T R_{X,i}^T R_{Z,\Omega}^T R_{Z,s}$ определяет ориентацию модели Земли (жестко связанной с ней системой EOCS) в системе UCS. Также ориентацию модели Земли в системе UCS можно задать с помощью последовательных поворотов системы EOCS на углы Эйлера-Крылова θ, ψ и φ, при которых углы φ и ψ сохраняют связь с геоцентрической долготой L и широтой B подспутниковой точки модели МКС. Представим последовательность поворотов по этим углам в виде матрицы R' ориентации модели Земли и из равенства $R' = R$ найдем искомую функцию $L(t)$.

В данной работе считается, что углы Эйлера-Крылова – это повороты системы координат вокруг базисных векторов по часовой стрелке, если смотреть из конца вектора, вокруг которого выполняется поворот. Для задания ориентации модели Земли используем следующую схему поворотов:

- поворот на угол θ вокруг оси $O_{UCS}X_{UCS}$;
- поворот на угол $\psi = -B$ вокруг оси $O_{UCS}Y_{UCS}$, полученной после первого поворота системы UCS;
- поворот на угол $\varphi = L$ вокруг оси $O_{UCS}Z_{UCS}$, полученной после второго поворота системы UCS. Согласно данной схеме запишем следующее выражение для матрицы R' ориентации модели Земли:

$$R' = R_{X,-\theta}R_{Y,-\psi}R_{Z,-\varphi}.$$

Чтобы восстановить угол φ из матрицы R, используем подход, описанный в работе [9]. Выпишем из матрицы R элементы a_{11}, a_{12}, a_{13}:
$a_{11} = \cos(\Omega - s(t))\cos u(t) - \sin(\Omega - s(t))\cos i \sin u(t)$,
$a_{12} = \sin(\Omega - s(t))\cos u(t) + \cos(\Omega - s(t))\cos i \sin u(t)$,
$a_{13} = \sin u(t) \sin i$.

Приравняв a_{11}, a_{12} и a_{13} к аналогичным элементам матрицы R', получим следующую систему уравнений:

$$\begin{cases} \cos\varphi(t)\cos\psi(t) = a_{11} \\ \sin\varphi(t)\cos\psi(t) = a_{12} \\ -\sin\psi(t) = a_{13} \end{cases}.$$

В рассматриваемой задаче орбита не является полярной ($i \neq \pi/2$), т.е. $-\pi/2 < \psi(t) < \pi/2$ и $\cos\psi(t) > 0$. Отсюда следует:

$$\begin{cases} \cos\varphi(t) = a_{11}/\cos\psi(t) \\ \sin\varphi(t) = a_{12}/\cos\psi(t) \\ \cos\psi(t) = \sqrt{1 - a_{13}^2} \end{cases}. \qquad (4)$$

Учитывая $\varphi = L$, получим из (4) искомую функцию $L(t)$:

$$L(t) = \begin{cases} \arccos\left(a_{11}/\sqrt{1 - a_{13}^2}\right), & a_{12} \geq 0 \\ -\arccos\left(a_{11}/\sqrt{1 - a_{13}^2}\right), & a_{12} < 0 \end{cases}. \qquad (5)$$

Вычисление момента времени t_P в данной работе выполняется численным приближением функции $L(t)$ из (5) к значению L_P на отрезке $[t_{s,k_P}, t_{e,k_P}]$ с точностью до погрешности ε машинного представления вещественных

чисел. Приближение выполняется с помощью разработанной модификации метода бисекции [10], в которой выбор половины временного отрезка выполняется на основе модуля разницы геоцентрической долготы середины отрезка и L_P. Это реализует следующий алгоритм:

1. Выполним приведение L_P к диапазону $[0, 2\pi)$, при котором $[0, \pi] \to [0, \pi]$, а $[-\pi, 0] \to [\pi, 2\pi)$:

$$L'_P = L_P - 2\pi\lfloor L_P/2\pi \rfloor. \quad (6)$$

2. Проверим границы t_{s,k_P} и t_{e,k_P} на совпадение с искомым t_P:

Если $\left|L'_P - L\left(t_{s,k_P}\right)\right| \le \varepsilon$, то:

$t_P = t_{s,k_P}$ и выходим из алгоритма.

Аналогично проверим границу t_{e,k_P}.

3. Вычислим приближенное значение t_P:

$t_A = t_{s,k_P}$, $t_B = t_{e,k_P}$, $t_P = 0$, $t_{prev} = 0$, $L_{cur} = 0$.

Цикл

$t_{prev} = t_P$, $t_P = 0.5(t_A + t_B)$, $L_{cur} = L(t_P)$.

$\Delta L_1 = |L_{cur} - L'_P|$, $\Delta L_2 = 2\pi - \Delta L_1$.

Если $\Delta L_1 < \Delta L_2$, то:

Если $L_{cur} > L'_P$, то $t_B = t_P$.

В противном случае, $t_A = t_P$.

В противном случае:

Если $L_{cur} > L'_P$, то $t_A = t_P$.

В противном случае, $t_B = t_P$.

пока $(|L_{cur} - L'_P| > \varepsilon)$ && $(|t_P - t_{prev}| > \varepsilon)$.

Конец цикла.

Конец алгоритма.

Отметим, что в пп. 2 и 3 алгоритма вычисленные значения функции $L(t)$ также приводятся к диапазону $[0, 2\pi)$ согласно формуле (6). Выполнив приведенный алгоритм для пар (k_s, L_s) и (k_e, L_e), получим искомые значения моментов времени t_s и t_e.

4 Интерфейс управления орбитальным наблюдателем

На основе предложенных технологии, методов и алгоритмов был разработан интерфейс управления орбитальным наблюдателем. Интерфейс включает в себя диалоговое окно сеанса орбитального наблюдения модели Земли и горячие клавиши оперативного управления ходом визуализации.

Диалоговое окно включает в себя блоки полей ввода значений параметров орбиты МКС, модели Земли, границ участка суточной трассы, а также параметров управления камерой и временем моделирования (см. рис. 4). Исходные значения этих параметров считываются из конфигурационного файла при запуске системы визуализации. В процессе визуализации пользователь (инструктор) может вызывать диалоговое окно, изменять в нем значения параметров, и применив их, немедленно видеть соответствующие изменения визуальной картины.

Для обеспечения ввода корректных значений параметров наблюдателя был разработан метод «умного» интерактивного отслеживания вводимых данных, включающий в себя контроль типа вводимого параметра, допустимых элементов синтаксиса, выхода за границы допустимых диапазонов значений. В случае ввода некорректного значения рядом с полем всплывает подсказка с информацией об ошибке и рекомендацией по ее исправлению. Если корректное значение так и не было получено, то в поле фиксируется последнее успешно введенное значение, в том числе, из конфигурационного файла. Также в предложенном методе реализован связанный контроль пар «номер суточного витка, геоцентрическая долгота»: изменение номера витка приводит к изменению диапазонов допустимых значений долготы, а изменение номера начального витка ограничивает диапазон допустимых номеров конечного витка. На рисунке 4 изображен момент ввода некорректной долготы начала участка суточной трассы и вывод подсказки с диапазоном допустимых значений.

Для обеспечения возможности оперативного просмотра отдельных участков полета, их повтора, разбора сложных тренировочных ситуаций и др. в созданном интерфейсе реализован режим управления скоростью течения модельного времени с помощью горячих клавиш. В частности, реализованы: ускорение, замедление, остановка, обратное течение времени (повторное нажатие соответствующей клавиши увеличивает/уменьшает скорость времени),

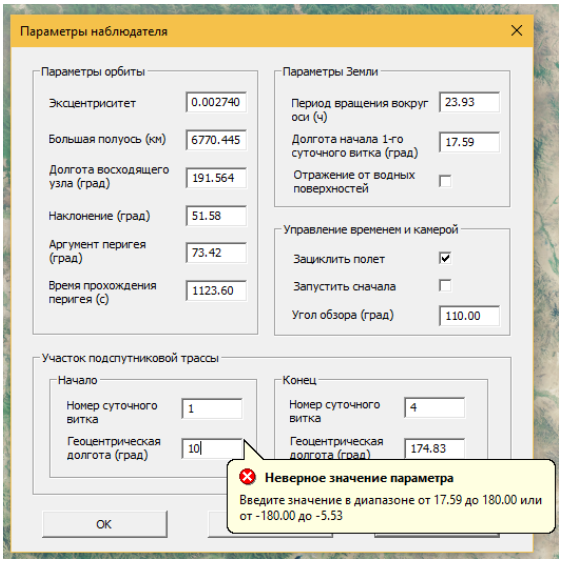


Рис. 4. Диалоговое окно сеанса орбитального наблюдения

восстановление реальной скорости течения времени, возврат в начало заданного участка трассы и перемещение в его конец.

5 Заключение

В данной работе была рассмотрена задача реализации орбитальных наблюдений трехмерной виртуальной модели Земли в космических видеотренажерах. Была предложена модифицированная технология моделирования вида Земли в орбитальном средстве наблюдения, которая обеспечивает видимое движение земной поверхности вдоль произвольного заданного участка суточной трассы МКС. Предложенная технология включает в себя новый этап вычисления моментов времени начала и окончания полета над заданным участком трассы, а также этап вычисления координат модели Земли в системе координат модели средства наблюдения, связанной с моделью МКС. Были созданы эффективные методы и алгоритмы задания участка трассы с помощью пар типа «номер суточного витка, геоцентрическая долгота подспутниковой точки», контроля корректности этих пар, а также вычисления по ним моментов времени начала и конца процесса моделирования. На основе предложенных технологии, методов и алгоритмов разработан эффективный интерфейс управления орбитальным наблюдателем, включающий диалоговое коно и горячие клавиши, которые позволяет инструктору оперативно управлять сеансом орбитального наблюдения (вводить новые участки суточной трассы, зацикливать, ускорять/замедлять, останавливать, «отматывать» назад «бег местности» и др.).

Предложенное решение было реализовано в программном комплексе построения и визуализации детализированной трехмерной виртуальной модели Земли, разработанном в ФГУ ФНЦ НИИСИ РАН. Была проведена апробация комплекса, при которой имитировался полет вдоль ряда объектов земной поверхности, обследуемых с борта МКС (оз. Байкал, Асуанская плотина, о. Кипр и др.). На рисунке 5 приведен пример кадра полученной визуализации модели подстилающей земной поверхности вдоль участка трассы первого суточного витка орбиты МКС (центр изображения соответствует подспутниковой точке с координатами 51,43°с.ш. и 107,51° в.д.). Проведенная апробация подтвердила адекватность полученных научных и практических результатов поставленной задаче и возможность их применения при построении космических видеотренажеров, имитационных стендов, систем виртуального окружения, информационно-образовательных приложений и др.

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 17-07-00243.

Рис. 5. Пример кадра полученной визуализации модели подстилающей поверхности Земли

Simulation of visible Earth motion along daily tracks of ISS orbit in space simulators

P.Yu. Timokhin

Abstract. The paper considers the task of implementing of orbital observations on 3D virtual model of the Earth in space video simulators. A modified technology to simulate visible motion of the Earth's surface along daily orbit tracks of the International Space Station (ISS) is proposed. In the proposed technology, the world coordinate system is bound with ISS model, and Earth model is placed and rotated in such a way as to simulate a flight along a given section of daily orbit track. There were created effective methods and algorithms to specify the boundaries of an arbitrary section of daily orbit track, to calculate time points of starting and ending of simulation process, as well as to interactively track the correctness of setting parameters of orbital observer. Based on proposed technology, methods and algorithms, the orbital observer control interface was developed allowing the instructor to set arbitrary valid orbit track sections, as well as operatively control the orbital observation session by means of developed dialog box with “smart” validation of input parameters and hot keys. The proposed solution was implemented in program complex of modeling and visualization of the Earth, created at SRISA RAS. Testing of the complex confirmed the adequacy of obtained results to the task. The obtained scientific and practical results can be used in the development of space video simulators, construction of simulation stands for conducting space experiments, in virtual environment systems, educational applications, etc.

Keywords: dynamics simulation, orbit tracks, Earth observation, space simulator, graphical interface, input data validation

Литература

1. Направления научно-технических исследований на борту российского сегмента МКС. http://www.gctc.ru/main.php?id=278.

2. Г.Д. Орешкин, А.Н. Ядренцев, А.В. Севериненко. Результаты выполнения программы КЭ «ЭКОН-М» космонавтом А.А. Мисуркиным в составе экипажа МКС-53/54. «Пилотируемые полеты в космос», 2018, № 3(28), 34–44.

3. РС МКС. Справочник пользователя. «РКК «Энергия» им. С. П. Королева»», Королев: 2016. https://www.energia.ru/ru/iss/researches/iss-researches.html.

4. А.И. Митин, В.И. Брагин. Пути повышения адекватности моделирования визуальных условий мониторинга земной поверхности на тренажере служебного модуля российского сегмента Международной космической станции. «Пилотируемые полеты в космос», 2014, № 3(12), 60–70.

5. М.В. Михайлюк, П.Ю. Тимохин. Моделирование взаимного расположения планеты и спутника в космических видеотренажерах. «Mathematica Montisnigri», V. XXIX (2014), 91–97.

6. J.J. McConnell. Computer Graphics: Theory Into Practice. Jones and Bartlett Publishers, 2005.

7. Г.Н. Дубошин. Справочное руководство по небесной механике и астродинамике. М., Наука, 1976.

8. J. Meeus. Astronomical algorithms. Willmann-Bell, 1991.

9. K. Shoemake. Animating Rotation with Quaternion Curves. «Computer Graphics, SIGGRAPH Conference Proceedings, San-Francisco, July 22-26, 1985», V. 19 (1985), №3, 245–254.

10. Е.А. Волков. Численные методы. М., Наука, 1987.

Автоматическая разметка кадров видеопотока для машинного обучения

Н.О. Бесшапошников[1], М.С. Дьяченко[1], М.А. Кузьменко[1,2], А.Г. Леонов[1,2,3], М.А. Матюшин[1,2], К.А. Прокин[1]

[1] ФГУ ФНЦ НИИСИ РАН, Москва, Россия
[2] МГУ им. М.В.Ломоносова, Москва, Россия
[3] МПГУ, Москва Россия

E-mail's: nbesshaposhnikov@vip.niisi.ru, mdyachenko@niisi.ru, gmk.@vip.niisi.ru, dr.l@vip.niisi.ru, itsaprank@yandex.ru, prokin@vip.niisi.ru

Аннотация. В задаче машинного обучения с учителем всегда важным является вопрос, на каких данных планируется обучение модели, и, когда данные уже выбраны, встает вопрос о разметке этих данных. В большинстве случаев, если предполагается улучшение имеющихся результатов работы некоторого алгоритма, или даже работа над совершенно новой неизученной областью, нередко выходом является ручная разметка набора данных. Однако, данный процесс может быть весьма трудоемким, особенно, когда решаемая задача связана с детектированием объектов. В данной статье приводится описание подхода к сбору и разметке обучающего набора данных, позволяющего существенно снизить трудозатраты на этот процесс.

Ключевые слова: машинное обучение, разметка данных, автоматизирование.

1.Введение

В настоящее время ведутся активные исследования в области машинного обучения. И хотя открытые размеченные обучающие наборы данных доступны каждому, а их объем, благодаря усилиям, приложенным энтузиастами и крупными корпорациями, исчисляется сотнями тысяч примеров, в связи с широким многообразием информации, доступной человеку в современном мире, почти каждая прикладная задача, использующая методы машинного обучения, требует сбора и подготовки, разметки обучающего набора данных. В связи с этим, а также с непрерывным устареванием информации, остро стоит проблема сбора и разметки новых обучающих наборов данных. Поскольку разметка данных для машинного обучения, и в особенности, разметка изображений, является трудоемкой задачей, требующей длительного труда, авторами данной статьи была предпринята попытка построения алгоритма, позволяющего частично или полностью автоматизировать процесс сбора и разметки обучающего набора данных для прикладной задачи машинного обучения.

Стоит отметить, что в настоящее время разметка обучающих наборов изображений производится преимущественно вручную, существуют интернет сервисы, предоставляющие удобный инструментарий, помогающий размечать изображения (например, CVAT от Intel [1]), услуги по ручной разметке наборов данных [2] либо имплементирующие так называемые алгоритмы слабой разметки, использующие генеративные модели для создания неточной разметки с помощью предопределенных правил (например, Snorkel [3]). Таким образом, в настоящее время методы разметки разметки изображений делятся на три категории:

1) ручные, трудоёмкие
2) ручные, дорогие
3) автоматизированные, генерирующие зашумленные данные

Целью данной статьи является презентация алгоритма сбора и разметки изображений для обучающих наборов данных, позволяющих практически полностью автоматизировать процесс разметки, и дающих на выходе результат высокого качества.

2. Описание алгоритма

Ранее [4] было показано, как при помощи технологий OpenCV можно сократить трудозатраты на разметку обучающего набора данных до разметки четырех ключевых точек на изображение. Данная статья является логическим завершением [4], и описывает алгоритм, позволяющий полностью избавиться от необходимости

разметки всех изображений кроме очень малого их числа.

Допустим, что все изображения, которые у нас есть, разделены на N групп, и в каждой из групп изображения можно выстроить в такую последовательность, что расстояния между двумя соседними изображениями будут малы (такие изображения можно получить, например, производя непрерывную раскадровку N видеопотоков, содержащих целевые объекты обучения). Выделим в каждой из N последовательностей начальное изображение и разметим его полностью. Из всех целевых объектов обучения, найденных на начальном изображении, выберем 1-3 объекта, все точки которых на всех кадрах группы образуют неподвижное тело и находятся в пространстве в некоторой малой окрестности некоторой неподвижной плоскости. Назовем их якорными объектами. Далее, рассматривая начальное и следующее за ним в последовательности изображение, воспользуемся алгоритмом SURF [5], чтобы найти ключевые точки на обоих изображениях.

Из всех ключевых точек, найденных на начальном изображении, оставим только те, которые принадлежат якорным объектам.

Теперь, ставя каждой ключевой точке, найденной на начальном изображении, в соответствие ближайшую к ней ключевую точку, найденную на втором изображении, получаем два набора точек, которые, предположительно, переходят один в другой под действием некоторого непрерывного проективного преобразования, описывающего движение камеры.

Чтобы восстановить данное проективное преобразования, достаточно иметь по 4 ключевые точки на каждом из изображений, как это описано в [4]. В случае работы с SURF, как правило, имеется гораздо большее число ключевых точек. В этом случае в качестве результирующего отображения берется среднее по всем четырехточечным отображениям.

Имея проективное преобразование, точно описывающее движение камеры между начальным и следующим за ним изображениями, применяем его к разметке начального изображения. Если расстояние между изображениями действительно мало, то полученная разметка будет корректна для второго изображения. Повторяя описанный алгоритм, применительно ко второму и третьему, затем к третьему и четвертому, и так далее, изображениям группы, получим в результате корректную разметку всей группы изображений.

