ИНСТИТУТ ПРОГРАММНЫХ СИСТЕМ
ИМЕНИ А.К.АЙЛАМАЗЯНА
РОССИЙСКОЙ АКАДЕМИИ НАУК

25 ЛЕТ

ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ

Институт программных систем им. А. К. Айламазяна
Российской академии наук

Программные системы: теория и приложения

Труды международной конференции
г. Переславль-Залесский, май 2009

Том 1

Переславль-Залесский

УДК 519.71
ББК 22.18

П78

Программные системы: теория и приложения. // Труды международной конференции «Программные системы: теория и приложения», ИПС им. А. К. Айламазяна РАН, г. Переславль-Залесский, май 2009 / *Под редакцией С. М. Абрамова и С. В. Знаменского.* В двух томах. — Переславль-Залесский: Изд-во «Университет города Переславля», 2009. — Т. 1, 308с., ил. — ISBN 978-5-901795-16-3

Program systems: Theory and applications. // Proceedings of international conference of Program Systems Institute named A. K. Ailamazyan of Russian Academy of Sciences, May 2009 / *Edited by S. Abramov and S. Znamenskij.* In two volumes. — Pereslavl-Zalesskij: "Pereslavl university", 2009. — Vol. **I**, 308p. — ISBN 978-5-901795-16-3

В первый том сборника включены статьи, представленные по направлениям: *Оптимальное управление; Системный анализ; Интеллектуальное управление; Интеллектуальные Интернет–технологии; Системное программное обеспечение вычислительных серверов и параллельные вычислительные системы; Моделирование социо-эколого-экономических систем.*

Во второй том сборника вошли статьи, представленные по направлению *Большие информационные системы*

Для научных работников, аспирантов и студентов, интересующихся современным состоянием фундаментальных исследований в области информатики и программирования.

Конференция проводится при финансовой поддержке Российской академии наук и Российского Фонда Фундаментальных Исследований (проект № 09-07-06014-г)

В сборнике сохранены авторские орфография и оформление.

ISBN 978-5-901795-16-3

Научное издание

Труды конференции

Труды международной конференции
«Программные системы: теория и приложения»
ИПС РАН, г. Переславль-Залесский, май 2009
Для научных работников, аспирантов и студентов

Редакционная коллегия сборника: **С. М. Абрамов**, В. И. Гурман, В. М. Хачумов, А. М. Цирлин, С. В. Знаменский, Ю. Л. Сачков, Е. В. Рюмина, С. А. Амелькин, Я. И. Гулиев, В. Б. Новосельцев.

Том I

Ответственный за выпуск *С. В. Знаменский*
Дизайн обложки *Е. В. Шафранская и И. В. Шафранский*

Изд. лиц. ИД № 01389 от 30.03.2000
Подписано к печати 27.04.2009 Гарнитура Computer Modern (LH)
Формат 60 × 84/16 Усл. печ. л. 17,28 Уч.-изд. л. 19,25

Издательство «Университет города Переславля»

Издательство Университета города Переславля

Отпечатано в ЗАО "Атрус"152151 Ярославская обл., г. Ростов, ул. Луначарского д. 48 Тираж 150 экз. Заказ № _____

ISBN 978-5-901795-16-3

ИНСТИТУТ ПРОГРАММНЫХ СИСТЕМ
ИМЕНИ А.К.АЙЛАМАЗЯНА
РОССИЙСКОЙ АКАДЕМИИ НАУК

Россия, Ярославская обл., г. Переславль-Залесский
тел./факс. +7 (48535) 98-064
WWW.PSI-RAS.RU

Предисловие

По решению Правительства, направленному на развитие вычислительной техники и информатики в стране в апреле 1984 г. был создан Институт программных систем (ИПС) АН СССР[1].

Идея создания Института в г. Переславле-Залесском принадлежала одному из активных и плодотворных ученых современности — академику Евгению Павловичу Велихову. Эту идею подхватил и воплотил в жизнь основатель — профессор Альфред Карлович Айламазян, который был бессменным директором Института вплоть до весны 2003 г.

Таким образом, в текущем 2009 году исполняется 25 лет Институту программных систем Российской академии наук. И нам особенно приятно, что недавно решением Президиума нашему Институту присвоено имя А. К. Айламазяна.