Данный метод, как несложно понять, позволяет разметить практически неограниченное количество изображений для дальнейшего обучения модели. Однако, в силу того, что полноценно эвристически восстановить карту глубины изображения не представляется возможным, описанный метод является наиболее устойчивым в случае, когда целевой объект один на каждую группу изображений, и представляет собой некоторый плоский предмет.

Схема алгоритма, осуществляющего одну итерацию разметки, то есть переход от одного изображения к другому, приведена ниже.

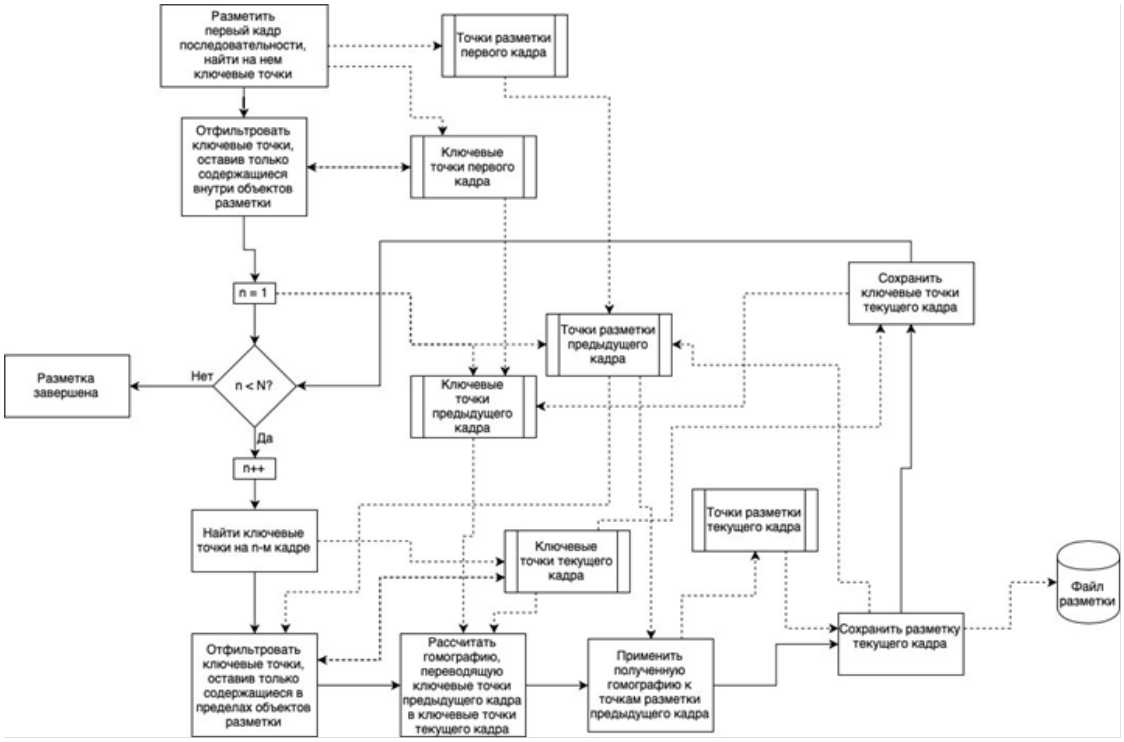


Рис.1. Алгоритм автоматической разметки для кадров одной последовательности

3. Оценка качества

Ясно, что описанный алгоритм дает большой выигрыш в скорости разметки, однако, учитывая, что в полученном наборе данных большинство изображений будут в известном смысле близки друг к другу, возникает естественный вопрос о качестве полученного набора данных в приложении к реальному машинному обучению. Стоит сразу отметить, что набор данных, собранный описанным алгоритмом, является более широким, чем полученный с помощью аугментации [6], поскольку любой метод аугментации не добавляет в данные информации контекст, в то время как наши изображения, полученные из видеопотока, контекстно различны.

С другой стороны, набор данных, полученный согласно описанному алгоритму, очевидно, более узкий, чем "честный" набор данных, собранный по одному изображению и размеченный вручную, поскольку наш набор данных очень легко кластеризуется на изначальные группы изображений, внутри которых происходила автоматизация процесса разметки. Встает закономерный вопрос, какое влияние оказывает данная особенность при использовании непосредственно в машинном обучении. Ответ на данный вопрос дает проведенный нами эксперимент.

С целью проверки качества описанного алгоритма был проведен эксперимент по обучению модели детектирования объектов на наборе данных, собранном и размеченном описанным в данной статье способом. В качестве целевого объекта обучения был выбран дорожный знак пешеходного перехода, в качестве тестовой модели нейронная сеть Faster RCNN Inception v2 [7], предобученная на открытом наборе данных COCO [8]. Выходные границы объектов обрабатывались алгоритмом Non-Maximum Suppression [9]. Обучающий набор данных состоял из 17 видео, содержащих различные знаки пешеходного перехода, и размеченный описанным выше алгоритмом. Разрешение изображений на входе нейронной сети составляло 1020 × 1020 пикселей. Объем получившегося набора данных составил 5526 изображений. Для валидационного набора данных были выбраны и сфотографированы знаки пешеходного перехода, не запечатленные в обучающих видео. Каждый знак был сфотографирован один-два раза. Объем валидационного набора данных составил 372 изображения. В качестве метрики качества была выбрана метрика *binary precision*, которая рассчитывается по следующей формуле:

$$bp = \frac{N_{good}}{N_{all}} \quad (1)$$

где N_{good} – количество изображений, на которых были задетектированы все целевые объекты, и не было задетектировано лишних объектов. Объект считается задетектированным, если метрика близости между его границей и известной границей IoU [10] составляет не менее 0.9, объект правильно классифицирован, и для данного целевого объекта нет повторных детекций, успешных или неуспешных. N_{all} – общее количество изображений.

Указанная модель обучалась в течение 40 эпох. В ходе обучения тренировочный набор данных подвергался стандартной процедуре аугментации, содержащей случайные афинные преобразования, изменения разрешения, добавления шумов и разноцветных заплаток, незначительное дрожание точек разметки, горизонтальные и вертикальные отражения. Тестовый набор данных, предназначавшийся для отбора лучшей версии весов модели, состоял из 30% обучающего набора данных.

По окончании обучения показатель метрики качества на тестовом наборе данных составлял 0.997. Показатель метрики качества на валидационном наборе данных составил 0.631, что является довольно неплохим показателем для таких условий эксперимента, и позволяет полагать, что описанный алгоритм пригоден для проведения разметки данных с целью последующего обучения модели. Подробные результаты эксперимента приведены

Таблица 1. Результаты эксперимента

Название набора	**Источник**	**Способ разметки**	**Объем**	**Binary precision**
Тренировочный	Видео	Автоматическая	3864	–
Тестовый	Видео	Автоматическая	1656	0.997
Валидационный	Отдельные фото	Ручная	372	0.631

4. Заключение

Из сказанного выше следует, что предложенный в данной статье алгоритм позволяет существенно ускорить сбор и разметку обучающего набора данных для прикладных задач, использующих технологии машинного обучения в области компьютерного зрения. С целью подтверждения работоспособности алгоритма был проведен показательный эксперимент, в ходе которого изучалась применимость описанного подхода к прикладной задаче детекции дорожного знака пешеходного перехода. Снижение качества на валидационной выборке может быть обусловлено её чрезмерной сложностью по сравнению с тренировочной выборкой, равно как и изменившимися в ходе сбора погодными условиями, степени загрязнения лобового стекла (мы нашли, что данный фактор оказывает существенное влияние на качество обучения конечной модели) и т.п. Помимо описанного в разделе 3 эксперимента данный метод также использовался в рамках трех разных прикладных задач детекции, где прекрасно себя зарекомендовал, показывая высокое качество детекции конечной модели на валидационной выборке.

Работа выполнена по теме Гранта РФФИ 18-07-00901 «Исследование и разработка системы распознавания элементов рукотворного интерьера на базе нейронных сетей для построения дополненной реальности и выработки алгоритмов взаимодействия управляемых объектов с реально-виртуальным окружением».

Automatic video frame markup for machine learning

N.O. Beshaposhnikov, M.S. Dyachenko, M.A. Kuzmenko, A.G. Leonov, M.A. Matushin, K.A. Prokin

Abstract. In supervised machine learning, there is always a need to markup training data. In most cases, if existing solutions shall be finetuned or even new solution shall be trained from scratch, often researchers shall organize markup of the data set. However, this process can be very time-consuming, mainly when the problem area is associated with the detection of objects. This article describes the approach to collecting and marking a training data set; the approach can significantly simplify and speed up the markup process.

Keywords: machine learning, data markup, automation.

Литература

1. Распространенное ПО для разметки от Intel. https://software.intel.com/en-us/articles/computer-vision-annotation-tool-a-universal-approach-to-data-annotation.

2. Amazon Mechanical Turok. https://www.mturk.com.

3. Snorkel. https://towardsdatascience.com/introducing-snorkel-27e4b0e6ecff.

4. Н.О. Бесшапошников, М.А. Кузьменко, А.Г. Леонов, М.А. Матюшин. Атоматизация разметки набора данных для нейронных сетей. «Вестник кибернетики», Т. 32 (2018), № 4, 204–210.

5. Herbert Bay Tinne Tuytelaars Luc Van Gool. SURF: Speeded up robust features. «Computer vision – ECCV», (2006).

6. Agnieszka Mikolajczyk, Michal Grochowski. Data augmentation for improving deep learning in image classification problem. «International Interdisciplinary PhD Workshop (IIPhDW)», Swinoujście, Poland, 09 - 12 May, Publisher: IEEE,2018, 117–122.

7. Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. https://arxiv.org/abs/1512.00567.

8. COCO Dataset. http://cocodataset.org.

9. J.Canny. A computation approach to edge detection. 1986.

10. Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, Silvio Savarese. Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression. «CVPR 2019 Expo», Long Beach, CA, 16 -20 June, 2019. https://arxiv.org/pdf/1512.00567v1.pdf.

Контейнеризация пользовательских заданий в суперкомпьютерной системе коллективного пользования

А.В. Баранов[1], Б.В. Долгов[2], А.В. Федотов[3]

[1]Межведомственный суперкомпьютерный центр РАН – филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия, antbar@mail.ru

[2]Межведомственный суперкомпьютерный центр РАН – филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия, borisdol@jscc.ru

[3] Московский физико-технический институт (государственный университет), Москва, Россия, andrey_university@mail.ru

Аннотация: В статье рассматривается задача представления суперкомпьютерных заданий в виде контейнеров. Необходимое для выполнения задания программное окружение в виде образа контейнера Docker помещается в репозиторий суперкомпьютерного центра. При запуске задания образ извлекается из репозитория и развертывается на выделенных заданию вычислительных модулях суперкомпьютера. На каждом модуле запускается отдельный экземпляр контейнера, который объединяется в единую виртуальную сеть с остальными экземплярами. В развернутой виртуальной среде запускаются процессы задания. В статье рассмотрена практическая реализация контейнеризации заданий для отечественной Системы управления прохождения параллельных заданий (СУППЗ), применяемой в Межведомственном суперкомпьютерном центре РАН. В рамках реализации предлагаемого подхода на суперкомпьютере МВС-10П ОП в составе задания СУППЗ была развернута виртуальная вычислительная инфраструктура на базе Sun Grid Engine.

Ключевые слова: суперкомпьютер, система управления заданиями, планирование заданий, контейнерная виртуализация, СУППЗ, Docker, SGE

1 Введение

Одной из главных тенденций развития суперкомпьютерных центров коллективного пользования (СКЦ) является расширение как спектра решаемых задач, так и круга пользователей. Достаточно часто новые пользователи приходят в суперкомпьютерные центры, имея в багаже технологию организации расчетов, сложившуюся исторически в течение нескольких лет. Такие пользователи могли долгое время применять для расчетов обычные серверы или рабочие станции, пока рост требований, например, к размеру расчетной сетки, точности моделирования и т.п., не привел к необходимости использования суперкомпьютерного оборудования для производства расчетов.

Суперкомпьютерные расчеты пользователи оформляют в виде вычислительных заданий, включающих расчетные программы и исходные данные, требования к характеру и объему запрашиваемых ресурсов. Специальные программные системы управления заданиями (СУЗ) [1] ведут очередь заданий, выделяют ресурсы и запускают на них пользовательские задания, контролируют их выполнение, высвобождают ресурсы после завершения заданий и предоставляют пользователю результаты расчетов. Другими словами, в суперкомпьютерных центрах принят определенный порядок обслуживания пользователей и их заданий. Часто этот порядок оказывается несовместим с исторически сложившейся технологией организации расчетов, которую новые пользователи применяли до обращения в суперкомпьютерный центр. Освоение принятого в СКЦ порядка работы в такой ситуации может повлечь необходимость полной реорганизации вычислительного процесса, вплоть до реинжиниринга программных кодов.

Подобная реорганизация вычислительного процесса может потребовать от пользователя в том числе отказа от работы с привычными программными средами и пакетами. Очевидно, что процесс перехода к расчетам в СКЦ будет связан со значительными затратами на переобучение сотрудников, а преимущества применения суперкомпьютерного оборудования в этом случае будут сведены на нет. Освоение суперкомпьютера новыми пользователями в такой ситуации будет проходить легче, если будут адаптированы для применения в СКЦ привычные для пользователя технология организации расчетов и соответствующие ей программная среда и пакеты.

Выходом видится применение технологий виртуализации для возможности переноса привычной пользователю программной среды в суперкомпьютерный центр. Как было показано в [2, 3], гипервизорная виртуализация вносит неоправданно высокие накладные расходы в процесс суперкомпьютерных вычислений, и по этой причине редко применяется в СКЦ. Более привлекательной с точки зрения накладных расходов выглядит контейнерная виртуализация [4], исследования в области применения которой в СКЦ активно ведутся в России [5, 6], в том числе и в Межведомственном суперкомпьютерном центре РАН (МСЦ РАН) [7, 8]. В работах [6, 8] в качестве одного из методов применения контейнерной виртуализации в СКЦ рассмотрена подготовка специального репозитория контейнеров с разными (в идеале – со всевозможными) стеками программного обеспечения ПО. Пользователь при запуске задания в СКЦ выбирает из репозитория наиболее подходящий для него контейнер. Отмечены [8] преимущества метода репозитория: простота и надежность реализации и высокая безопасность.

Настоящая работа посвящена практической реализации в МСЦ РАН метода контейнеризации заданий с использованием репозитория контейнеров, при помощи которого удалось адаптировать исторически сложившуюся в НИИСИ РАН технологию высокопроизводительных расчетов на базе платформы Sun Grid Engine.

2 Обработка заданий в СУППЗ

Суперкомпьютерные ресурсы состоят из многоядерных мультипроцессорных вычислительных модулей (ВМ), объединяемых между собой высокоскоростной коммуникационной средой. В требованиях пользовательского задания указываются, как минимум, число необходимых для расчетов ВМ и время, на которое ВМ будут предоставлены заданию. Особенность эксплуатации практически всех современных суперкомпьютеров состоит в том, что отдельный ВМ является атомарной единицей планирования ресурсов в СУЗ и не разделяется между различными заданиями.

Среди современных СУЗ лидирующие позиции занимают такие системы, как PBS [9], SLURM [10], LSF [11], Moab [12]. Все эти системы обладают развитыми механизмами планирования. Наибольшее распространение, благодаря своим качествам и открытому исходному коду, получила система SLURM. В МСЦ РАН на протяжении двух десятилетий в качестве СУЗ используется Система управления прохождением параллельных заданий (СУППЗ) [13], сравнение которой с известными СУЗ приводится в работах [14, 15].

Ядром СУППЗ является планировщик – сервер очередей, осуществляющий ведение очереди пользовательских заданий и составляющий расписание их запусков. После прохождения задания через очередь СУППЗ, оно запускается на выполнение сервером запуска, который осуществляет подбор и упорядочивание ВМ решающего поля для выполнения задания. Взаимодействие сервера запуска с компонентами суперкомпьютера производится с использованием системы командных сценариев. Перед началом выполнения задания СУППЗ инициализирует процедуру подготовки выделенных ВМ, а после окончания работы (или принудительной остановки) вызывается процедура очистки ВМ от последствий работы пользовательского задания.

После подготовки ВМ вызывается сценарий инициализации задания runtask. Сценарий предусматривает выполнение следующих действий:

- чтение параметров задания;
- запуск от имени пользователя командного сценария развертывания задания, который выполняется на первом по списку из выделенных заданию ВМ и осуществляет непосредственный запуск процессов задания на выделенных заданию ВМ решающего поля;
- ожидание завершения работы запущенного сценария развертывания задания.

После окончания работы сценария инициализации задания (runtask) или превышения указанного пользователем времени выполнения задания запускается процедура очистки выделенных заданию ресурсов.

Запускаемое рассмотренным выше образом задание будем называть стандартным. Стандартное задание представляет собой совокупность выполняющихся на разных ВМ процессов операционной системы Linux, взаимодействующих друг с другом посредством коммуникационной библиотеки (как правило, одной из реализаций MPI). Процессы стандартного задания выполняются в единой для всех заданий программной среде, включающей коммуникационные и иные библиотеки определенных реализаций и версий, установленные системным администратором суперкомпьютера. В стандартной среде затруднено не только применение альтернативных реализаций известных библиотек (например, различных вариантов MPI), но и отличающихся версий установленных библиотек. Как было отмечено

выше, эта проблема может быть устранена за счет помещения окружения процесса задания в контейнер и создания репозитория контейнеров для заданий.

3 Репозиторий контейнеров для заданий

На сегодняшний день для создания независимого и изолированного программного окружения широко применяется контейнерная виртуализация. Стандартом де-факто в этой области является программное обеспечение Docker [16], написанное на языке программирования Go компанией Docker. Применение технологии Docker позволяет упаковать процессы нестандартного задания с необходимым набором модулей и библиотек в единый контейнер – произвести создание Docker-образа с настроенным программным окружением задания. Далее на ВМ решающего поля при старте задания запускаются экземпляры контейнера из созданного образа, внутри которых функционируют необходимые процессы.

Для СУППЗ в этом случае необходимо разработать механизм запуска заданий, представленных в виде Docker-контейнеров. Несмотря на то, что подходы к решению этой задачи теоретически и экспериментально достаточно проработаны [7, 8], не было осуществлено их внедрение в реально работающую систему. Выделим ряд подзадач, которые необходимо решить для внедрения контейнерной виртуализации на базе технологии Docker в СУППЗ:

1. Обеспечить безопасность работы СУППЗ с технологией Docker.
2. Организовать сетевую связность контейнеров, расположенных на разных ВМ [7].
3. Создать Docker-репозиторий для загрузки образов заданий на ВМ.
4. Разработать процедуры запуска процессов задания на ВМ из образа контейнера при старте задания и остановки контейнеров после завершения задания.

На рисунке 1 представлена суперкомпьютерная система с Docker-репозиторием. Все ВМ решающего поля находятся в одной сети с репозиторием и могут загрузить из него необходимый Docker-образ. Для загрузки образа на ВМ на нем должна быть выполнена команда:

```
docker pull repository:port/ImageName
```

После выполнения команды начинается загрузка образа. Загрузка может производиться несколькими ВМ одновременно.

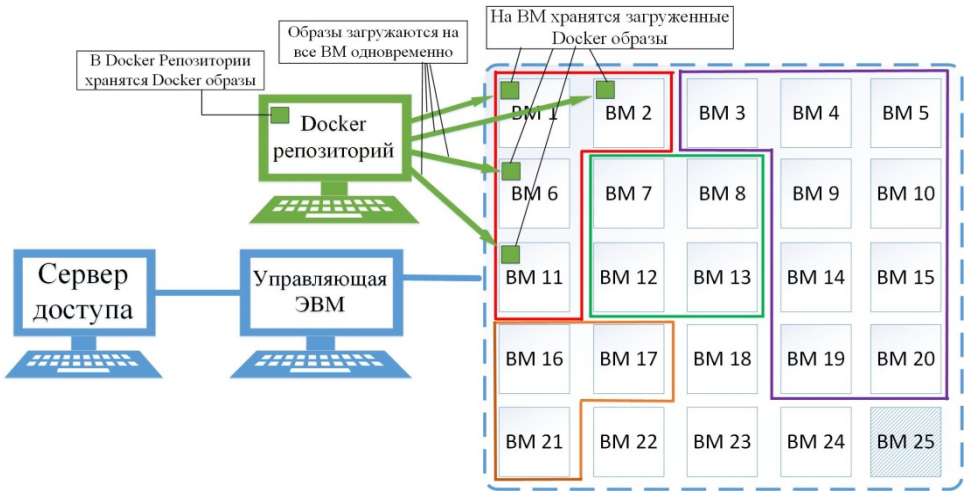


Рис. 1. Репозиторий Docker-контейнеров для заданий

Загрузка Docker-образов на ВМ и их конфигурирование может быть проведено на стадии подготовки ВМ, и этот функционал был реализован в соответствующем сценарии СУППЗ. Остановка и последующее удаление контейнеров с ВМ должны быть проведены по завершении задания, этого можно добиться, внеся изменения в командный сценарий очистки ВМ.

Запуск контейнеров и развертывание в них процессов задания на выделенных ВМ логично вынести в сценарий развертывания задания runmvs.bat, который запускается от имени пользователя на первом по списку из выделенных ВМ. Однако, запуск контейнеров подразумевает работу от имени суперпользователя, поэтому было решено внести изменения в сценарий инициализации задания runtask, в котором от имени

суперпользователя производится запуск контейнеров.

Для развертывания Docker-репозитория был выделен отдельный ВМ, к которому имеют доступ все остальные ВМ суперкомпьютера. Запуск Docker-репозитория производится путем запуска специального контейнера из образа под названием “registry:v2”. Образ “registry:v2” может быть загружен на узел с помощью команды

```
docker pull registry:v2
```

или при запуске контейнера с указанием данного образа:

```
docker run registry:v2
```

Для исключения возможности подмены пользователями загруженных в репозиторий образов на собственные и дальнейшего их использования на ВМ, существует возможность авторизации в репозитории с помощью сертификатов, которые могут быть созданы с использованием openssl, или с помощью логина и пароля пользователя.

Для загрузки новых образов в репозиторий администратору необходимо использовать команду

```
docker push reposAddr:port/ImageName
```

где reposAddr – адрес репозитория (ip-адрес, доменное имя), port – порт, открытый для работы с репозиторием на ВМ с адресом reposAddr, imageName – имя нового образа контейнера. Перед загрузкой необходимо пройти авторизацию в репозитории с помощью команды

```
docker login
```

Для взаимодействия с Docker-репозиторием в СУППЗ должна храниться информация об адресе репозитория, именах загружаемых из него образов контейнеров, и сертификатах для авторизации в репозитории. В СУППЗ присутствует возможность хранения указанной информации в паспорте задания. В паспорт задания была добавлена секция [Docker], в которой пользователь указывает тип контейнера, название образа в репозитории и другую сопутствующую информацию. Имя самого репозитория в целях безопасности определяется администратором в конфигурационном файле СУППЗ.

Вопросы безопасности, связанные с применением технологии Docker, подробно рассмотрены в работе [8]. Основная проблема состоит в том, что только суперпользователь и пользователь, находящийся в группе docker на ВМ, имеют возможность запускать контейнеры и управлять ими. Если у пользователя будет возможность собственноручно производить запуск контейнеров, то контейнер может быть запущен с правами суперпользователя (и, следовательно, с правами любых других пользователей) пользователем, не имеющим таких привилегий, что является недопустимым в системе коллективного пользования. Вариант с добавлением пользователя в группу docker на ВМ недопустим по этой же причине.

Запуск пользовательских процессов (или контейнеров) от имени суперпользователя root на выделенных ВМ является нетипичной задачей для СУППЗ, и для реализации этой операции были внесены изменения в используемые СУППЗ командные файлы.

Для безопасного старта контейнеров на ВМ при запуске задания СУППЗ производит следующие операции после загрузки образов на узлы:

1. Проверяет наличие прав у пользователя на чтение и запись в рабочий каталог задания.
2. Добавляет в Docker-образ пользователя с теми же идентификаторами (UID, GID) что и у пользователя, запустившего задание.
3. Предоставляет права на определенные каталоги внутри образа созданному в Docker-образе пользователю.
4. Производит запуск контейнера от имени созданного пользователя.

Стоит отметить, что пользователь не имеет возможности каким-либо образом подключиться к созданным контейнерам или иным образом повлиять на их работу, и единственным способом взаимодействия пользователя с созданными контейнерами является специально подключенный сетевой каталог.

4 Запуск на суперкомпьютере среды Sun Grid Engine в контейнерах

4.1 Организация расчетов на тестовом стенде

В отделении разработки вычислительных систем НИИСИ РАН исторически сложилась технология расчетов, основу которой составляет вычислительная инфраструктура под управлением Sun Grid Engine (SGE) [17, 18]. Под управлением SGE в отделении работает решающее поле из ВМ, на каждом из которых функционирует клиентский процесс SGE. Пользователи подключаются к специально выделенному узлу входа, на котором формируют свои расчетные задания и направляют их в очередь SGE. SGE распределяет задания из очереди по клиентским ВМ.

Производимые в отделении расчеты носят не постоянный, а периодический характер, но в

момент их проведения требуются значительные вычислительные мощности, которые способен предоставить только суперкомпьютерный центр. Как уже было отмечено, в МСЦ РАН в качестве основной СУЗ используется СУППЗ, которая полностью управляет ВМ суперкомпьютера. По этой причине невозможно совместное применение СУППЗ и SGE для управления одними и теми же ВМ суперкомпьютера, т.е. должно быть обеспечено разделение функций управления. Кроме этого, результаты расчетов могут носить конфиденциальный характер, и после окончания очередной порции расчетов в суперкомпьютерном центре коллективного пользования необходимо обеспечить полную очистку ВМ суперкомпьютера от любых данных, которые могли остаться после расчетов (временные файлы, файлы статистики, логи расчетов и т.п.).

Для исследования возможности проведения расчетов в центре коллективного пользования (ЦКП) вычислительными ресурсами МСЦ РАН на базе суперкомпьютера МВС-10П ОП (разделы Haswell и Broadwell) [19] был выделен тестовый стенд, использующий стек программного обеспечения для управления суперкомпьютером и запуска задач под управлением SGE. Перед производством расчетов ВМ тестового стенда изымались из-под управления СУППЗ, на них устанавливалась инфраструктура SGE и производились расчеты. По завершении расчетов ВМ тестового стенда возвращались в основное вычислительное поле суперкомпьютера МВС-10П ОП под управление СУППЗ. При возврате узлов на них производилась полная переустановка операционной системы для гарантии удаления любой информации, оставшейся от расчетов на тестовом стенде.

В состав стенда вошли вычислительные узлы разделов Haswell и Broadwell суперкомпьютера МВС-10П ОП, телекоммуникационные сети (сеть доступа и сеть администрирования), а также виртуальные машины со стеком управляющего программного обеспечения на платформе Proxmox [20]. Схема стенда представлена на рисунке 2.

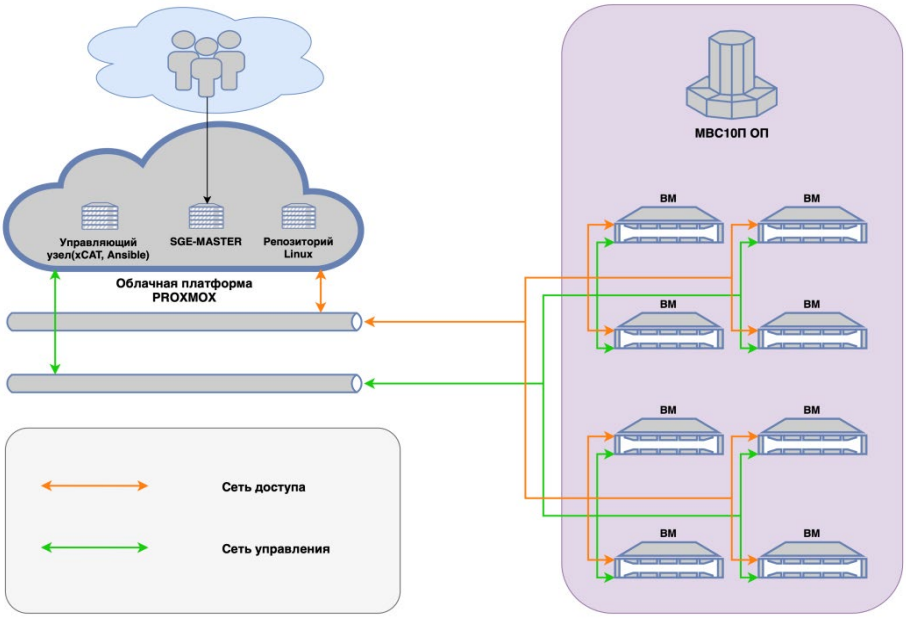


Рис. 2. Схема тестового стенда с инфраструктурой SGE

Для обеспечения работы тестового стенда было использовано следующее программное обеспечение:

- система управления распределенными вычислениями xCAT [21];
- система управления конфигурациями Ansible [22];
- web-сервер Nginx [23] с репозиториями Centos;
- NFS-сервер;
- система управления заданиями SGE.

Процесс запуска тестового раздела выглядел следующим образом. Во время еженедельного останова суперкомпьютера МВС-10П ОП на техническое обслуживание, после проверки работоспособности всех систем,

вычислительные модули в ручном режиме подключались администратором к сетям доступа и управления тестового стенда. Далее система xCAT в автоматическом режиме устанавливала на ВМ операционную систему. Последним шагом с помощью системы Ansible на ВМ устанавливались необходимые пакеты, библиотеки, рабочее окружение и пр. Клиентские процессы SGE на ВМ запускались автоматически после всех вышеперечисленных операций.

Работа пользователей тестового стенда организовывалась следующим образом: к виртуальной машине с мастер-сервером SGE был организован доступ по протоколу ssh. Пользователи подключались к ней удаленно со своих рабочих станций и далее работали с SGE в соответствии с принятым в НИИСИ РАН традиционным порядком. Вычислительные задания корректно запускались на ВМ суперкомпьютера МВС-10П ОП с использованием планировщика заданий SGE. Благодаря использованию системы xCAT и полной переустановке операционной системы при перемещении вычислительных модулей из основного вычислительного поля в тестовый стенд и обратно никаких остаточных данных от расчетов на тестовом стенде не оставалось.

Реализованный на тестовом стенде подход с разделением управления ВМ между СУППЗ и SGE во времени обладает рядом существенных недостатков. Основным является необходимость полного останова суперкомпьютера (или выделенного сегмента), что является недопустимым для штатного режима работы центра коллективного пользования. К недостаткам также следует отнести значительное время, затрачиваемое системным администратором на перевод ВМ из основного вычислительного поля в тестовый стенд и обратно.

4.2 Организация расчетов в системе контейнеризации

Для устранения недостатков было принято решение об использовании рассмотренной в разделе 3 настоящей статьи системы представления пользовательских заданий в виде контейнеров Docker. Схема решения представлена на рисунке 3.

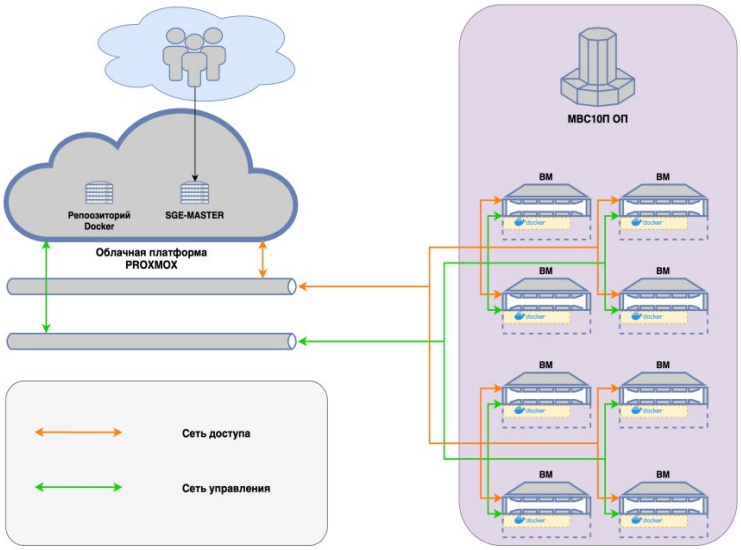


Рис. 3. Задание с инфраструктурой SGE в контейнерах

Был создан образ Docker, содержащий весь необходимый набор программ, библиотек, пользовательского окружения и клиентский сервер SGE. Среди способов создания контейнеров Docker был выбран вариант с использованием Dockerfile [24], т.к. он позволяет создавать образ контейнера путем описания выполняемых инструкций. Содержимое созданного образа сравнительно легко модифицируется путем изменений соответствующих инструкций в Dockerfile.

Собранный контейнер попадает в локальное хранилище образов на машине, где была запущена сборка. Далее его необходимо загрузить в рассмотренный в разделе 3 репозиторий Docker. После этого образ из репозитория может быть извлечен запущенным заданием, прошедшим через очередь СУППЗ. Успешный запуск задания и, соответственно, контейнеров на ВМ суперкомпьютера фактически означает развертывание вычислительной инфраструктуры SGE в

режиме коллективного доступа. Далее пользователи могут обычным порядком обращаться к мастер-серверу SGE и ставить свои расчетные задания в очередь SGE. Эти задания планировщиком SGE будут автоматически распределяться по контейнерам, запущенным под управлением СУППЗ на ВМ суперкомпьютера. Полная очистка ВМ от всех возможных остатков вычислений достигается простым удалением контейнера по завершении задания СУППЗ.

5 Заключение

В настоящей работе авторами решена задача представления суперкомпьютерных заданий в виде контейнеров Docker в системе коллективного пользования научного суперкомпьютерного центра. Необходимые для выполнения задания окружение и программная среда упаковываются в контейнер, который размещается администратором суперкомпьютерного центра в специально организованном репозитории. При запуске задания, прошедшего через очередь системы управления заданиями, образ контейнера автоматически извлекается из репозитория, после чего на выделенных для задания вычислительных модулях суперкомпьютера разворачивается отдельная сеть контейнеров. Внутри этих контейнеров начинают выполняться процессы задания. При завершении задания контейнеры останавливаются и удаляются с модулей суперкомпьютера.

Рассмотренный подход был реализован для Системы управления прохождением параллельных заданий (СУППЗ), используемой в МСЦ РАН. Реализованное решение было успешно применено для динамического развертывания на суперкомпьютере МВС-10П ОП вычислительной инфраструктуры под управлением платформы Sun Grid Engine (SGE), которая применяется в отделении разработки вычислительных систем НИИСИ РАН для высокопроизводительных расчетов. При этом был сохранен привычный пользователям исторически сложившийся порядок осуществления этих расчетов.

Публикация выполнена в рамках программы №2 Президиума РАН «Механизмы обеспечения отказоустойчивости современных высокопроизводительных и высоконадежных вычислений» по теме (проекту) «Исследование технологий виртуализации для создания цифровой платформы суперкомпьютерного проектирования и моделирования, ориентированной на субъекты цифровой экономики».

Job Containerization in a Supercomputer Job Management System

A.V. Baranov, B.V. Dolgov, A.V. Fedotov

Abstract: The article considers the representing supercomputer jobs in the container form. The job software environment packed as the Docker container image is placed in the supercomputer center repository. When the job starts, the image is extracted from the repository and deployed to the supercomputer nodes allocated to the job. Each node launches a separate container, which is combined into a single virtual network with the job's other containers. Job processes start in the virtual container environment deployed at the allocated nodes. The article discusses the practical implementation of job containerization for the domestic job management system called SUPPZ used at the Joint Supercomputer Center of the Russian Academy of Sciences (JSCC RAS). As part of the SUPPZ jobs, the virtual container environment included the Sun Grid Engine infrastructure was deployed on the MVS-10P OP supercomputer.

Keywords: HPC, supercomputer, job scheduling, workload manager, containerization, Docker, SGE

Литература

1. A. Reuther et al.: Scalable system scheduling for HPC and big data. “Journal of Parallel and Distributed Computing”, V. 111 (2018), 76–92. DOI: 10.1016/j.jpdc.2017.06.009

2. А.О. Кудрявцев, В.К. Кошелев, А.О. Избышев, А.И. Аветисян. Высокопроизводительные вычисления как облачный сервис: ключевые проблемы. «Международная научная конференция «Параллельные вычислительные технологии 2013 (ПаВТ'2013)», Россия, Челябинск, 31 марта – 5 апреля 2013, 432–438.

3. O.S. Aladyshev, A.V. Baranov, R.P. Ionin, E.A. Kiselev, B.M. Shabanov. Variants of deployment the high performance computing in clouds. “IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)”, Russia, Moscow, 2018, 1453–1457. DOI: 10.1109/EIConRus.2018.8317371

4. А.В. Баранов, Д.С. Николаев. Использование контейнерной виртуализации в организации высокопроизводительных вычислений. «Программные системы: теория и приложения», Т. 7 (2016), № 1(28), 117–134. DOI: 10.25209/2079-3316-2016-7-1-117-134

5. В.А. Щапов., С.Р. Латыпов. Способы запуска задач на суперкомпьютере в изолированных окружениях с применением технологии контейнерной виртуализации Docker. «Научно-технический вестник Поволжья», (2017), № 5, 172–177.

6. Д.С. Николаев, В.В. Корнеев. Использование механизмов контейнерной виртуализации в высокопроизводительных вычислительных комплексах с системой планирования заданий Slurm. «Программная инженерия», Т. 8 (2017), № 4, 157–160.