Альфред Карлович Айламазян заложил в Институте ядро научного коллектива, который можно сегодня характеризовать как один из самых динамичных и высококвалифицированных коллективов, работающих в Отделении нанотехнологий и информационных технологий РАН.

В его составе в настоящее время около 150 научных сотрудников, в том числе один академик РАН, один член-корреспондент РАН, 13 докторов наук, 20 кандидатов наук, многие из которых, несмотря на свой сравнительно молодой возраст, получили широкое признание в научном мире.

С первых лет работы Институт уделял внимание не только фундаментальным академическим исследованиям в области информатики и программирования, но и вопросам доведения своих разработок до внедрения их в высокотехнологичные отрасли (которые по своей природе готовы к внедрению новых IT-технологий) страны. Такой стиль ведения исследований и разработок был заложен в коллективе еще профессором А. К. Айламазяном.

Важнейшим событием в деле развития Института явилось создание в 1993 году впервые в малом городе России университета — Университета города Переславля (УГП) имени А. К. Айламазяна. Основной костяк профессорско-преподавательского состава в УГП имени

[1]Институт был создан вначале как Филиал Института проблем кибернетики (ФИПК) АН СССР, а в 1986 году ФИПК АН СССР был преобразован в самостоятельный Институт программных систем АН СССР

А. К. Айламазяна образуют сотрудники ИПС имени А. К. Айламазяна РАН.

Наши выпускники и студенты (в рамках курсового и дипломного проектирования) достойно трудятся в Институте, на предприятиях и в научных учреждениях региона, столицы и за рубежом. Нацеленные на научную карьеру выпускники УГП имени А. К. Айламазяна заканчивают аспирантуру, получают первые свои научные степени. Сегодня более 40% штата Института являются выпускниками и студентами УГП имени А. К. Айламазяна. Они сегодня подтверждают высокий уровень УГП имени А. К. Айламазяна в деле научной работы и подготовки специалистов — в чем читатель легко может убедиться, просмотрев работы, включенные в данный сборник, авторами и соавторами большинства из которых являются наши студенты и выпускники.

Мы благодарны всем авторам, кто принял участие в подготовке нашей конференции и данного сборника. Особая признательность коллегам из ряда академических институтов, вузов, нашим зарубежным друзьям, без участия и поддержки которых вряд ли состоялся бы и этот сборник, да и сам Институт программных систем имени А. К. Айламазяна РАН, по крайней мере в том состоянии и статусе, который сегодня есть у него.

Я надеюсь, данный сборник объективно отражает широкий научный спектр проводимых исследований, оригинальность представленных решений и перспективы будущих работ. Дорогой читатель, примите этот сборник, как определенный итог деятельности Института за прошедшие 25 лет.

С. М. Абрамов,

Директор Института программных систем РАН,
член-корреспондент РАН

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 517

Ю. Л. Сачков, А. А. Ардентов, А. П. Маштаков

Конструктивное решение задачи управления на основе метода нильпотентной аппроксимации

Аннотация. Описан метод нильпотентной аппроксимации нелинейных управляемых систем с линейным управлением. Представлен алгоритм решения задачи управления для таких систем на основе нильпотентной аппроксимации. Рассматривается реализация этого алгоритма в классах оптимальных и кусочно-постоянных управлений.

1. Постановка задачи и определения

Работа посвящена исследованию нелинейных управляемых систем с линейными управлениями:

$$\dot{x} = \sum_{i=1}^{m} u_i g_i(x),\ x \in M,\ u = (u_1, \dots, u_m) \in \mathbb{R}^m, \tag{1.1}$$

где M — связное гладкое многообразие, g_i — аналитические векторные поля на M, а $u_i(\cdot)$ — измеримые локально ограниченные управления.

Рассматривается следующая задача управления. Для заданных точек $p, q \in M$ и времени $T > 0$, требуется найти управление $u(t)$, $t \in [0, T]$, переводящее систему (1.1) из точки p в точку q:

$$x(0) = p, \qquad x(T) = q.$$

Напомним, что система называется вполне управляемой, если для любых $p, q \in M$ эта задача управления разрешима для некоторого $T > 0$. Для линейной по управлениям системы (1.1) критерий полной управляемости дается в терминах алгебры Ли векторных полей, порожденной полями в правой части системы:

$$\mathrm{Lie}(g_1, \dots, g_m) = \mathrm{span}(g_1, \dots, g_m, [g_i, g_j], [g_i, [g_j, g_k]], \dots).$$

Определение 1. Система (1.1) имеет полный ранг в точке $x \in M$, если

$$\mathrm{Lie}(g_1, \dots, g_m)(x) = T_x M.$$

Работа поддержана Российским Фондом Фундаментальных Исследований, проект No. 09-01-00246.

Теорема 1 (Рашевский–Чжоу, [1]). *Система* (1.1) *вполне управляема на* M *тогда и только тогда, когда она имеет полный ранг в любой точке* $x \in M$.

Несмотря на то, что вопрос управляемости для линейных по управлению систем (1.1) имеет полное решение (ответ сводится к дифференцированию векторных полей и вычислению размерности их линейной оболочки), задача построения управления, переводящего систему из одной точки в другую, весьма нетривиальна (в основном для нас случае, когда размерность пространства состояний меньше, чем размерность управлений). Основное препятствие заключается в том, что такие системы имеют существенно разные скорости перемещения по разным направлениям. Величина смещения в направлении полей g_i за малое время t есть $O(t)$, в направлении коммутаторов $[g_i, g_j]$ есть $O(t^2)$, в направлении $[g_i, [g_j, g_k]]$ есть $O(t^3)$, и т.д. Даже для систем, имеющих наиболее простую структуру при сохранении свойства управляемости (нильпотентных), эта задача не имеет общего решения, и должна рассматриваться независимо для каждой канонической формы нильпотентной системы в каждой размерности.

Прежде чем перейти к алгоритмам приближенного решения задачи управления, рассмотрим следующие основные примеры, играющие важную роль в механике и робототехнике.

Пример 1 (Качение шара по плоскости). Рассмотрим сферу, катящуюся по горизонтальной плоскости без прокручивания и проскальзывания; например, сферу, катящуюся между двумя горизонтальными плоскостями — неподвижной нижней и подвижной верхней плоскостью.

Обозначим через $(x_1, x_2) \in \mathbb{R}^2$ точку контакта сферы и плоскости, и через $R \in \mathrm{SO}(3)$ ортогональную матрицу, задающую ориентацию сферы в трехмерном пространстве. Пусть E_{ij} обозначает 3×3 матрицу, все элементы которой равны нулю, кроме элемента в i-ой строке и j-ом столбце, который равен единице. Тогда движение системы описывается следующей управляемой системой [1, 2]:

$$\dot{x}_1 = u_2, \qquad \dot{x}_2 = u_2, \tag{1.2}$$

$$\dot{R} = u_1 R(E_{31} - E_{13}) + u_2 R(E_{23} - E_{32}). \tag{1.3}$$

Эта система вполне управляема и имеет вектор роста $(2, 3, 5)$.

Отметим, что качение двух произвольных поверхностей без проскручивания и проскальзывания также моделируется системой вида (1.1) с двумерным управлением и пятимерным пространством состояний; если в точке касания поверхности имеют разные гауссовы кривизны, то система также имеет вектор роста $(2, 3, 5)$, и имеет ту же нильпотентную аппроксимацию, что и система (1.2), (1.3). Система вполне управляема тогда и только тогда, когда катящиеся поверхности неизометричны, см. [1].

Пример 2 (Машина с двумя прицепами). Рассмотрим модель машины на плоскости с центром масс $(x, y) \in \mathbb{R}^2$ и углом наклона $\theta \in S^1$ относительно положительного направления оси x. Пусть к машине последовательно присоединены два прицепа. Обозначим через φ_1 угол поворота первого прицепа относительно машины, и через φ_2 угол поворота второго прицепа относительно первого. Соответствующая управляемая система имеет вид

$$\dot{q} = u_1 X_1(q) + u_2 X_2(q), \qquad q = (x, y, \theta, \varphi_1, \varphi_2),$$
$$X_1 = \cos\theta \frac{\partial}{\partial x} + \sin\theta \frac{\partial}{\partial y} - \sin\varphi_1 \frac{\partial}{\partial \varphi_1} + (\sin\varphi_1 + \sin(\varphi_1 - \varphi_2)) \frac{\partial}{\partial \varphi_2},$$
$$X_2 = \frac{\partial}{\partial \theta} - (1 + \cos\varphi_1) \frac{\partial}{\partial \varphi_1} + (\cos\varphi_1 + \cos(\varphi_1 - \varphi_2)) \frac{\partial}{\partial \varphi_2}.$$

Эта система вполне управляема, и имеет вектор роста $(2, 3, 5)$ в точках, где $\varphi_1 \neq \varphi_2$.