7. А.В. Баранов, А.С. Шитик. Способы и средства динамической реконфигурации сетей суперкомпьютера при представлении пользовательских заданий в виде контейнеров. «Программные продукты, системы и алгоритмы», (2018), № 3, 62–70. DOI: 10.15827/2311-6749.18.3.8

8. Г.И. Савин, П.Н. Телегин, А.В. Баранов, А.С. Шитик. Способы и средства представления пользовательских суперкомпьютерных заданий в виде контейнеров Docker. «Труды научно-исследовательского института системных исследований Российской академии наук», Т. 8 (2018), № 6, 84–93. DOI: 10.25682/NIISI.2018.6.0012

9. R.L. Henderson. Job scheduling under the Portable Batch System. “Lecture Notes in Computer Science”, V. 949 (1995), 279–294. DOI: 10.1007/3-540-60153-8_34

10. A.B. Yoo, M.A. Jette, M. Grondona. SLURM: Simple Linux Utility for Resource Management. “Lecture Notes in Computer Science”, V. 2862 (2003), 44–60. DOI: 10.1007/10968987_3

11. IBM Spectrum LSF overview. https://www.ibm.com/support/knowledgecenter/en/SSWRJV_10.1.0/lsf_foundations/chap_lsf_overview_foundations.html (дата обращения 04.10.2019)

12. Moab HPC Suite Enterprise Edition. http://www.adaptivecomputing.com/products/hpc-products/moab-hpc-suite-enterprise-edition (дата обращения 04.10.2019)

13. Система управления прохождением параллельных заданий. Руководство программиста (пользователя), (2016). http://www.jscc.ru/wp-content/uploads/2017/06/SUPPZ-user-guide-2016.pdf (дата обращения: 04.10.2019)

14. А.В. Баранов, А.В. Киселев, В.В. Старичков, Р.П. Ионин, Д.С. Ляховец. Сравнение систем пакетной обработки с точки зрения организации промышленного счета. «Научный сервис в сети Интернет: поиск новых решений: Труды Международной суперкомпьютерной конференции», Россия, Новороссийск, 17–22 сентября 2012, 506–508. http://agora.guru.ru/abrau2012/pdf/506.pdf (дата обращения: 10.10.2019)

15. А.В. Баранов, Д.С. Ляховец. Сравнение качества планирования заданий в системах пакетной обработки SLURM и СУППЗ. «Научный сервис в сети Интернет: все грани параллелизма: Труды Международной суперкомпьютерной конференции», Россия, Новороссийск, 23–28 сентября 2013, 410–414. http://agora.guru.ru/abrau2013/pdf/410.pdf (дата обращения: 04.10.2019)

16. Общие сведения о контейнерах и Docker. https://docs.microsoft.com/ru-ru/dotnet/standard/containerized-lifecycle-architecture/ (дата обращения 14.11.2019)

17. W. Gentzsch. Sun Grid Engine: towards creating a compute power grid. “First IEEE/ACM International Symposium on Cluster Computing and the Grid”, Australia, Brisbane, Queensland, 2001, 35–36. DOI: 10.1109/CCGRID.2001.923173

18. Y. Yang, Y. Chen. Sun Grid Engine (SGE) and its application. “International Symposium on Computers & Informatics”, 2015, 975–982. DOI: 10.2991/isci-15.2015.129

19. Вычислительные ресурсы МСЦ РАН. Суперкомпьютер МВС-10П ОП, (2019). http://www.jscc.ru/resources/hpc/#item1459 (дата обращения: 01.11.2019)

20. Е.А. Киселев, А.В. Баранов. Облачная среда для высокопроизводительных вычислений на базе платформы Proxmox. «Информационные технологии и высокопроизводительные вычисления: материалы V Международной научно-практической конференции», Россия, Хабаровск, 16–19 сентября 2019, 60–65.

21. Extreme Cloud Administration Toolkit. https://xcat-docs.readthedocs.io/en/latest/ (дата обращения: 01.11.2019)

22. M. Heap. Ansible: From Beginner to Pro. Apress, Berkeley, 2016. DOI: 10.1007/978-1-4842-1659-0
23. R. Soni. Nginx: From Beginner to Pro. Apress, Berkeley, 2016. DOI: 10.1007/978-1-4842-1656-9
24. J. Cook. The Dockerfile. “Docker for Data Science”, Apress, Berkeley, 2017. DOI: 10.1007/978-1-4842-3012-1_5

Средства автоматического сохранения контрольных точек в суперкомпьютерной системе коллективного пользования

А.В. Баранов[1], Р.С. Федоров[2]

[1]Межведомственный суперкомпьютерный центр РАН – филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия, antbar@mail.ru

[2]Московский физико-технический институт (государственный университет), Москва, Россия, aspcartman@gmail.com

Аннотация: Статья посвящена задаче автоматического сохранения контрольных точек для заданий, использующих для своего выполнения единственный вычислительный узел суперкомпьютера. В статье сформулированы требования к средствам автоматического сохранения контрольных точек в системе коллективного пользования суперкомпьютером. Рассмотрены средства Berkeley Lab Checkpoint/Restart (BLCR), Checkpoint Restore In Userspace (CRIU) и Distributed MultiThreaded Checkpointing (DMTCP). Показано, что сформулированным требованиям более всего удовлетворяет продукт DMTCP, для которого экспериментально получены оценки быстродействия и влияния на производительность вычислений. Рассмотрены вопросы интеграции с Системой управления прохождением параллельных заданий (СУППЗ), применяемой в МСЦ РАН, и рекомендации по практическому применению средств автоматического сохранения контрольных точек.

Ключевые слова: суперкомпьютер, контрольная точка, система управления заданиями, СУППЗ, DMTCP, CRIU, BLCR

1 Введение

Современные суперкомпьютерные системы, как правило, функционируют в режиме коллективного пользования. Суперкомпьютерные расчеты пользователи оформляют в виде вычислительных заданий, включающих расчетные программы и исходные данные, требования к характеру и объему запрашиваемых ресурсов. Специальные программные системы управления заданиями (СУЗ) [1] ведут очередь заданий, выделяют ресурсы и запускают на них пользовательские задания, контролируют их выполнение, высвобождают ресурсы после завершения заданий и предоставляют пользователю результаты расчетов.

Практика суперкомпьютерных центров коллективного пользования показывает, что время нахождения задания в очереди может составлять от нескольких часов до нескольких дней. В этих условиях администраторы суперкомпьютерных центров вынуждены ограничивать время, которое может быть выделено для выполнения отдельного задания. Например, в Межведомственном суперкомпьютерном центре РАН (МСЦ РАН) это время ограничено 24 часами.

Подобные ограничения отрицательно сказываются на качестве обслуживания тех пользователей, которым необходимы длительные расчеты. В применяемой в МСЦ РАН системе управления прохождением параллельных заданий (СУППЗ) [2] существует механизм т.н. фоновых заданий, которые могут выполняться произвольное время, но при этом могут прерываться системой с возвратом в очередь. Для организации длительных расчетов пользователь может применить механизм фоновых заданий, но задача сохранения состояния вычислений (контрольных точек) перед возвращением задания в очередь и восстановления состояния вычислений после возобновления выполнения задания целиком и полностью ложится на самого пользователя. Достаточно большое число пользователей МСЦ РАН успешно решают эту задачу – на суперкомпьютерах центра доля ресурсов, потребленных фоновыми заданиями, составляет от 15% до 30%. Однако, самостоятельная организация контрольных точек (КТ) возможна, как правило, только при использовании авторских кодов. Если же пользователь применяет в расчетах сторонние программные пакеты (а таких пользователей с каждым годом становится все больше), далеко не всегда присутствует возможность сохранения состояния вычислений.

Для параллельных программ, способных масштабироваться до большого числа процессорных ядер, возникает еще одна проблема – суммарный размер контрольных

точек всех параллельных процессов может быть настолько большим, что время сохранения и восстановления состояния вычислений может оказаться неприемлемо долгим. С другой стороны, наблюдается тенденция роста числа процессорных ядер как на отдельных кристаллах, так и на вычислительных узлах суперкомпьютеров. Например, вычислительный узел суперкомпьютера МВС-10П ОП [3] на базе процессора Intel Xeon Broadwell содержит 32 процессорных ядра, а узел на базе процессора Slylake – 36 ядер. Для достаточно большого числа параллельных программ это является пределом масштабируемости, подобные задания заказывают для своего выполнения один вычислительный узел, при этом могут проводить в очереди длительное время. Несмотря на то, что потребление ресурсов заданиями, требующими для выполнения единственный узел, не превышает 5%, их число достаточно велико и может достигать до 40% от общего числа заданий.

Изложенные факты свидетельствуют об актуальности исследования средств автоматического сохранения контрольных точек для вычислительных заданий, использующих для выполнения единственный узел суперкомпьютера. Проведение подобного исследования является целью настоящей работы.

2 Требования к средствам автоматического сохранения контрольных точек

Для возможности эффективного применения в системах коллективного пользования средств автоматического сохранения контрольных точек последние должны удовлетворять следующим требованиям.

1. Поддержка ядер операционных систем для суперкомпьютеров. Как правило, ядро ОС Linux в суперкомпьютерных системах является достаточно «старым», что обусловлено требованием надежности стека системного ПО, включающего поддержку высокоскоростных сетевых интерфейсов типа Infiniband и Omni-Path. Например, CentOS актуальной версии 7.3 использует ядро версии 3.10, в то время как последняя на сегодняшний день версия ядра Linux – 4.17. В то же время значительное число решений в области автоматического сохранения состояния вычислений рассчитаны на последние версии ядра Linux.

2. Средство автоматического сохранения контрольных точек не должно требовать модификации кода пользовательской программы. Система управления заданиями (СУЗ) научного суперкомпьютерного центра не накладывает ограничений на применяемые пользователями при подготовке параллельных программ алгоритмы, языки программирования и инструментальные средства. В результате на выполнение поступает широкий спектр различных типов параллельных программ, и фактически невозможно потребовать от разработчиков внесения изменений для поддержки автоматического сохранения контрольных точек.

3. Поддержка как многопоточных (разработанных при помощи OpenMP или pthread), так и многопроцессных (разработанных при помощи MPI) пользовательских программ. Средство автоматического сохранения КТ должно «понимать», что параллельная программа представляет собой довольно сложный динамически изменяющийся объект, состоящий из множества взаимодействующих друг с другом процессов и потоков.

4. Поддержка сохранения состояния объектов ядра ОС, таких как дескрипторы открытых файлов, области разделяемой памяти, очереди IPC, семафоры и т.п. Например, в случае работы пользовательской программы с файлом особая структура в области памяти ядра хранит информацию о местоположении открытого файла, режимах ввода-вывода, блокировках, текущем положении указателя чтения-записи и др. Механизмы межпроцессного взаимодействия, такие как очереди IPC, семафоры, области разделяемой памяти также имеют свои дескрипторы и состояние. TCP-соединения при снятии программы с выполнения будут разорваны, и после возобновления выполнения их надо будет восстановить. Если параллельная программа состоит из нескольких взаимодействующих процессов, необходимо будет восстановить идентификаторы всех процессов.

5. Минимальное влияние на производительность вычислений. Средство автоматического сохранения КТ не должно привносить в процесс вычислений существенных накладных расходов, снижающих быстродействие суперкомпьютерных приложений.

6. Возможность применения без получения привилегий суперпользователя. Как уже было отмечено, на характер исполняемых в научном суперкомпьютерном центре пользовательских программ не накладывается ограничений, никакой административной проверке перед выполнением эти программы не подвергаются. В такой ситуации наличие у средства автоматического сохранения КТ привилегий

суперпользователя является очевидной уязвимостью в системе безопасности суперкомпьютерного центра, которой теоретически и практически могут воспользоваться потенциальные злоумышленники. Применение средства, требующего привилегий суперпользователя, потребует, как минимум, дополнительного аудита безопасности и в любом случае увеличит площадь потенциальной атаки.

7. Лицензия должна позволять свободное применение в научном суперкомпьютерном центре коллективного пользования.

3 Решения для автоматического сохранения и восстановления состояния вычислений

3.1 Berkeley Lab Checkpoint/Restart (BLCR)

BLCR разработан командой Future Technologies Group в составе Lawrence Berkeley National Lab под финансированием SciDAC, США. BLCR добавляет в процесс функционал контрольных точек путем т.н. обертки системных вызовов. BLCR – это программное обеспечение с открытым исходным кодом, специально разработанное для высокопроизводительных вычислений, включающее поддержку MPI и OpenMP [4].

BLCR состоит из подсоединяемой к пользовательской программе динамической библиотеки (лицензия LGPL) и модуля (лицензия GPL), загружаемого в ядро ОС Linux, причем, что существенно, поддерживаются ядра сравнительно старых версий 2.6 и 2.7, при этом BLCR обеспечивает поддержку ядер до версии 3.7 [5]. Наличие отдельного загружаемого модуля делает BLCR неприменимым, поскольку требует модификации стека системного ПО.

3.2 Distributed MultiThreaded Checkpointing (DMTCP)

DMTCP был разработан в College of Computer and Information Science Northern University в 2006 году [6], в 2014 году продукт обрел поддержку MPI и Infiniband [7], а в 2016 году –возможности по восстановлению уникальных идентификаторов процессов [8]. DMTCP добавляет в пользовательское приложение функционал контрольных точек без необходимости модификации исходных текстов программы или модулей ядра. DMTCP обладает модульной архитектурой, позволяющей реализовать недостающий функционал и поддержку необходимых системных вызовов и устройств самостоятельно.

DMTCP производит все операции в пространстве пользователя через перехват системных вызовов в рамках пользовательского процесса. Это положительно влияет на безопасность и интеграционную простоту, но негативно сказывается на сложности самой реализации решения. DMTCP вводит в систему дополнительный процесс-координатор, задача которого состоит:

- в контроле над процессом сохранения и восстановления;
- поддержании согласованности состояния в случаях, когда это состояние не может быть получено из пространства пользовательского процесса;
- паравиртуализации системных вызовов на уровне пользовательского процесса.

Рассмотрим используемый в DMTCP механизм паравиртуализации на примере воспроизведения уникальных идентификаторов процессов. Для этого в пространстве процесса сохраняется соответствие между реальным идентификатором, известным ядру ОС, и виртуальным, который известен самому процессу и который будет воспроизводиться после каждый раз восстановления из контрольной точки.

Первым шагом отключается готовность к созданию контрольной точки процесса. Во время системного вызова нельзя останавливать выполнение, поскольку ядро находится в состоянии выполнения вызова, и это состояние невозможно сохранить. Вторым шагом виртуальный идентификатор заменяется на реальный, что делается также без выхода за пределы процесса: копия таблицы виртуальных идентификаторов хранится в каждом процессе и обновляется посредством обмена сообщения между ними. Затем выполняется настоящий системный вызов, возможность создания контрольных точек восстанавливается, и пользовательской программе возвращается результат. Поскольку системный вызов несет в себе одну из самых длительных операций – смену контекста, накладные расходы на паравиртуализацию являются пренебрежимо малыми [8].

3.3 Checkpoint Restore In Userspace (CRIU)

Продукт CRIU [9] разработан в рамках проекта Virtuozzo в 2011 году в России и распространяется по лицензии GNU GPL. CRIU полностью реализован в пространстве пользователя без загрузки динамической библиотеки при запуске процесса, что означает отсутствие необходимости в дополнительных модулях ядра, а также отсутствие перехвата системных вызовов, и обеспечивает

минимальное влияние на производительность вычислений. Особенностью CRIU является использование внешних по отношению к процессу ресурсов, таких как procfs, и получение всей необходимой информации по возможности не из самого процесса. Именно благодаря отсутствию необходимости подготовки процесса особым образом перед его запуском и влияния на его поведение CRIU со временем получил широкое распространение и был интегрирован в Docker – одну из самых известных систем контейнеризации [10].

Декларируется, что CRIU не требует привилегий суперпользователя для своего функционирования. Однако реализация CRIU на текущий момент не имеет явной поддержки MPI и InfiniBand. В этой области ведутся исследования и разработки, в частности, следует отметить работу над плагином, добавляющим поддержку MPI в CRIU [11]. Изначально отечественное решение по своим возможностям сильно опережало DMTCP из-за поддержки контейнеров и tcp-, udp-, unix-сокетов. Однако за последнее время DMTCP активно развивался, и большинство сравнительных статей того времени потеряли свою актуальность.

3.4 Сравнительный анализ

В рамках работы был произведен сравнительный теоретический анализ возможностей и технических ограничений существующих решений, результаты которого представлены в таблице 1. Как видно, BLCR имеет значительно худшие характеристики по сравнению с двумя другими решениями, которые очень близки между собой. Преимуществом CRIU является возможность работы с любым процессом системы без необходимости его подготовки, полное отсутствие влияния на производительность и, конечно же, сам факт отечественного производства. DMTCP, с другой стороны, выделяется наличием явной поддержки всех используемых технологий и ресурсов, подготовленностью к области высокопроизводительных вычислений. Для CRIU и DMTCP было проведено экспериментальное исследование их возможностей и характеристик.

Таблица. 1. Сравнительный анализ документированных возможностей средств автоматического сохранения контрольных точек

Документированная возможность	CRIU	DMTCP	BLCR
Не требует изменений в ядре	Да	Да	Нет
Не требует изменений в программе	Да	Да	Да
Не требует административных привелегий	Да	Да	Да
Не требует изменений в запуске задачи	Да	Нет	Нет
Не влияет на производительность	Да	Нет	Да
Поддержка скриптов	Да	Да	Да
Поддержка MPI	OpenMPI	Да	Да
Поддержка Unix-сокетов	Да	Да	Нет
Поддержка UDP-сокетов	Да	Нет	Нет
Поддержка TCP-сокетов	Да	Да	Нет
Восстановление TCP-соединения	Да	Да	Нет
Поддержка Infiniband	Нет	Да	Нет
Поддержка многопоточных процессов	Да	Да	Да
Поддержка многопроцессных приложений	Да	Да	Да
System V IPC	Да	Да	Нет
Отображения в память	Да	Да	Частично
Каналы (pipes)	Да	Да	Нет
Таймеры	Да	Нет	Да
Разделяемые ресурсы	Да	Да	Нет

4 Экспериментальная проверка работоспособности решений

4.1 Тестовое программное обеспечение

Для проверки работоспособности решений CRIU и DMTCP они были установлены на разделе Haswell суперкомпьютера МВС-10П ОП [3]. От имени и с правами обычного пользователя в его домашнем каталоге было создано локальное окружение, в которое была произведена установка всех необходимых компонентов CRIU и DMTCP, а также их зависимостей. Для проведения эксперимента в воспроизводимых условиях было разработано

тестовое программное обеспечение: две служебных программы и две нагрузочных. Рассмотрим эти программы.

1. Тест simple. Простейшая нагрузочная программа, написанная на языке на C, выводящая в стандартный поток вывода с заданным периодом времени последовательность натуральных чисел, начиная с нуля.

2. Нагрузочная программа simple_mpi написана на языке C и служит простейшей проверкой совместимости тестируемых решений с реализацией стандарта MPI. В ней множество процессов постоянно передают свои ранги главному процессу, который собирает эти ранги и помещает в стандартный вывод запись об успешном получении ранга вместе с номером итерации. Программа использует функцию MPI_Barrier для гарантированного останова («зависания») в случае, если какое-либо сообщение не будет доставлено.