Конструктивная задача управления активно изучается в последнее время в современной нелинейной теории управления. Эта задача имеет удовлетворительное решение в случае систем, находящихся в цепной форме, и приводимых к такой форме обратной связью [3, 4], а также для дифференциально плоских систем [5]. Однако системы общего положения с двумя управлениями имеют вектор роста $(2, 3, 5)$ (как, например, в примерах 1, 2), потому эти результаты неприменимы к таким системам.

В данной работе мы рассматриваем метод решения задачи управления, основанный на нильпотентной аппроксимации. Идея метода заключается в том, что управляемая система локально приближается более простой системой (нильпотентной), для которой задача

управления часто может быть решена точно. Управления, точно решающие нильпотентную систему, дают приближенное решение исходной задачи управления в малой окрестности целевой точки. Метод нильпотентной аппроксимации применим к задачам управления общего вида; существенно только уметь решать задачу управления для нильпотентной аппроксимации. Этот метод предложен в работах [6–10].

Цель данной работы — применение метода нильпотентной аппроксимации к системам (1.1) общего вида на пятимерных многообразиях с двумерным управлением. Для нильпотентных аппроксимаций таких систем в последние годы было получено точное решение задачи управления в классе оптимальных управлений (в смысле функционала субримановой длины) [11–14], которое будет основным инструментом в решении задачи управления для общих нелинейных систем.

Напомним определения некоторых важных понятий.

Для точки $x_0 \in M$, обозначим через η траекторию системы (1.1), выходящую из точки x_0 под действием управления $u(t), t \in [0,T]$. Длина траектории определяется как

$$\operatorname{length}(\eta) = \int_0^T \sqrt{u_1^2(t) + \dots + u_m^2(t)}dt. \tag{1.4}$$

Система (4.1) индуцирует субриманово расстояние в $\mathbb{R}^n$, определяемое как

$$d(x_1, x_2) = \inf_{\eta} \operatorname{length}(\eta) \tag{1.5}$$

с инфимумом, берущимся по всем траекториям η, соединяющим x_1 с x_2. Из теоремы Рашевского–Чжоу следует, что если система (1.1) имеет полный ранг, то для любых точек $x_1, x_2 \in M$ выполняется неравенство $d(x_1, x_2) < \infty$.

Зафиксируем $x_0 \in M$ и обозначим через $L^s(x_0)$ векторное пространство, порождаемое значениями в точке x_0 скобок Ли векторных полей $g_1,\dots, g_m$ длины $\leq s$, $s = 1, 2, \dots$ (сами поля g_i — скобки длины 1). Условие полного ранга гарантирует, что существует наименьшее целое число $r = r(x_0)$ такое, что $\dim L^r(x_0) = n = \dim M$. Это число r называется порядком неголономности системы (1.1) в точке x_0.

Вектором роста системы (1.1) в точке x_0 называется вектор с компонентами $(n_1(x_0), \dots, n_r(x_0))$, где $n_s(x_0) = \dim L^s(x_0)$, $s = 1, \dots, r$.

Точка x_0 называется регулярной, если вектор роста постоянен в окрестности точки x_0, иначе x_0 называется сингулярной. Далее будем предполагать, что все точки системы (1.1) регулярны.

Управляемая система (1.1) называется нильпотентной, если соответствующая алгебра Ли $Lie(g_1, \dots, g_m)$ нильпотентна, то есть для некоторого $N \in \mathbb{N}$

$$[g_{i_1}, [g_{i_2}, \dots, [g_{i_N}, g_{i_{N+1}}] \dots]] = 0, \ \forall i_1, \dots, i_{N+1} \in \{1, \dots, m\}.$$

2. Нильпотентная аппроксимация неголономных систем

Понятие неголономной аппроксимации управляемых систем было предложено независимо А. А. Аграчевым и А. В. Сарычевым [6] и Х.Хермсом [8]. Инвариантная конструкция нильпотентной аппроксимации была получена А. А. Аграчевым и А. Мариго.