3. Утилита crwrap предназначается для «оборачивания» вычислительного процесса вкупе с координированием работы DMTCP и CRIU, предоставляя единый интерфейс управления контрольными точками. Утилита запускает вычислительный процесс необходимым для выбранного решения образом (в случае DMTCP подгружается его библиотека и стартует процесс-координатор), и по сигналу SIGUSR2 запускает процедуру сохранения контрольной точки, делая соответствующий вызов к DMTCP или CRIU. Если при запуске утилиты в текущем каталоге была обнаружена сохраненная контрольная точка, то утилита вместо запуска нового вычисления восстанавливает его из контрольной точки и, при успешном запуске, удаляет контрольную точку. Утилита дожидается завершения вычисления или завершения процедуры создания контрольной точки, после чего завершается с соответствующим результату статусом. Важно учесть, что утилита выполняет создание контрольной точки и завершение процессов вычислений атомарно, что не является поведением по умолчанию для DMTCP: без указания специальных флагов координатора и процедуры создания контрольной точки DMTCP продолжает выполнение процесса после создания КТ. Если после создания контрольной точки процесс продолжит свое выполнение, то финальное состояние вычислений не будет соответствовать состоянию в сохраненной контрольной точке. Например, если процесс записывал в файл, то состояние процесса в сохраненной контрольной точке будет не соответствовать состоянию файла, и после восстановления процесс запишет в файл дубликат данных. Утилита реализована с помощью языка программирования golang. Встроенные в язык возможности по управлению дочерними процессами и его многопоточная основа позволяют достигнуть результата проще и быстрее, чем в случае с реализацией на других языках.

4. Утилита crtest, также реализованная на языке golang, является головной частью тестирования работоспособности решений. Утилита запускает вычисления через crwrap и создает контрольные точки и восстанавливает с них вычисления с указанным периодом. Каждые N секунд утилита отправляет сигнал SIGUSR2 процессу crwrap и дожидается его завершения, которое происходит по окончании создания контрольной точки. После его успешного завершения утилита вновь вызывает crwrap, который должен восстановить вычисления из ранее созданной контрольной точки.

4.2 Результаты тестирования

Тестирование проводилось на специально выделенном вычислительном узле раздела Haswell суперкомпьютера МВС-10П ОП [3], установленного в МСЦ РАН.

На тесте simple CRIU не выполнил ожидаемого сохранения контрольной точки, сообщив о невозможности сохранить состояние псевдотерминала, с которым был связан стандартный вывод процесса. Проблема состоит в поддержке псевдотерминала Linux, который используется при доступе к узлу через протокол ssh. CRIU заявляет о полной поддержке указанного типа терминала, однако в проведенном эксперименте ни одна из попыток не привела к положительному результату. Так как в режиме коллективного доступа к суперкомпьютеру любая СУЗ перенаправляет стандартные потоки ввода, вывода и ошибок в файлы, то выявленной особенностью CRIU было решено пренебречь и в дальнейших экспериментах перенаправить стандартные потоки в файлы, как это делают СУЗ.

Дальнейшие эксперименты выявили более существенную проблему в CRIU, которому потребовались привилегии суперпользователя для создания контрольной точки. После предоставления этих привилегий контрольная точка была успешно создана, восстановление из нее также прошло благополучно. Однако требование привилегий суперпользователя является неприемлемым по причинам, изложенным в разделе 2 настоящей статьи.

DMTCP успешно выполнил сохранение контрольной точки и восстановление из нее для теста simple. Корректное поведение

сохранялось на всем протяжении эксперимента, в ходе которого в общей сложности было осуществлено 2500 сохранений и восстановлений в течение часа. Существенных флуктуаций в показателях количества используемой оперативной памяти или процессорного времени выявлено не было.

DMTCP так же успешно выполнил сохранение контрольной точки и восстановление из нее процессов теста simple_mpi. После восстановления процессы продолжили свое выполнение корректно, верно отсчитывая итерации. Однако на третьей итерации восстановления из контрольной точки отсчет сбрасывался в нулевое состояние и начинался заново. Такое поведение является устойчивым и сохраняется при любых изменениях версий DMTCP и реализации Intel MPI.

Проведенный анализ показал, что, несмотря на то, что в вычислениях теста не используется сетевой интерфейс InfiniBand, он оказывается загруженным в процессы тестового приложения. Было сделано предположение, что ошибка восстановления процессов вычисления приводит к неявному перезапуску этих процессов. Поскольку в настоящей работе рассматривается случай проведения вычислений на одном суперкомпьютерном узле, допустимо отключить Infiniband в Intel MPI путем установки переменной окружения

I_MPI_FABRICS=shm

Такое значение указанной переменной заставляет процессы параллельной программы использовать для обмена разделяемую память, а не Infiniband. После такой настройки DMTCP начал выполнять тест simple_mpi без сбоев.

Для обеспечения работы на нескольких вычислительных узлах DMTCP может быть собран с интеграцией плагина поддержки Infiniband. Однако, на данный момент плагин InfiniBand вносит некоторые негативные изменения в процедуру создания контрольных точек: при использовании DMTCP с поддержкой InfiniBand происходит создание контрольной точки без учета заполненности страниц памяти данными. DMTCP имеет встроенную возможность определять страницы, заполненные нулями, и не включать такие страницы в дамп, в некоторых случаях существенно экономя место. Последствия данного изменения будут рассмотрены в следующем разделе статьи, посвященном анализа производительности средств создания контрольных точек.

5 Анализ производительности создания контрольных точек

5.1 Тестовые программы

Для анализа производительности DMTCP было разработано следующее тестовое программное обеспечение.

1. Тест cr_mem, реализованный на языке C. Программа выделяет три блока памяти указанного размера. Первый блок – блок «грязной» памяти, выделенный с помощью функции malloc [12]. Второй блок выделяется при помощи вызова calloc [13], во время которого заполняется нулями. Третий блок выделяется при помощи malloc, после чего заполняется случайными числами. После выделения указанных блоков программа в бесконечном цикле пишет постоянный поток данных в именованный канал в текущем каталоге для индикации своей активности.

2. Тест crbench, реализованный на языке golang, запускает cr_mem через crwrap несколько раз для разного размера блоков памяти, замеряя время. Далее ожидает готовности cr_mem с помощью чтения из именованного канала, создает контрольную точку, замеряя время и финальный объем контрольной точки, и удаляет контрольные точки при изменении объема блоков памяти.

Перед началом эксперимента предполагалось, что зависимость времени создания контрольной точки от объема данных близка к линейной пропорциональности, поэтому объем сохраняемых в КТ данных рассматривался как один из главных экспериментальных параметров. Кроме этого, DMTCP может использовать сжатие данных (gzip) при сохранении контрольной точки. Для исследования влияния сжатия на эффективность создания контрольных точек были разработаны упоминавшиеся тестовые программы crbench и cr_mem. В случае использования сжатия ожидалось полное поглощение неиспользуемых и заполненных нулями страниц. Объем контрольной точки в этом случае ожидался примерно равным объему блока памяти, заполняемого случайными числами, по причине фактической невозможности его сжатия. Скорость сохранения и восстановления контрольной точки ожидалась равной минимальной между скоростью сжатия/декомпрессии и скоростью доступа к системе хранения данных. Без использования сжатия объем контрольной точки ожидался равным сумме объемов всех трех выделенных блоков данных, а скорость создания контрольной точки должна была быть близка к скорости доступа к системе хранения данных.

5.2 Скорость сохранения контрольной точки и восстановления из нее

Результаты запусков теста crbench на узле суперкомпьютера МВС-10П ОП (раздел Haswell) представлены в таблице 2 (с включенным и отключенным сжатием). Отчетливо видна как линейная зависимость объема контрольной точки от заданного объема выделенных блоков, так и факт ожидаемого уменьшения объема контрольной точки в три раза при включенном сжатии.

Неожиданным результатом, в свою очередь, стало значительное ускорение создания КТ (в 6 раз) и восстановления из КТ (в 2 раза) при выключенном сжатии. Более того, такое улучшение имеет место быть с учетом увеличения объема контрольной точки в 3 раза. Такие показатели говорят о том, что отключение сжатия позволяет гораздо быстрее создавать контрольные точки, при этом порогом является скорость доступа к системе хранения данных.

Таблица 2. Результаты запусков теста crbench

Объем блока, ГБ	Итерация	Время сохранения КТ, с		Время восстановления, с		Объем КТ, МБ	
		Сжатие	Без сжатия	Сжатие	Без сжатия	Сжатие	Без сжатия
0	1	1,5	0,9			6,9	93,8
	2	1,2	1,7	2,8	4,1	7,1	94,4
	3	1,8	0,9	2,3	3,7	7,1	94,3
1	1	76,4	17,1			1040	3165,8
	2	89,6	18,9	22,4	11,1	1040	3166,4
	3	89,7	18,5	23,6	14,7	1040	3166,3
2	1	160,5	33,9			2073,1	6237,8
	2	149,5	34,5	40,6	20,3	2073,3	6238,4
	3	175,6	34,3	44,3	26,7	2073,3	6238,4
3	1	233,4	52,3			3106,2	9309,8
	2	250,3	50,6	66,8	30,7	3106,4	9310,4
	3	254,4	51	64,5	41,4	3106,4	9310,4
4	1	331,7	72,3			4139,3	12381,8
	2	352,4	68,3	81,7	40,6	4139,5	12382,4
	3	304,9	67,8	98,5	54,7	4139,6	12382,4

Для проверки данного предположения были произведены дополнительные запуски теста с использованием в качестве хранилища оперативной памяти (/dev/shm), результаты этих запусков можно наблюдать в таблице 3. Как видно, в случае использования оперативной памяти скорость создания и восстановления из контрольной точки превосходят скорость запуска тестового процесса.

Таблица 3. Результаты запусков теста crbench без сжатия с сохранением в оперативную память

Объем блока, ГБ	Время запуска теста, с	Итерация	Время сохранения КТ, с	Время восстановления, с	Объем КТ, МБ
0	1,3	1	0,9		93,8
		2	0,8	1,4	94,4
		3	1	1,8	94,4
1	7,4	1	2,5		3165,8
		2	2,5	3,6	3166,4
		3	3	3,5	3166,5
2	14,2	1	4,6		6237,8
		2	5,9	7,2	6238,4
		3	5,6	7,2	6238,4
3	20,3	1	5,5		9309,8
		2	7,8	9,8	9310,4
		3	6,8	8,4	9310,4
4	27,3	1	8,6		12381,8
		2	12,3	10,4	12382,4
		3	9,9	11,9	12382,4

5.3 Анализ влияния DMTCP на производительность вычислений

Теоретически DMTCP может оказывать влияние на производительность вычислений, поскольку использует механизм обертки системных вызовов и функций MPI. Для исследования влияния DMTCP на производительность был использован пакет Intel MPI Benchmarks [14] версии «IMB 2018 Update1». Пакет служит для анализа производительности реализации стандарта MPI от компании Intel. В ходе эксперимента были запущены группы тестов MPI1, EXT, IO, NBC, RMA.

Пакет Intel MPI Benchmarks содержит большое количество тестов, каждый из которых запускается с разными параметрами и на выходе предоставляет таблицу множества результатов. В подавляющем большинстве тестов существенных отличий в производительности, выходящих за рамки статистической погрешности, выявлено не было. Однако некоторые тесты, такие как EXT:Window, измеряющий производительность операций с «окнами» MPI [15], показали существенное ухудшение производительности до двух раз.

Отметим, что сами функции MPI не являются основной частью вычислений в большинстве случаев: они являются способом коммуникации между процессами, но сами не участвуют в вычислениях. Например, в работе [8] отмечено, что точечное падение производительности некоторых функций MPI влечет за собой падение общей производительности, составляющее менее 1%, или же неотличимое от статистической погрешности.

6 Вопросы интеграции с СУППЗ и практического применения средств автоматического создания контрольных точек

Как уже упоминалось, в МСЦ РАН в качестве основной системы управления заданиями применяется СУППЗ [2]. Полученные экспериментальные результаты позволяют сделать положительный вывод о целесообразности интеграции средств автоматического сохранения КТ в СУППЗ. Для запуска задания с поддержкой сохранения/восстановления КТ возможно применить рассмотренную выше в настоящей статье утилиту crwrap. Утилита должна выступать брокером между СУППЗ и процессами задания: корректно запускать задание с координатором DMTCP, принимать команду об останове от СУППЗ, производить создание контрольной точки и, после завершения процедуры создания контрольной точки, сигнализировать СУППЗ о завершении задания. При повторном запуске задания утилита должна обнаруживать ранее созданную контрольную точку и восстанавливать из нее состояние вычислений. Утилита была доработана для передачи флага использования сжатия и указания каталога хранения контрольных точек.

СУППЗ принимает на вход паспорт задания [2]. Для автоматического формирования паспорта и постановки в очередь задания, использующего MPI, пользователям предлагается применять стандартный сценарий СУППЗ mpirun. Этот сценарий формирует, в том числе, обращение к системе MPI во время запуска прошедшего очередь задания, например:

mpirun -np 8 the_task

По факту требуется изменить процедуру формирования паспорта таким образом, чтобы итоговая команда запуска MPI-программы приняла следующий вид:

crwrap mpirun -np 8 the_task

Сценарий СУППЗ mpirun требуется модифицировать таким образом, чтобы по получении флагов создания КТ добавлялись соответствующие изменения в команду запуска MPI-программы. Те пользователи, которые составляют паспорт задания вручную, могут самостоятельно добавить вызов crwrap в команду запуска задания.

При практическом применении DMTCP на текущий момент не рекомендуется использование сжатия в силу существенного замедления создания контрольной точки. При постановке заданий в СУЗ следует учитывать тот факт, что создание контрольной точки занимает некоторое, возможно, длительное время. Это время пользователю следует заранее рассчитать, разделив объем используемой процессами программы оперативной памяти на скорость доступа к хранилищу данных и прибавив некоторый запас. Например, результаты настоящей работы, относящиеся к разделу Haswell суперкомпьютера МВС-10П ОП, говорят, в случае отсутствия сжатия, о скорости в 200 МБ/с, при которой 1 ГБ данных будет сохраняться около 5 секунд. Кроме расчета времени сохранения контрольной точки рекомендуется проведение тестирования совместимости пользовательской прикладной

программы со средством автоматического сохранения КТ. В настоящей работе выявлен, как минимум, один случай несовместимости – при использовании в прикладной программе таймеров.

7 Заключение

Проведенное в работе исследование возможностей и характеристик средств автоматического создания контрольных точек показало целесообразность применения таких средств для заданий, использующих для выполнения единственный узел суперкомпьютера. Экспериментальное исследование средств автоматического сохранения КТ показало, что наиболее полно сформулированным в работе требованиям соответствует продукт Distributed MultiThreaded Checkpointing (DMTCP). DMTCP оказывает минимальное влияние на производительность вычислений, не требует изменений в исходных текстах программ и способен функционировать с правами обычного пользователя. Оценка производительности DMTCP показала нецелесообразность использования сжатия при сохранении контрольных точек. Для интеграции продукта с СУППЗ была разработана соответствующая утилита.

Если говорить о перспективах работы, следует отметить активное развитие проекта Checkpoint Restore In Userspace (CRIU), пока не выполняющего на практике требование неиспользования привилегий суперпользователя. Возможно, в ближайшем будущем CRIU обретет необходимый функционал.

В работе были получены негативные результаты использования сжатия контрольных точек. Спекулятивно предполагается зависимость скорости сжатия от разбиения участков памяти на блоки, и негативные результаты были получены именно из-за большого объема памяти, выделенной одним блоком. Логичным продолжением работы было бы исследование причин такого поведения DMTCP и возможных решений проблемы, а также статистическое исследование влияния сжатия на скорость создания и итоговый объем контрольных точек реальных вычислительных заданий.

Полезным может быть исследование возможности улучшения рассмотренных решений с помощью инкрементального создания контрольной точки. Для DMTCP имеется такая возможность с помощью HBICT (Hash Based Incremental Checkpointing Tool) [16]. Целесообразным видится исследование этого инструмента в условиях режима коллективного пользования научного суперкомпьютерного центра.

Исследование выполнено при финансовой поддержке РФФИ в рамках научных проектов № 18-07-01325 и № 19-07-01088.

Checkpoint and Restore Tools in a Supercomputer Job Management System

A.V. Baranov, R.S. Fedorov

Abstract: The article is devoted to the problem of automatically checkpoints for jobs that runs at a single supercomputer node. The paper formulates the requirements for the checkpoint and restore tools in the supercomputer job management system. Berkeley Lab Checkpoint / Restart (BLCR), Checkpoint Restore In Userspace (CRIU), and Distributed MultiThreaded Checkpointing (DMTCP) tools are reviewed. It is shown that the DMTCP product is the most satisfactory for the formulated requirements. For the DMTCP the estimates of speed and impact on computing performance have been experimentally obtained. The issues of integration with the job management system called SUPPZ used at the JSCC RAS, and recommendations on the checkpoint and restore tools practics are considered.

Keywords: HPC, supercomputer, checkpoint, job management system, workload manager, SUPPZ, DMTCP, CRIU, BLCR

Литература

25. A. Reuther et al.: Scalable system scheduling for HPC and big data. “Journal of Parallel and Distributed Computing”, V. 111 (2018), 76–92. DOI: 10.1016/j.jpdc.2017.06.009

26. Система управления прохождением параллельных заданий. Руководство программиста (пользователя), (2016). http://www.jscc.ru/wp-content/uploads/2017/06/SUPPZ-user-guide-2016.pdf (дата обращения: 04.10.2019).

27. Вычислительные ресурсы МСЦ РАН. Суперкомпьютер МВС-10П ОП, (2019). http://www.jscc.ru/resources/hpc/#item1459 (дата обращения: 01.11.2019).

28. P. Hargrove, J. Duell. Berkeley lab checkpoint/restart (BLCR) for Linux clusters. “Journal of Physics: Conference Series”, V. 46 (2006), 494–499. DOI: 10.1088/1742-6596/46/1/067

29. J. Cornwell, A. Kongmunvattana. Efficient System-Level Remote Checkpointing Technique for BLCR. “2011 Eighth International Conference on Information Technology: New Generations”, USA, Las Vegas, 2011, 1002–1007. DOI: 10.1109/ITNG.2011.172

30. M. Rieker, J. Ansel, G. Cooperman. Transparent user-level checkpointing for the Native POSIX Thread Library for Linux. “Parallel and Distributed Processing Techniques and Applications (PDPTA-06)”, 2006, 492–498.

31. J. Cao, G. Kerr, K. Arya, G. Cooperman. Transparent checkpoint-restart over Infiniband. “23rd international symposium on High-performance parallel and distributed computing (HPDC '14)”, USA, New York, 2014, 13–24. DOI: 10.1145/2600212.2600219

32. K. Arya, R. Garg, A. Polyakov, G. Cooperman. Design and Implementation for Checkpointing of Distributed Resources Using Process-Level Virtualization. “IEEE International Conference on Cluster Computing (CLUSTER)”, Taipei, 2016, 402–412. DOI: 10.1109/CLUSTER.2016.55

33. CRIU. https://criu.org/Main_Page (дата обращения: 04.11.2019).

34. Y. Chen. Checkpoint and Restore of Micro-service in Docker Containers. “3rd International Conference on Mechatronics and Industrial Informatics (ICMII 2015)”, 2015, 915–918. DOI: 10.2991/icmii-15.2015.160

35. A. Reber, P.Väterlein. Checkpoint/Restore in User-Space with Open MPI. “BW-CAR Symposium on Information and Communication Systems (SInCom 2014)”, Germany, Villingen-Schwenningen, 2014, 50–54. ISBN 978-3-00-048182-6, S. 55-592014.

36. Программирование на C и C++. Функция malloc. http://www.c-cpp.ru/content/malloc (дата обращения: 04.11.2019).

37. Программирование на C и C++. Функция calloc. http://www.c-cpp.ru/content/calloc (дата обращения: 04.11.2019).

38. Intel MPI Benchmark User Guide and Methodology Description, (2014). https://www.lrz.de/services/compute/courses/x_lecturenotes/mic_workshop/IMB_Users_Guide.pdf (дата обращения: 04.11.2019).

39. L. Nguyen, MPI One-Sided Communication, (2014). https://software.intel.com/en-us/blogs/2014/08/06/one-sided-communication (дата обращения: 04.11.2019).

40. About HBICT. http://hbict.sourceforge.net/about.html (дата обращения: 04.11.2019).

Инструмент для моделирования балансировки потока задач пользователей между несколькими независимыми вычислительными кластерами

М.Ю. Воробьев[1], А.А. Рыбаков[2], А.Н. Сальников[3]

[1,2]МСЦ РАН – филиал ФГУ ФНЦ НИИСИ РАН, Москва, Россия
[1,3]МГУ им. М.В. Ломоносова, Москва, Россия

e-mails: [1]mikhail.vorobyov@jscc.ru, [2]rybakov@jscc.ru, [3]salnikov@cs.msu.ru

Аннотация. Работа посвящена организации программного окружения для анализа алгоритмов балансировки нагрузки на ресурсы группы вычислительных кластеров, используемых для высокопроизводительных вычислений. Моделирование производилось путем одновременного запуска нескольких систем ведения очередей SLURM, работающих на нескольких виртуальных машинах в режиме мультиузла. Поток задач создавался посредством инструмента моделирования pseudo-cluster, модифицированного для поддержки одновременной работы с несколькими кластерами. Система апробирована на модельном журнале вычислительных задач пользователей, состоящем из 2000 задач, составленном по типичным паттернам поведения очередей реальных вычислительных кластеров.

Ключевые слова. Распределённые системы, вычислительный кластер, инфраструктура, система ведения очередей, балансировка нагрузки.

Введение

Для распределения ресурсов вычислительного кластера между задачами пользователей используются системы ведения очередей (планировщики), которые выделяют процессорное время вычислительного кластера задачам в зависимости от запрошенных вычислительных ресурсов, времени, приоритета и других параметров.

Специалисту или группе специалистов может быть доступен более чем один вычислительный кластер. В таком случае для выполнения потока задач пользователей разумно использовать все доступные кластеры. Эти кластеры имеют свои независимые системы ведения очередей, возможно построенные на различных программных продуктах. Также они могут управляться различными организациями. Поэтому мы не имеем полной информации о состоянии очередей и не можем влиять на распределение ресурсов [1]. Такая ситуация характерна при функционировании распределенной сети суперкомпьютерных центров коллективного пользования [2].

Для изучения поведения алгоритмов планирования в условиях группы вычислительных кластеров целесообразно применять моделирование для рассмотрения большого количества вариантов за разумное время [3,4]. Известны работы, направленные на решение данной проблемы. В частности в работе [5] описан инструментарий для моделирования систем из нескольких кластеров для изучения алгоритмов планирования. Он основан на системе моделирования процессов SimJava. Но сам SimIC публично не доступен. Также можно отметить следующие существующие программные средства для моделирования работы систем ведения очередей: slurm-sim [6], pseudo-cluster [7]. Имитационный пакет slurm-sim использует модифицированный SLURM [8], позволяющий регулировать скорость течения модельного времени. Программная система pseudo-cluster использует оригинальный SLURM, собранный с поддержкой запуска нескольких демонов вычислительного узла slurmd на одной машине. Для ускорения тестирования скрипт, моделирующий задачу, засыпает на время меньшее реального времени работы задачи в заданное число раз.

Постановка задачи

Пусть имеется множество вычислительных кластеров, на каждом из

которых установлена своя система ведения очередей. Каждый кластер имеет множество узлов и обслуживает свою очередь задач. Также присутствует общая очередь задач и метапланировщик (балансировщик). Метапланировщик решает, в очередь какого вычислительного кластера отправить появившуюся задачу пользователя. Назовём такое множество кластеров мультикластером. Каждый кластер имеет множество узлов и обслуживает свою очередь задач. Каждый узел имеет множество процессоров. Задача в рассматриваемом случае определяется требуемым числом процессоров, требуемым временем работы и реально использованным временем.

В рамках данной работы рассмотрим расширение функционала программной системы pseudo-cluster путем добавления поддержки работы с мультикластерами. Для модификации выбран pseudo-cluster, так как он не требует модифицированной версии SLURM, а позволяет использовать стандартный пакет из репозитория большинства дистрибутивов Linux.

Была выполнена модификация программы запуска потока задач и получения лога потока задач. Для обеспечения удобной работы с модельными кластерами также понадобятся скрипты для запуска SLURM с правильными настройками и для его остановки.

Целью работы является создание программной среды для имитационного моделирования прохождения потока задач через системы очередей нескольких вычислительных кластеров с добавлением свойства сжатия реального времени до модельного.

Описание решения и программная реализация

Как и в однокластерном режиме, будем запускать специально настроенный SLURM. Отличием будет то, что SLURM будет запускаться не на той машине, на которой работает головная часть pseudo-cluster. На одной машине будем запускать один SLURM, симулирующий один кластер. Головная часть pseudo-cluster будет общаться с ними по SSH.

Для тестирования разрабатываемого окружения были использованы следующие простые алгоритмы выбора кластера при распределении задач:

- «Round Robin» – данный алгоритм перебирает кластеры по очереди и каждому назначает следующую задачу из очереди.
- «Наиболее свободный» – в этом алгоритме регулярно собирается информация о числе свободных процессоров. При распределении текущая задача отправляется на кластер с наибольшим числом свободных процессоров.

Рассмотрим устройство pseudo-cluster, выбранного в качестве базового программного средства для расширения функционала. Он написан на скриптовом языке общего назначения Python. Программа работает с журналом в формате csv. Журнал содержит данные о запрошенном числе процессоров и лимите времени, времени отправки, времени начала и завершения работы. Точками входа для пользователя являются следующие скрипты:

- scripts/parse_llsummary_output.py, scripts/parse_slurm_db.py – используются для получения лога запуска задач в формате, понимаемом pseudo-cluster, от LoadLeveler с помощью llsummary и от SLURM напрямую из базы данных MySQL, соответственно;
- scripts/print_characteristics.py – для вычисления метрик по логу;
- scripts/pseudo_users_and_groups_operations.py – реализация работы с именами пользователей и групп в логах;
- scripts/run_pseudo_tasks_slurm.py – симуляция активности пользователей по данным из лога.

Опишем схему работы симулятора. Сначала он загружает лог в память в виде списка задач. Для того, чтобы симуляция проходила быстрее чем реальное выполнение, в качестве программы, запускаемой задачей, используется скрипт, засыпающий на заданное модельное (с учетом сжатия) время. Время, на которое засыпает скрипт, определяется как реальное время выполнения задачи, делённое на коэффициент сжатия comp, задаваемый пользователем опцией --time-compress. Далее раз в период времени T_{model}, заданный пользователем через опцию --time-interval в минутах, симулятор выбирает из списка задачи, отправленные, согласно логу, на выполнение за соответствующий период длительностью $T_{real} = T_{model} \times comp$ и отправляет их системе SLURM одного из виртуальных кластеров.

Для поддержки мультикластерного режима были добавлены следующие классы:

- SlurmCluster
- ClusterSelector
- RoundRobinClusterSelector
- MaxFreeCPUsClusterSelector

и доработаны следующие классы:

- Actions_List
- Scheduled_action
- Extended_task_record

SlurmCluster – предоставляет собой интерфейс взаимодействия с кластером. Команды передаются кластеру через SSH. Для этого была использована библиотека paramiko [9]. В конструктор передаются имя хоста (IP адрес) и, при необходимости, путь к файлу секретного SSH ключа. Для начала работы с кластером нужно вызвать метод connect(), который создаст экземпляр paramiko.SSHClient и создаст соединение с хостом. После этого можно выполнять на удалённой машине команды с помощью метода execute_command(command). Получить текущее состояние SLURM (число свободных, занятых и общее число узлов и процессоров) можно с помощью метода updateUsageInfo().

ClusterSelector – базовый класс для реализаций алгоритмов балансировки. В конструктор передаётся список кластеров SlurmCluster, для которых будет выполняться балансировка. Метод select_clusters_for_tasks(task_list) нужно переопределять в классах-наследниках, реализуя в нём алгоритм балансировки. Результатом работы метода является присвоение полю cluster_to_send элементов task_list выбраного для запуска кластера. Элементы task_list, которым не были назначены кластеры, остаются неотправленными и передаются для распределения снова при следующем запуске.

RoundRobinClusterSelector – реализация алгоритма «Round Robin». На каждой итерации цикла по task_list задаче назначается следующий по порядку кластер. Очерёдность кластеров обеспечивается хранением индекса следующего кластера для назначения в списке. Когда индекс сбрасывается в 0, когда доходит до конца списка.

MaxFreeCPUsClusterSelector – реализация алгоритма «Наиболее свободный». При вызове select_clusters_for_tasks(task_list) состояние всех SlurmCluster из clusters обновляется с помощью метода updateUsageInfo(). На каждой итерации цикла по task_list ищется кластер с максимальным числом свободных процессоров, который и назначается задаче. Кроме этого, в состоянии кластера изменяется число занятых и свободных процессоров на число процессоров, требуемых задачей. Последняя операция выполняется вместо получения реальной информации с кластера из-за больших накладных расходов по времени при использовании сетевого соединения SSH.

Actions_List и Scheduled_action – список действий для выполнения в ближайший период моделирования. Именно эти классы ставят задачи на выполнение и отменяют их. В оба класса добавлено поле логического типа multicluster, определяющее режим работы симулятора. Значение устанавливается одноимённым аргументом конструктора. В методы submit_task() и cancel_task() класса Scheduled_action добавлены альтернативные ветки выполнения для мультикластерного режима. В нём подготовленная команда вместо локального выполнения с помощью execvp() выполняется на соответствующем кластере с помощью метода execute_command().

Extended_task_record – класс, который содержит информацию о задаче. Добавлено поле cluster_to_send, для указания кластера, выбранного для запуска задачи.

Также были внесены изменения в скрипт симулирования потока задач run_pseudo_tasks_slurm.py. В основной цикл был добавлен вызов метода ClusterSelector.select_clusters_for_tasks(task_list) в мультикластерном режиме для распределения задач по кластерам.

Для нового режима добавлены опции командной строки:

- --multicluster – для включения мультикластерного режима;
- --cluster-selector – выбор алгоритма балансировки (RoundRobinn, MaxFreeCPUs);
- --hosts-file – путь к файлу со списком сетевых адресов машин, на которых запущены кластеры SLURM;
- --ssh-key – путь к файлу секретного ключа SSH.

На основе скрипта для получения лога запуска задач напрямую из базы данных parse_slurm_db.py был написан скрипт для получения лога с помощью утилиты sacct parse_slurm_sacct.py. Опции нового скрипта аналогичны опциям исходного за исключением добавления опции --host, с помощью которой указывается сетевой адрес машины, с которой нужно получить лог.

Для быстрой перенастройки SLURM в различные конфигурации были написаны скрипты на языке командной оболочки POSIX shell:

- start-slurm.sh – запуск SLURM с заданным числом узлов и процессоров на заданных машинах. В качестве опций принимает количество узлов (-n), количество процессоров на каждом узле (-c). Также скриптом используется шаблон конфигурационного файла, который должен находиться в текущем каталоге и называться slurm.template.conf. Значения опций -n и -c подставляются вместо $nn и $nc соответственно и получившийся файл копируется на машины.
- stop-slurm.sh – остановка SLURM на заданных машинах.
- clean-accounting-logs.sh – удаление логов запуска задач. Очищает состояние SLURM перед началом следующего теста.

Каждый из этих перечисленных скриптов имеет опцию -h, с помощью которой передаётся путь к файлу, содержащему список сетевых адресов машин, над которыми нужно совершить операцию.

Тестовые запуски

Для проведения тестирования разработанной программной среды на 8-и виртуальных машинах, управляемых системой libvirt [10] (см. рис. 1), была установлена система ведения очередей SLURM 17.11.2. Характеристики виртуальных машин:

- операционная система: Ubuntu 18.04;
- процессор: 1 ядро Intel Core i7-3770K CPU 3.50GHz;
- оперативная память: 1ГБ.

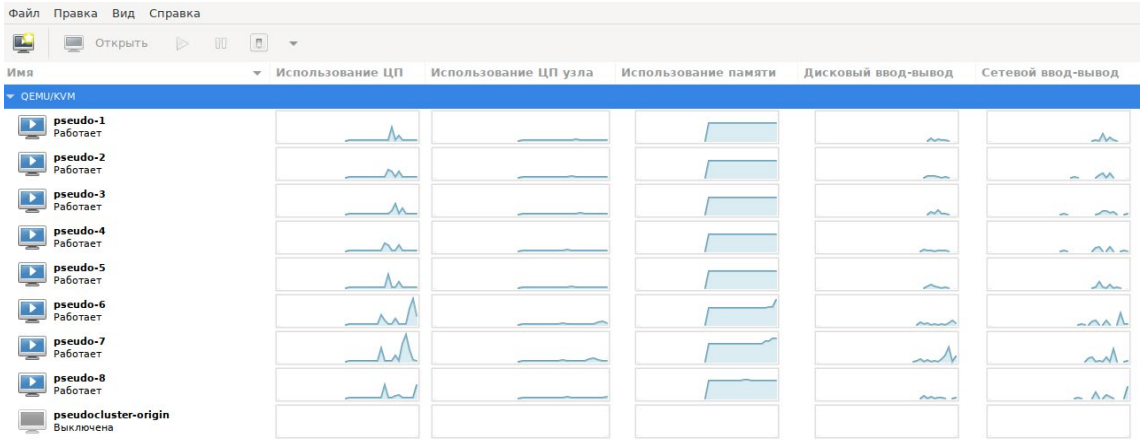


Рис. 1. Пример интерфейса приложения для управления виртуальными машинами virt-manager

В качестве тестовых данных использовался модельный журнал вычислительных задач пользователей, содержащий 2000 задач.

Для тестирования были использованы модели кластеров, состоящих из 256 узлов по 8 процессоров. Характеристики модельных задач выбраны таким образом, чтобы все задачи могли поместиться на модельном кластере. Число моделируемых узлов было ограничено доступными ресурсами виртуальных машин. При большем числе запущенных slurmd SLURM завершался с ошибками, не отработав положенное время. Что означает, что физические ресурсы были использованы полностью.

Было выполнено по 3 запуска с каждым из 2 алгоритмов («Round Robin» и «Наиболее свободный») на 1, 2, 4 и 8 кластерах.

Для осуществления всех намеченных запусков в автоматическом режиме был написан скрипт на POSIX shell, который использовал ранее написанные start-slurm.sh, stop-slurm.sh и clean-accounting-logs.sh. Пример запуска скрипта моделирования потока задач:

```
scripts/run_pseudo_tasks_slurm.py
    --time-compress 1024
    --time-interval 1
    --multicluster
    --path-to-task-script
        ./pseudo_cluster_task.sh
    --hosts-file ./hosts.0-15
    --cluster-selector $sel
> ./exp/${sel}\_16/$i/run.log
```

Объяснение опций:

- --time-compress 1024 – модельное время ускоряется по сравнению с реалльным в 1024 раза;
- --time-interval 1 – модельный период равен одной минуте реального времени;
- --multicluster – мультикластерный режим включен;
- --path-to-task-script ./pseudo_cluster_task.sh – путь к скрипту, моделирующему задание путём выполнения sleep с заданным временем;
- --hosts-file ./hosts.0-15 – путь к файлу со списком сетевых адресов машин, отдаваемых под управление pseudo-cluster;
- --cluster-selector $sel – выбор алгоритма балансировки.

Также данный скрипт после каждого теста сохраняет лог с помощью parse_slurm_sacct.py.

Используя скрипт print_characteristics.py, получим метрику среднего времени ожидания задачи для каждого запуска отдельно. При этом кроме исходного модельного потока задач будем использовать еще один поток, в котором нагрузка увеличена в 10 раз (вместо каждой поступающей на выполнении задачи на распределение отправляется 10 ее копий). Объединим полученные данные при помощи shell скрипта в таблицу с колонками: алгоритм, число кластеров, номер запуска, имя кластера, среднее время ожидания задачи. Для дальнейшей обработки данных воспользуемся средой Jupyter Notebook, библиотекой анализа данный pandas и библиотекой построения графиков Matplotlib. Усредним метрики, сгруппировав строки по алгоритму и числу кластеров. По этим данным построим графики среднего времени ожидания задачи в очереди для двух рассмотренных алгоритмов. Графики представлены на рис. 2 и рис. 3.

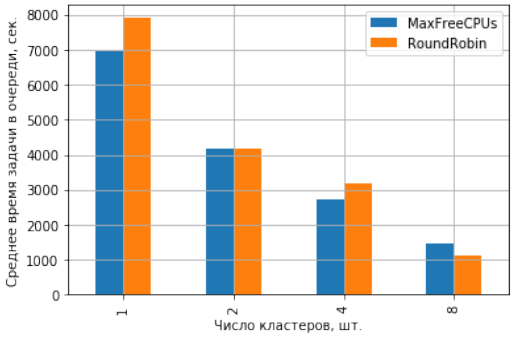


Рис. 2. Зависимость среднего времени ожидания задачи в очереди от числа кластеров для разных алгоритмов балансировки

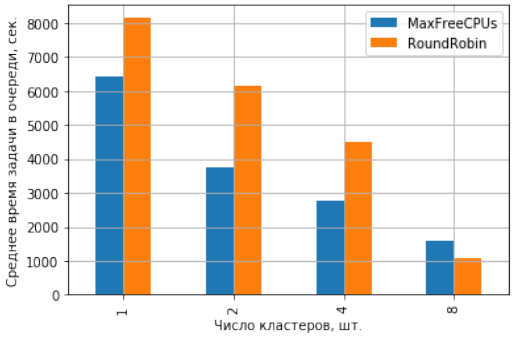


Рис. 3. Зависимость среднего времени ожидания задачи в очереди от числа кластеров для разных алгоритмов балансировки при использовании увеличенной в 10 раз нагрузки

Заключение

В систему моделирования кластеров pseudo-cluster был добавлен режим симулирования мультикластера и средства для автоматического развёртывания инфраструктуры мультикластера, которые позволяют изучать поведение различных алгоритмов балансировки нагрузки в мультикластере. Были добавлены два простые алгоритма балансировки нагрузки, которые могут быть использованы в качестве шаблонов для добавления новых алгоритмов с более сложной логикой. Таким образом разработана система, позволяющая произвести быстрое тестирование разрабатываемых алгоритмов распределения задач пользователей для мультикластера, отдельные кластеры которого обладают произвольными характеристиками. При этом тестирование алгоритмов распределения задач может быть осуществлено как для случайно генерируемого потока задач, так и для реальных логов работы вычислительных систем.