Локальное приближение управляемой системы другой, более простой системой, широко используется в теории управления. Обычно в качестве локальной аппроксимации используется линеаризация управляемой системы. Однако для линейных по управлению систем вида (1.1) линеаризация дает слишком грубое приближение. Если размерность управления меньше размерности состояния (этот важный случай наиболее интересен), то линеаризация не может быть вполне управляемой. Естественную замену линейной аппроксимации в этом случае доставляет нильпотентная аппроксимация — наиболее простая система, сохраняющая структуру управляемости исходной системы (в частности, сохраняется такой важный инвариант как вектор роста).

Во избежание излишних технических деталей, опишем свойства нильпотентной аппроксимации в основном для нас случае — для систем с пятимерным состоянием и двумерным управлением:

$$\dot{q} = u_1 X_1(q) + u_2 X_2(q), \qquad q \in M, \dim M = 5, \quad (u_1, u_2) \in \mathbb{R}^2, \tag{2.1}$$

в предположении, что в рассматриваемой точке $q_0 \in M$ эта система имеет вектор роста $(2, 3, 5)$, т.е. следующие векторные поля линейно независимы:

$$X_1, \quad X_2, \quad X_3 = [X_1, X_2], \quad X_4 = [X_1, X_3], \quad X_5 = [X_2, X_3].$$

Известно, что для системы (2.1) общего положения это условие на вектор роста выполняется в точке q_0 общего положения.

В некоторой окрестности $U_{q_0} \subset M$ точки q_0 существует пара векторных полей (Y_1, Y_2), доставляющих нильпотентную аппроксимацию системы (X_1, X_2) в точке q_0, со следующими свойствами. Алгебра Ли

$$L = \mathrm{Lie}(Y_1, Y_2)$$

нильпотентна и порождена как линейное пространство векторными полями

$$Y_1,\ Y_2,\ Y_3 = [Y_1, Y_2],\ Y_4 = [Y_1, Y_3],\ Y_5 = [Y_2, Y_3], \tag{2.2}$$

все остальные скобки Ли в L либо равны нулю, либо следуют из (2.2) в силу линейности и кососимметричности. Существует система координат $\Phi : U_{q_0} \to \mathbb{R}^5$, называемая привилегированной, в которой Y_i, $i = 1, \dots, 5$, становятся полиномиальными векторными полями. Пространство $\mathbb{R}^5$ может быть отождествлено со связной односвязной группой Ли G, соответствующей алгебре Ли L.

Управляемая система (2.1) и субриманова структура

$$\Delta = \mathrm{span}(X_1, X_2), \quad \langle X_i, X_j \rangle = \delta_{ij},\ i, j = 1, 2 \tag{2.3}$$

приближаются в окрестности U_{q_0} соответственно системой

$$\dot g = u_1 Y_1(g) + u_2 Y_2(g), \quad g \in G, \quad u_1,\, u_2 \in \mathbb{R}, \tag{2.4}$$

и субримановой структурой

$$D = \mathrm{span}(Y_1, Y_2), \quad \langle Y_i, Y_j \rangle = \delta_{ij},\ i, j = 1, 2, \tag{2.5}$$

в следующем смысле.

Пусть $q(t)$ и $g(t)$ суть траектории систем (2.1) и (2.4), соответствующие некоторым допустимым управлениям $u_1(t)$, $u_2(t)$, $u_1^2(t) + u_2^2(t) \equiv 1$, выходящие из точки $Q \ni q_0 = e \in G$.