Работа выполнена в МСЦ РАН в рамках государственного задания по теме 0065-2019-0016. При проведении исследований использовался суперкомпьютер МВС-10П, находящийся в МСЦ РАН.

Development of algorithm for user tasks flow balancing over a number of computational clusters

M.Yu. Vorobyov, A.A. Rybakov, A.N. Salnikov

Abstract. The work is devoted to the organization of a software environment for the analysis of load balancing algorithms for a group of computing clusters for high-performance computing. To simulate the cluster, the SLURM queuing system was used, running on several virtual machines in multi-node mode. The task flow was created using the pseudo-cluster modeling tool, modified for supporting multiple clusters. The system was tested using model task flow journal of 2000 tasks based on typical task queue behavioural patterns of real computing clusters.

Keywords. Distributed systems, computational cluster, infrastructure, queue management system, load balancing.

Литература

1. A. Kertesz, Z. Farkas, P. Kacsuk, T. Kiss. Grid interoperability by multiple broker utilization and meta-broking. // Grid Enabled Remote Instrumentation, Springer, 2009, P. 303–312.
2. Б.М. Шабанов, А.П. Овсянников, А.В. Баранов и др. Проект распределенной сети суперкомпьютерных центров коллективного пользования. // Программные системы: Теория и приложения, 2017, 8:4(35), С. 245–262.
3. M. Rezaei, A. Salnikov. Machine learning techniques to perform predictive analytics of task queues guided by slurm. // 2018 Global Smart Industry Conference (GloSIC), IEEE, 2018, P. 1–6.
4. Tanash M. et al. Improving HPC system performance by predicting job resources via supervised machine learning. // Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning), ACM, 2019.
5. S. Sotiriadis, N. Bessis, N. Antonopoulos, A. Anjum. SimIC: Designing a new inter-cloud simulation platform for integrating large-scale resource management. // Proceedings of the 2013 IEEE 27th International Conference on Advanced Information Networking and Applications, 2013, P. 90–97.
6. Репозиторий slurm-sim. https://github.com/marinazapater/slurm-sim // Дата обращения 12.11.2019.
7. А.Н. Сальников, А.Н. Бойко. Программная система для моделирования активности пользователей вычислительного кластера на основе системы ведения очередей SLURM. // Параллельные вычислительные технологии (ПаВТ'2015): труды международной научной конференции, Издательский центр ЮУрГУ Челябинск, 2015, С. 463–470.
8. Slurm Workload Manager – Documentation. https://slurm.schedmd.com // Дата обращения 12.11.2019.
9. Welcome to Paramiko! – Paramiko documentation. http://www.paramiko.org // Дата обращения 12.11.2019.
10. libvirt: The virtualization API. https://libvirt.org // Дата обращения 12.11.2019.

О различных подходах к проверке решения графических задач

Н.О. Бесшапошников[1], М.С. Дьяченко[1], М.А. Кузьменко[1,2], А.Г. Леонов[1,2,3], М.А. Матюшин[1,2], К.А. Прокин[1]

[1] ФГУ ФНЦ НИИСИ РАН, Москва, Россия
[2] МГУ им. М.В.Ломоносова, Москва, Россия
[3] МПГУ, Москва Россия

E-mail's: nbesshaposhnikov@vip.niisi.ru, mdyachenko@niisi.ru, gmk.@vip.niisi.ru, dr.l@vip.niisi.ru, itsaprank@yandex.ru, prokin@vip.niisi.ru

Аннотация. В настоящее время образование всех уровней в России проходит трудный процесс цифровой трансформации. Это включает не только перевод образовательных материалов в цифровую форму и цифровой учет и контроль посещаемости студентами и школьниками образовательных организаций, но и перестройку всего образовательного процесса, например, дополнением цифровых методов контроля достигнутых компетенций студентами и школьниками, что включает в себя автоматизацию проверки контрольных работ, сданных решений заданий и т.п. Для детерминированных результатов задач (выбор из нескольких ответов или числовой ответ) проблема автоматической проверки не составляет труда, однако проверка работ, содержащих изображения в качестве решения, и в сейчас отнимает время преподавателей и ассистентов. В данной статье описано исследование нескольких вариантов автоматизации проверки решений задач, в которых присутствуют графические элементы. Описанные здесь алгоритмы позволяют частично или полностью автоматизировать процесс проверки решений в образовательных проектах, например, обучающих студентов основам компьютерной графики.

Ключевые слова: автоматизация проверки, нейронные сети, сравнение алгоритмов, проверка, автоматизация, компьютерная графика, образование, тесты.

1. Введение

Допустим, что имеется N задач графического содержания, ответом к каждой из которых является некоторое изображение. Пусть для каждой из задач также есть n_i^+ положительных примеров, прошедших проверку, и n_i^- отрицательных примеров, не прошедших проверок, $i \in \{1, \dots N\}$. Данные примеры могут быть взяты из ранее сданных студентами решений, проверенных ручным способом, либо подготовлены заранее.

Задача проверки изображения нового решения технически является задачей классификации на 2 класса: «правильно» и «неправильно». Данная задача появляется, например, в рамках автоматической проверки чертежей в обучающих системах и была всесторонне освещена в рамках работы [1]. Несмотря на свою простоту, предложенный в рамках указанной работы метод проверки наложением показывает высокую эффективность и точность работы при работе с чертежами. Тем не менее, в широком спектре задач, решения к которым представлены в графическом виде, а особенно в задачах с широкой постановкой, допускающих вариативность решения, или в случаях, когда невозможна сложная поэтапная разметка входящих данных, данный подход очевидно будет работать плохо. С целью расширения возможностей автоматизации проверки этого рода задач в данной статье предложены альтернативные возможные подходы к решению описанной проблемы.

Допустим для простоты, что все изображения имеют фиксированный размер 64 × 64 × 3, и цветовую палитру RGB. Задача данной статьи состоит в отыскании оптимального алгоритма автоматизации, при котором все или частично все сдаваемые впоследствии студентами решения можно доверять проверять некоторому алгоритму.

Далее описываются некоторые возможные решения данной проблемы. Описание тестового набора данных, на котором получено приведенное качество, приведено в приложении А. Метрики качества, выбранные для оценки, имеют следующее описание:

• **Error** – минимальная односторонняя ошибка, $min(FP, FN)$, где FP FP - доля

решений, ошибочно оцененных как верные, *FN* - доля решений, ошибочно оцененных как неверные. Мотивировка данной метрики состоит в том, что, возможно, алгоритму в текущей реализации нельзя доверить полностью проверку решений, однако можно доверить, например, отделять правильные решения (если $FP \approx 0$), а предсказанные как неверные отправлять на ручную проверку, либо наоборот, отделять неправильные решения (при $FN \approx 0$), а предсказанные как верные отправлять на ручную проверку.

• **F1** – метрика f1 score [2] при рассмотрении задачи проверки решения как задачи классификации на два класса - верное и неверное решение.

2. Проверка кластеризацией

Первым из рассмотренных в ходе исследования алгоритмов был метрический алгоритм, рассматривающий изображение как вектор в многомерном пространстве. Подробное обсуждение возможностей данного семейства алгоритмов приведено в [3]. Пусть входящее изображение имеет размер 64 × 64 × 3. Гиперпараметрами алгоритма являются ширина W, высота H и алгоритм кластеризации *A,* применяемый для отделения корректных решений от некорректных. Пошаговый алгоритм приведен ниже.

Алгоритм проверки кластеризацией:

1. Получить статистические данные для каждой из N задач, включающие решения, оцененные как верные, и решения, оцененные как неверные.
2. Каждое полученное изображение привести к размеру $H \times W \times 3$.
3. Для каждой задачи рассмотреть объединение множеств корректных и некорректных решений, полученных в результате ручной проверки, как одно множество векторов пространства размерности $3 \cdot H \cdot W$. В рамках каждой задачи i применить алгоритм A к $n_i^+ + n_i^-$ векторам с числом кластеров равным 2. Обученные алгоритмы являются внутренними обучаемыми параметрами алгоритма проверки.
4. При поступлении нового решения, требующего автоматической проверки, привести его к размеру $H \times W \times 3$, составить из него вектор, определить его принадлежность к одному из полученных на шаге 3 кластеров. В случае, если в кластере больше корректных решений, относить решение к правильным, в противном случае считать решение неверным.

В рамках тестирования был выбран алгоритм KMeans [4], результаты работы на тестовом наборе данных следующие:

Таблица 1. Результаты тестов для проверки кластеризацией

Идентификатор задачи	**Количество корректных решений**	**Количество некорректных решений**	**Error**	**F1**
1	36	5	0	0.909091
2	16	1	0	1.000000
3	40	10	0	0.666667
4	26	18	0.022727	0.566667
5	35	5	0.075000	0.352941
6	10	1	0.090909	0.250000
7	33	5	0.131579	0.357143
8	27	9	0.166667	0.333333
9	27	18	0.311111	0.734694

Недостатком данного алгоритма является низкая устойчивость в случае проверки задач, допускающих вариативность решений, например, когда от студента требуется изобразить куб, но не уточняется, где он должен находиться и какой размер иметь. Также недостатками является низкая различающая способность в случае

использования маленьких H и W , и большое время работы в случае использования больших H и W , тенденция к переобучению в последнем случае. Также, как нетрудно понять, качество работы данного алгоритма сильно зависит от применяемого алгоритма кластеризации A и от реального распределения правильных решений, допускаемых условиями задачи.

Из достоинств стоит отметить простоту реализации на любом языке программирования, сравнительно простое дообучение алгоритма на новых данных.

3. Проверка алгоритмом детекции особенностей

В данном разделе мы исследовали возможность автоматизации проверки решений графических задач с использованием алгоритмов детекции ключевых точек и особенностей изображения семейства *SIFT (Scale Invariant Feature Transform)*, *SURF (Speeded Up Robust Features)*. Описания данных алгоритмов можно найти в [5], [6].

Предлагаемый алгоритм автоматической проверки основывается на большом количестве совпадений ключевых точек и особенностей изображений у правильных решений задачи. Так в примере с кубом положение и размер куба в силу особенностей работы применяемых алгоритмов детекции ключевых точек могут быть проигнорированы, тогда как сам факт того, что на двух изображениях представлен один и тот же куб, будет учтен алгоритмом по совпадению найденных ключевых точек. Гиперпараметром алгоритма проверки является алгоритм детекции ключевых точек A, а также пороговое значение γ совпавших ключевых точек. Описание работы алгоритма приведено ниже.

Алгоритм проверки с использованием детекции ключевых точек:

1. Получить статистические данные для каждой из N задач, включающие решения, оцененные как верные, и решения, оцененные как неверные.
2. К каждому изображению корректного решения применить алгоритм детекции A. Полученные ключевые точки и их описания являются внутренними обучаемыми параметрами алгоритма проверки.
3. При поступлении нового решения, требующего автоматической проверки, применить к нему алгоритм A, затем сравнить количество совпавших ключевых точек последовательно с каждым набором ключевых точек, полученных на шаге 2. Если получено количество совпадений выше порогового значения γ, считаем решение корректным. Иначе считаем решение неверным.

В рамках нашего исследования в качестве алгоритма A был выбран алгоритм *SURF* [6], реализованный в библиотеке компьютерного зрения OpenCV, полученные результаты качества на тестовом наборе данных следующие:

Таблица 2. Результаты тестов для проверки детекцией особенностей

Идентификатор задачи	**Количество корректных решений**	**Количество некорректных решений**	**Error**	**F1**
2	16	1	0	1.000000
9	27	18	0	0.981132
6	10	1	0	0.952381
1	36	5	0	0.921053
3	40	10	0	0.888889
4	26	18	0	0.724638
5	35	5	0.025000	0.971429
7	33	5	0.026316	0.825397
8	27	9	0.166667	0.777778

Недостаток данного алгоритма является одновременно его достоинством по сравнению с алгоритмом проверки кластеризацией. Как было сказано выше,

данный алгоритм будет иметь тенденцию игнорировать точное пространственное положение объектов, соответственно, в случае строгой формулировки задачи показатель FP будет иметь неприемлемо высокое значение.

Достоинства данного алгоритма проявляются в его более высокой устойчивости при проверке решений вольно сформулированных задач. Также плюсом является простота добавления новых данных и, в отличие от проверки кластеризацией, независимость от статистически полученных некорректных примеров.

4. Проверка с помощью нейронных сетей

Последний проверенный в рамках данного исследования алгоритм использует в качестве ключевого фактора расстояние между изображениями, предсказываемое нейронной сетью. Для этих целей обучается специальная нейронная сеть, преобразующая входящее изображение в вектор в пространстве низкой размерности, при чем таким образом, чтобы расстояние между полученными векторами отражало отношение между изображениями, заданное условиями задачи. Способ обучения сети классифицирует её как сиамскую нейронную сеть [7]. Обученная сиамская нейронная сеть A является гиперпараметром алгоритма проверки, также как и некоторое пороговое значение уверенности $\alpha \in [0, 1]$. Пошаговое описание алгоритма проверки приведено ниже.

Алгоритм проверки с использованием детекции ключевых точек:

1. Получить статистические данные для каждой из N задач.
2. С помощью A получить вектора низкой размерности e_i^k, $k \in \{1, \dots, n_i^+\}$, $i \in \{1, \dots, N\}$ для каждого корректного решения для всех N задач. Для каждой из N задач получить центроид кластера корректных решений c^i и диаметр кластера $d^i, i \in \{1, \dots, N\}$. Эти данные являются внутренними параметрами алгоритма проверки.
3. При поступлении нового решения, требующего проверки, с помощью A получить вектор низкой размерности v, затем найти $d = min\{\|v - ci\|\}$, $D = max\{\|v - c^i\|\}$, $I = argmin\{\|v - c^i\|\}$, $i \in \{1, \dots, N\}$, и в случае, если $\frac{d}{d+D} < \alpha, d < c^I$, считать решение верным, если I - идентификатор проверяемой задачи. В противном случае считать решение некорректным.

В рамках данного исследования использовалась сиамская сеть следующей архитектуры:

Рис. 1. Архитектура сиамской сети

Сверточная часть нейронной сети, извлекающая признаки, была построена на основе InceptionV2 блоков ([8]) и предобучалась отдельно методом последовательного построения антишумовых автокодировщиков [9], в качестве обучающего набора данных была использована часть ImageNet 2011 размером в 150 000 изображений.

Таблица 3. Результаты тестов для проверки с помощью нейронных сетей

Идентификатор задачи	**Количество корректных решений**	**Количество некорректных решений**	**Error**	**F1**
1	36	5	0	0.98
4	26	18	0	0.98
5	35	5	0	0.98
2	16	1	0	0.96
3	40	10	0	0.85
6	10	1	0	0.82
8	27	9	0.02	0.94
9	27	18	0.02	0.81
7	33	5	0.05	0.93

Полносвязная часть нейронной сети, строящая низкоразмерное представление, затем дообучалась с использованием функции потерь *lossless triplet loss* [10] на тестовом наборе данных при замороженных весах сверточной части. Метрикой прогресса обучения нейронной сети служила следующая величина:

$$Q = exp\left(-\frac{1}{m}\sum_{j=1}^{m}\frac{diam_j}{dist_j}\right) \quad (1)$$

Здесь

$$m = \sum_{i=1}^{N} n_i^+$$

$$diam_j = \max\left\{\left\|e_j^{i_1} - e_j^{i_2}\right\|\right\},$$
$$i_1, i_2 \in \left\{1, \ldots, n_j^+\right\}$$

$$dist_j = \min\left\{\left\|e_j^{i_1} - e_k^{i_2}\right\|\right\},$$
$$i_1 \in \left\{1, \ldots, n_j^+\right\},$$
$$i_2 \in \{1, \ldots, n_k^+\},$$
$$k \in \{1, \ldots, j-1, j+1, \ldots, N\}\}$$

Результаты точности данного подхода на тестовом наборе данных следующие: К недостаткам данного алгоритма можно отнести сильную зависимость от архитектуры и качества обучения сиамской сети, а также высокую трудоемкость обучения последней, заключающуюся в первую очередь в предобучении сверточного блока сети, отвечающего за извлечение признаков из изображения.

Безусловным достоинством алгоритма является высокая генерализационная способность, возможность использования одной хорошо обученной модели во всех задачах.

5. Заключение

В рамках данной работы были исследованы несколько алгоритмов проверки решений графических задач. В таблице ниже приведены сравнительные результаты точности по всем задачам из тестовой выборки.

Из описанного можно сделать вывод о том, что несмотря на очевидные преимущества нейросетевого алгоритма проверки, важно не упускать из виду особенности формулировки задачи, в рамках которой проверяются решения. Очевидно, что для задач формулировкой, не допускающей никакой вариативности, будет прекрасно себя показывать метод наложения, описанный в [1], либо же метод проверки кластеризацией.

Таблица 4. Сравнительная таблица результатов для разных подходов

Идентификатор задачи	**Количество корректных решений**	**Количество некорректных решений**	**Error best**	**F1 best**
1	36	5	–	SIAMESE
2	16	1	–	SURF CLUSTERING
3	40	10	–	SURF
4	26	18	SURF SIAMESE	SIAMESE
5	35	5	SIAMESE	SIAMESE
6	10	1	SURF SIAMESE	SURF
7	33	5	SURF	SIAMESE
8	27	9	SIAMESE	SIAMESE
9	27	18	SURF	SURF

Лишь в случае, когда формулировка задачи предполагает вариативность в решениях, не ограничивающуюся разным цветом пикселей, действительно имеет смысл использовать обучающиеся подходы, предложенные в данной работе.

Работа выполнена по теме Гранта РФФИ 18-07-00901 «Исследование и разработка системы распознавания элементов рукотворного интерьера на базе нейронных сетей для построения дополненной реальности и выработки алгоритмов взаимодействия управляемых объектов с реально-виртуальным окружением».