В адаптированных координатах $(g_1, \dots, g_5)$ в окрестности U_{q_0} рассмотрим дилатации

$$\delta_\varepsilon : (g_1, g_2, g_3, g_4, g_5) \mapsto (\varepsilon g_1, \varepsilon g_2, \varepsilon^2 g_3, \varepsilon^3 g_4, \varepsilon^3 g_5).$$

Траектории $g(t)$ нильпотентной системы (2.4) однородны относительно этих дилатаций:

$$\delta_\varepsilon g(t) = g(\varepsilon t)$$

и приближают траектории $q(t)$ системы (2.1):

$$\delta_{\frac{1}{t}}(q(t) - g(t)) = O(t), \quad t \to 0.$$

Существуют такие $\gamma > 0$ и $C > 0$, что для всех $|t| < \gamma$ выполнено неравенство

$$d_G(q(t), g(t)) \leq Ct^{4/3},$$

где d_G есть субриманово расстояние на группе Ли G, соответствующее субримановой структуре (2.5):

$$d_G(g_0, g_1) = \inf \left\{ \int_0^1 \sqrt{u_1^2(t) + u_2^2(t)}\, dt \right\},$$

инфимум берется по всем траекториям $g(\cdot)$ системы (2.4), для которых $g(0) = g_0$, $g(1) = g_1$.

3. Алгоритм нильпотентной аппроксимации

А.Белаиш [7] разработал алгебраический алгоритм вычисления нильпотентной аппроксимации системы вида (1.1) в окрестности регулярной точки $x_0 \in M$. Далее в пункте 3.1 этот алгоритм изложен на основе работы М. Вендиттелли и соавторов [10]. В пункте 3.2 этот алгоритм конкретизируется для систем (2.1) с вектором роста $(2, 3, 5)$.

В силу локальности процедуры нильпотентной аппроксимации, в этом разделе полагаем $M = \mathbb{R}^n$.

3.1. Нильпотентная аппроксимация в привилегированных координатах

Рассмотрим систему (1.1), аппроксимируемую в точке $x_0 \in \mathbb{R}^n$. Алгоритм для вычисления привилегированных координат и построения нильпотентной аппроксимации в точке x_0 выглядит следующим образом:

(1) В точке x_0 вычисляются вектор роста $(n_1, \ldots, n_r)$ и веса w_1, $\ldots$, w_n. Веса определяются из условия $w_j = s$, если

$$n_{s-1} < j \leq n_s,$$

где $n_s = n_s(x_0)$ и $n_0 = 0$.

(2) Выбираются такие векторные поля $\gamma_1, \ldots, \gamma_n$, что их значения в точке x_0 образуют базис в касательном пространстве

$$L^r(x_0) = T_{x_0}\mathbb{R}^n$$

и для них выполнено условие

$$\gamma_{n_{s-1}+1}(x), \ldots, \gamma_{n_s}(x) \in L^s(x), \;\; s = 1, \ldots, r$$

для любой точки x в окрестности точки x_0.

(3) Используя исходные координаты x, в которых задана система (1.1), вычисляются новые координаты y на пространстве состояний $\mathbb{R}^n$ следующим образом:

$$\alpha: \qquad y = \Gamma^{(-1)}(x - x_0), \tag{3.1}$$

где Γ — это матрица размерности $n \times n$, составленная из элементов Γ_{ij}, определяемых по формуле:

$$\gamma_j(x_0) = \sum_{i=1}^{n} \Gamma_{i,j}\, \partial x_i|_{x_0}.$$

(4) Вычисляются функции, задающие привилегированные координаты $z = (z_1, \dots, z_n)$ в окрестности точки x_0 по рекуррентной формуле:

$$\lambda: \qquad z_j = y_j + \sum_{k=2}^{w_j - 1} h_k(y_1, \dots, y_{j-1}),\ j = 1, \dots, n$$

, где

$$h_k(y_1, \dots, y_{j-1}) = - \sum_{\substack{|\alpha| = k \\ w(\alpha) < w_j}} m_j \gamma_1^{\alpha_1} \dots \gamma_{j-1}^{\alpha_{j-1}} (y_j + \sum_{q=2}^{k-1} h_q)(x_0),$$

и $m_j = \prod_{i=1}^{j-1} y_i^{\alpha_i}/\alpha_i!$ и $|\alpha| = \sum_{i=1}^{j-1} \alpha_j$.

(5) Динамика исходной системы (1.1) выражается в привилегированных координатах:

$$\dot z = \sum_{i=1}^{m} g_i(z) u_i.$$

(6) Строится разложение в ряд Маклорена векторных полей $g(z)$ и группируются поля одного веса:

$$g_i(z) = g_i^{(-1)}(z) + g_i^{(0)}(z) + g_i^{(1)}(z) + \dots,$$

где $g_i^{(s)}$ — поле веса s.