On various approaches to automated grading of graphic assignments

N.O. Beshaposhnikov, M.S. Dyachenko, M.A. Kuzmenko, A.G. Leonov, M.A. Matushin, K.A. Prokin

Abstract. In various areas of educational activity, it is often a need to grade student's assignments automatically. While in case of checking traffic regulations, there are existing solutions to automate the process, then, automatic grading can be a severe difficulty for some courses, and this difficulty slows down the educational process and requires significant resources of teachers and assistants. This article describes the study of several options for automating the grading of assessments containing graphic elements; it means that the answer to the question is an image. The algorithms described here make it possible to automate the process of grading assessments in educational projects, for example, teaching students the basics of computer graphics.

Keywords: grading automation, neural networks, comparison of algorithms, grading, automation, computer graphics, education, tests.

Литература

1. А.А. Бойков. Конструктивно-геометрические основы и методика машинной проверки чертежей в обучающих системах. «Вестник Костромского государственного университета», Т. 22 (2016), № 1, 163–167.

2. Sasaki Yutaka. The truth of the F-measure. «Teach Tutor Mater», (2007).

3. Алгоритм быстрого нахождения похожих изображений, 2011, https://habr.com/ru/post/122372/.

4. S. Lloyd. Least squares quantization in PCM. «IEEE Transactions on Information Theory», V. 28 (1982), № 2, 129–137.

5. David.G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. «Int. J. Comput. Vision. Norwell», V. 60 (2004), № 2, 91–110, https://doi.org/10.1023/B:VISI.0000029664.99615.94.

6. Herbert Bay, Tinne Tuytelaars, Luc Van Gool. SURF: Speeded up robust features. «Computer vision – ECCV», (2006).

7. Gregory.R. Koch Siamese Neural Networks for One-Shot Image Recognition, (2015).

8. Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision, https://arxiv.org/pdf/1512.00567v1.pdf.

9. Yoshua Bengio, Pascal Lamblin, Dan Popovici, Larochelle Hugo. Greedy Layer-Wise Training of Deep Networks. «Technical Report 1282», (2006).

10. Marc-Olivier.Arsenault. Lossless triplet loss, 2018, February, https://towardsdatascience.com/lossless-triplet-loss-7e932f990b24.

Приложения

А. Описание тестового набора данных

Набор данных, на котором проводилось тестирование алгоритмов проверки, предложенных в данной работе, состоял из 322 изображений, полученных в ходе обучения студентов решению задач по компьютерной графике и работе с OpenGL. Изображения были разбиты на 9 непересекающихся задач:

Таблица 5. Описание тестового набора данных

Идентификатор задачи	**Описание задачи**	**Количество корректных решений**	**Количество некорректных решений**	**Общее количество примеров**
1	Изобразить белый квадрат (залитый внутри) с центром в (0, 0) и длиной ребра 1.	36	5	41
2	Изобразить эллипсоид с полуосями 1, 2, 3 используя VertexBuffer и IndexBuffer. Расчет цвета треугольников должен происходить в fragment шейдере.	16	1	17
3	Изобразить катеноид с a = 2 в области [-4,4]x[-4,4]x[-4,4].	40	10	50
4	Изобразить сферу радиуса 2 в центре (0, 0, 0) в области [-3,3]x[-3,3]x[-3,3]. Цвет треугольников должен быть быть равен (n, (0, 1, 1)) * rgb(0, 0, 1), где n - нормаль к рисуемому треугольнику.	26	18	44
5	Изобразить тетраэдр с длиной ребра 3 в центре (0, 0, 0) в области [-4,4]x[-4,4]x[-4,4] в режиме GL_TRIANGLE_STRIP.	35	5	40
6	Изобразить двуполостный гиперболоид в области [-4,4]x[-4,4]x[-4,4]. a = 1, b = 1, c = 0.5.	10	1	11
7	Изобразить тор с R = 4, r = 1 используя VertexBuffer и IndexBuffer.	33	5	38

Реализация памяти структур данных в векторном потоковом процессоре

Н.И. Дикарев, Б.М. Шабанов, А.С. Шмелёв

Межведомственный суперкомпьютерный центр РАН - филиал ФГУ ФНЦ НИИСИ РАН
nic@jscc.ru, shabanov@jscc.ru, guest8993@rambler.ru

Аннотация: Сложность работы с массивами данных явилась одной из главных причин неконкурентоспособности известных проектов разработки процессора с архитектурой управления потоком данных (потокового процессора). Память структур данных, предназначенная для хранения массивов в этом процессоре, была значительно сложнее по сравнению с обычной (линейно адресуемой) памятью, и при выполнении программ обработки массивов данных требовала в 2 - 3 раза большего числа команд. Показано, что в разрабатываемом векторном потоковом процессоре (ВПП) за счет оригинального метода хранения массивов можно использовать обычную память и значительно увеличить производительность одного процессорного ядра. Приводятся результаты моделирования тестовых программ на этом процессоре и оценка требуемой ёмкости локальной памяти векторов при реализации макета ВПП на ПЛИС.

Ключевые слова: векторный процессор, архитектура управления потоком данных, память структур данных, оценка производительности

.

Введение

В настоящее время становится ясно, что существенно повысить число транзисторов на кристалле БИС за счет уменьшения размеров элементов при изготовлении БИС уже не удастся, и нужно искать другие пути повышения производительности высокопроизводительных вычислительных систем помимо увеличения числа процессорных ядер на кристалле БИС. Также ясно, что не удастся существенно повысить производительность одного процессорного ядра традиционной архитектуры ни за счет роста тактовой частоты, ни за счет увеличения числа команд, выполняемых в такт.

Используемые для повышения производительности фон-неймановского процессора конвейер команд и суперскалярный режим приводят к проблеме, связанной с невозможностью чтения операндов команды до записи результата одной из предыдущих команд по тому же адресу. В результате современные суперсклярные процессоры не могут выполнять более 6 команд в такт.

Процессор с архитектурой управления потоком данных (потоковый процессор) получает значения операндов для выполняемых команд из специальной памяти, сохраняющей результаты предыдущих команд вместе с информацией о командах, которые будут использовать эти результаты, до прихода последнего операнда. Источником такой информации в потоковом процессоре служит граф выполняемой программы, и команды, выданные на выполнение по приходу последнего операнда, уже содержат значения операндов. Соответственно их не приходится читать из линейно адресуемой памяти, как у процессора традиционной архитектуры, и конфликтов информационной зависимости не может быть в принципе. Поэтому в потоковом процессоре можно достичь более высокой производительности.

В МСЦ РАН разрабатывается векторный потоковый процессор (ВПП), в котором устранены многие недостатки известных проектов потокового процессора и подтверждена более высокая производительность процессорного ядра [1]. Цель данной работы – показать, каким образом в разрабатываемом процессоре можно преодолеть одну из главных причин неконкурентоспособности известных проектов разработки потокового процессора – сложность реализации памяти структур данных, предназначенной для хранения и работы с массивами данных в этом процессоре, и подтвердить это результатами моделирования.

Память структур данных в потоковом процессоре и сложности её реализации

Принцип работы потокового процессора в корне отличается от процессора традиционной (фон-неймановской) архитектуры. Если работа фон-

неймановского процессора основана на использовании линейно адресуемой памяти, к которой относятся оперативная память и регистровые файлы, то в основе функционирования потокового процессора лежит память поиска пар (ППП) готовых операндов, которая и выдаёт готовые команды на выполнение. Соответственно, если в фон-неймановском процессоре источником операндов для выполняемых команд служит оперативная память или регистровые файлы, и туда же записываются результаты команд, то в потоковом процессоре готовые команды, выдаваемые из ППП, содержат операнды в явном виде для подачи на вход исполнительного устройства (ИУ). Вычисленный в ИУ результат отправляется в составе пакета – токена в ППП для поиска тех команд, в которых согласно графу программы этот результат будет использоваться.

Наконец, если в фон-неймановском процессоре записанные в память результаты можно читать много раз до записи нового значения по тому же адресу, то в потоковом процессоре принцип единственного присваивания операндов, в соответствии с которым операнды выполняемой команды уничтожаются, не предусматривает само наличие памяти для хранения массивов данных. Конечно, такая память необходима для работы процессора, и большинство проектов разработки потокового процессора имело память структур данных в своём составе. Часто в качестве такой памяти использовалась память I-структур (the I-structure scheme) [2, 3], которая позволяет создавать одномерные массивы произвольной длины, подавать запросы на чтение элементов массива до записи массива целиком и, более того, до записи нужного элемента. В последнем случае запросы на чтение элемента хранятся в памяти I-структур до момента записи элемента в память, и токены результаты операции чтения выдаются сразу после записи элемента. При этом допускается лишь однократная запись элемента, чтобы исключить конфликты запись после записи.

К недостаткам метода I-структур относят большие аппаратные затраты (из-за необходимости реализации списков отложенных запросов по чтению к еще незаписанным элементам) и дополнительные потери времени при чтении элементов массива (из таких списков). Кроме того, при работе с такой памятью нужно после выполнения всех команд чтения к массиву проводить "сборку мусора", то есть стирать массив, чтобы освободить ресурс памяти. Наконец, если в программе требуется изменить один или несколько элементов существующего массива, то из-за требования однократной записи элемента нужно создать новый массив, в который нужно копировать почти все элементы исходного массива, поскольку могут оставаться ещё не выполненные команды чтения к исходному массиву.

Таким образом, произвольный порядок выполнения команд в потоковом процессоре, связанный с динамикой поступления токенов операндов в ППП, приводит к необходимости исключения конфликтов информационной зависимости при работе с памятью структур данных. Реализация памяти структур данных по методу I-структур обеспечивает отсутствие конфликтов, но за счёт значительно больших аппаратных затрат по сравнению с обычной памятью. Кроме того, избыточное копирование массивов при необходимости модификации его элементов, а также необходимость "сборки мусора" приводят к увеличению общего числа выполняемых команд в потоковом процессоре, которое оказывается в 2 - 3 раза больше, чем у фон-неймановского процессора [4].

С обработкой и хранением массивов данных в потоковых ЭВМ тесно связана проблема избыточного параллелизма. Естественный параллелизм у многих задач при увеличении размера обрабатываемых массивов может быть настолько большим, что для одновременного выполнения всех готовых команд не хватит ресурсов ни в каком реальном процессоре [5]. Для недопущения перегрузки ИУ применяется система автоматического регулирования нагрузки, которая использовалась в проекте Манчестерской потоковой машины (MDFM) [3]. Система авторегулирования реагирует на превышение числом готовых команд на входе ИУ заданного порога и приостанавливает выдачу новых гранул работы, таких как выполнение циклов и подпрограмм. Поскольку для выполнения нового программного блока нужно найти свободный фрагмент необходимого размера в памяти структур данных, то большое время на поиск такого фрагмента памяти увеличивает инерционность системы авторегулирования. В MDFM это проявилось на программе умножения матриц, когда даже при малом размере обрабатываемых матриц число команд, ожидающих выполнения в ИУ,

достигало 6764, а число токенов в ППП - 39500 [3].

Реализация памяти в векторном потоковом процессоре

Память структур данных в ВПП имеет два уровня - быструю локальную память на процессорном кристалле и основную с большой ёмкостью на микросхемах динамической памяти. Эта память, как и в известных проектах потокового процессора с использованием I-структур реализуется на основе линейно адресуемой памяти, но имеет два существенных отличия. Во-первых, в ВПП помять под создаваемые массивы выделяется фрагментами одной и той же длины (256 слов), а не произвольной длины как в методе I-структур. Во-вторых, до записи всех элементов массива не допускается к нему обращений по чтению, то есть токен с указателем массива выдаётся, когда все его элементы уже записаны. Тем самым в ВПП исключаются конфликты информационной зависимости RAW, и нет необходимости усложнять память введением списков отложенных запросов по чтению к еще незаписанным элементам, как это имеет место в памяти I-структур.

Память структур данных с такими ограничениями будет сужать параллелизм выполняемых в процессоре операций и, следовательно, его производительность, но лишь на скалярной обработке. При векторной обработке, на которой ВПП и показывает высокую производительность (256 флоп в такт), все элементы вектора могут обрабатываться и записываться одновременно. Тогда выделяя в памяти под каждый результат векторной команды свободный фрагмент с числом слов, равным аппаратной длине вектора VL_{max}=256, и формируя токен с указателем вектора после записи его элементов, мы гарантируем отсутствие конфликтов RAW (чтения после записи).

Заметим, что указатель вектора, переданный в составе токена на вход векторной команды, позволяет читать его элементы в любом из двух уровней иерархии памяти в ВПП. А именно, из локальной памяти векторов (ЛПВ) и из основной памяти векторов (ПВ), на что указывает специальный бит, сопровождающий адрес в указателе вектора. Это бит проставляется в указателе вектора результата при выполнении векторной команды в соответствии с тем, как это было указано компилятором или программистом при создании графа программы. Основная цель – локализовать обращения к памяти в процессе вычислений в пределах быстрой ЛПВ.

Определение адреса в памяти для записи результата векторной команды производится в ВПП аппаратно в момент выдачи векторной команды на выполнение, то есть в динамике. При одинаковой длине выделяемых для записи вектора результата фрагментов памяти аппаратное распределение ресурса ПВ и ЛПВ можно осуществить при наличии списков свободных векторов, как для ПВ, так и для ЛПВ. Новый адрес вектора в ВПП может выдаваться из такого списка каждый такт. Токен с указателем вектора после записи его элементов передается в ППП и активирует выполнение тех векторных команд, которые используют его в качестве операнда.

Заметим, что вектор, записанный в ПВ или ЛПВ, можно читать неоднократно. Нужно лишь пометить признаком стирания вход команды, которая будет использовать этот вектор в последний раз, либо использовать специальную команду «удалить вектор» по завершению выполнения определённого программного блока. Причем доля команд «удалить вектор» и команд синхронизации для задания момента возвращения вектора в список свободных оказывается настолько малой, что не влияет на производительность ВПП.

Таким образом, указатель вектора результата в ВПП, как и скалярный операнд, передаётся на входы последующих команд, дублируется в случае необходимости и уничтожается при последнем использовании. При этом выделение свободного места в памяти ВПП производится аппаратно и, следовательно, значительно быстрее, чем в памяти I-структур, в которой из-за произвольной длины создаваемого массива для выделения места в памяти требуется вмешательство операционной системы. При этом для хранения больших массивов в ВПП предлагается использовать вектора, элементами которых являются указатели векторов более низкого уровня, последний из которых хранит числовые данные. Это позволяет одной командой «формирование потока» считать и выдать все необходимые токены из элементов вектора указателя матрицы для команд обработки векторов строк этой матрицы во всех итерациях цикла. Таким образом, команда «формирование потока» позволяет устранить цикл, который

в фон-неймановском процессоре необходим для выборки из массива векторов с целью их последующей обработки [1].

Аппаратное распределение памяти, реализованное в ВПП, позволяет выделять память для записи результатов векторных команд по мере необходимости в процессе выполнения программы и возвращать вектора в список свободных по мере их использования. Это позволяет экономно расходовать ресурс памяти, что особенно важно для ЛПВ. Кроме того, быстрое распределение памяти позволило осуществить в ВПП регулировку параллелизма, приостанавливая или разрешая выдачу не крупных программных блоков, а отдельных векторных команд. Такая тонкая регулировка параллелизма уменьшила число команд, ожидающих выполнения в очереди на входе ИУ, и токенов в ППП примерно в 100 раз по сравнению с MDFM на той же программе умножения матриц. При этом реальная производительность ВПП на этой программе оказалась близка к пиковой производительности [1].

Заключение

Для моделирования времени выполнения программ в ВПП использовалась его VHDL модель уровня регистровых станций, в основу которой легла проработка таких ключевых устройств процессора как ППП и расслоенная на большое число банков локальная память векторов. Моделирование показало, что на программе умножения матриц реальная производительность одного ядра ВПП не только превосходит производительность вычислительной системы из 16 ядер Intel Xeon, но и сохраняет её при меньшем размере обрабатываемых матриц [1]. Более высокую производительность показывает ВПП и на программах сортировки, а также решения систем дифференциальных уравнений 2D и 3D Stencil. Так на программе битонная сортировка и сортировка с использованием команд редукции, производительность ВПП оказалась в 4 – 7 раз выше, чем у Intel Xeon [6].

Предварительная оценка возможности реализации макета ВПП на одном кристалле ПЛИС показала, что при использовании последнего поколения Virtex UltraScale+ фирмы Xilinx на кристалле ПЛИС с максимальной степенью интеграции XC VU13P можно разместить ВПП с производительностью 32 – 64 флоп в такт. Причем ограничение производительности макета ВПП обусловлено реализацией большого числа ИУ с плавающей точкой и коммутаторами, с помощью которых входы и выходы этих ИУ подключаются к банкам ЛПВ. Что же касается требуемой ёмкости этой памяти, а именно 2 MB, необходимых для хранения 1 K векторов по 256 слов, то такую память можно реализовать, поскольку ёмкость памяти XC VU13P превышает 7 MB.

Работа выполнена в МСЦ РАН в рамках Государственного задания по теме 0065-2019-0016 (рег. номер АААА-А19-119011590098-8). В исследованиях использовался суперкомпьютер МВС-10П.

Structure store implementation in vector dataflow processor

N.I. Dikarev, B.M. Shabanov, A.S. Shmelev

Abstract: Array processing complexity was one of the main reasons of not competitiveness known dataflow processor projects. Structure store was much more complicated compared to conventional (linearly addressable) memory, and array processing requires 2-3 times more instructions in dataflow processor. It's shown that the developing vector dataflow processor (VDP), due to the original method of storing arrays, is capable of using conventional memory and the performance of one processor core is significantly higher. This paper presents results of test programs simulation on VDP and assessing the required capacity of the local vector memory when implementing VDP project on FPGA.

Keywords: Vector processor; dataflow architecture; shared-memory multiprocessor; structure store, performance evaluation.

Литература

1. Н.И.Дикарев, Б.М.Шабанов, А.С.Шмелёв. Векторный потоковый процессор: оценка производительности. «Известия ЮФУ. Технические науки», 2014, № 12, 36 – 46.

2. Arvind and R.S.Nikhil. Executing a program on the MIT tagged-token data-flow architecture. «IEEE Transactions on Computers», vol. 39 (1990), № 3, 300 – 318.

3. J.Gurd et.al. Performance Issues in Dataflow Machines. «Future generations computer systems», vol. 3 (1987), № 4, 285 – 297.

4. G.V.Papadopoulos, K.R.Traub, Multithreading: A revisionist view of dataflow architectures. «Proc. 18-th Ann. Symp. on Computer Architecture», 1991, 342 – 351.

5. D.E.Culler and Arvind. Resource Requirements of Dataflow Programs. «Proc. 15-th Ann. Symp. on Computer Architecture», 1988, 141 – 150.

6. Н.И.Дикарев, Б.М.Шабанов, А.С.Шмелёв. Быстрые алгоритмы сортировки для векторного потокового процессора. «Суперкомпьютерные технологии (СКТ-2018) (17 - 22 сентября 2018 г.), Материалы 5-й Всероссийской научно-технической конференции», Ростов-на-Дону – Таганрог, изд-во ЮФУ, 2018, т. 1, 87 – 91.