(7) Определяются поля $\hat g_i(z) = g_i^{(-1)}(z)$ и аппроксимирующая система

$$\dot z_j = \sum_{i=1}^{m} \hat g_{ij}(z_1, \dots, z_{j-1}) u_i,\ j = 1, \dots, n,$$

где $\hat g_{ij}$ — однородный полином веса $w_j - 1$.

Так выглядит общий алгоритм построения нильпотентной аппроксимации в привилегированных координатах. Для систем (2.1) с вектором роста $(2,3,5)$ конкретизируется следующим образом.

3.2. Аппроксимация систем с вектором роста $(2,3,5)$

Рассмотрим систему (2.1) в пространстве $\mathbb{R}^5$, имеющую в некоторой точке $x_0 \in \mathbb{R}^5$, а потому и в некоторой ее окрестности, вектор роста $(2,3,5)$. Опишем процедуру построения нильпотентной аппроксимации в точке x_0.

(1) Вычисляются коммутаторы

$$X_3 = [X_1, X_2], \qquad X_4 = [X_1, X_3], \qquad X_3 = [X_2, X_3].$$

Веса полей $X_1, \dots, X_5$ соответственно равны $(1,1,2,3,3)$.

(2) В качестве векторных полей γ_i выбираются поля X_i. Для них выполняются все условия, наложенные в пункте 3.1 на поля γ_i.

(3) Вычисляются координаты y_i по формуле (3.1). При такой замене точка x_0 переходит в начало координат.

(4) Выполняется переход в привилегированные координаты с помощью замены:

$$\begin{aligned} z_i &= y_i, \qquad i = 1, \dots, 3, \\ z_4 &= y_4 - \frac{1}{2}(y_1^2\sigma_1 + 2y_1y_2\sigma_2 + y_2^2\sigma_3), \\ z_5 &= y_5 - \frac{1}{2}(y_1^2\sigma_4 + 2y_1y_2\sigma_5 + y_2^2\sigma_6), \end{aligned}$$

где

$$\begin{aligned} &\sigma_1 = X_1(X_1(y_4))(x_0), \quad \sigma_2 = X_1(X_2(y_4))(x_0), \quad \sigma_3 = X_2(X_2(y_4))(x_0), \\ &\sigma_4 = X_1(X_1(y_5))(x_0), \quad \sigma_5 = X_1(X_2(y_5))(x_0), \quad \sigma_6 = X_2(X_2(y_6))(x_0). \end{aligned}$$

(5) С использованием разложения векторных полей в ряд Маклорена строится нильпотентная аппроксимация

$$\dot{z} = \sum_{j=1}^{2} \hat{X}_j(z) u_j, \tag{3.2}$$

где

$$\hat{X}_j(z) = X_j^1(0)\frac{\partial}{\partial z_1} + X_j^2(0)\frac{\partial}{\partial z_2} + \left(\sum_{i=1}^{2}\frac{\partial X_j^3(z)}{\partial z_i}|_{z=0}\ z_i\right)\frac{\partial}{\partial z_3} +$$
$$+\left(\frac{\partial X_j^4(z)}{\partial z_3}|_{z=0}\ z_3 + \frac{\partial^2(X_i^4(z))}{\partial z_1^2}|_{z=0}\ z_1^2 + \frac{\partial^2(X_i^4(z))}{\partial z_2^2}|_{z=0}\ z_2^2 + \right.$$
$$\left. + \frac{\partial^2(X_i^4(z))}{\partial z_1 \partial z_2}|_{z=0}\ z_1 z_2\right)\frac{\partial}{\partial z_4} +$$
$$+\left(\frac{\partial X_j^5(z)}{\partial z_3}|_{z=0}\ z_3 + \frac{\partial^2(X_i^5(z))}{\partial z_1^2}|_{z=0}\ z_1^2 + \frac{\partial^2(X_i^5(z))}{\partial z_2^2}|_{z=0}\ z_2^2 + \right.$$
$$\left. + \frac{\partial^2(X_i^5(z))}{\partial z_1 \partial z_2}|_{z=0}\ z_1 z_2\right)\frac{\partial}{\partial z_5},\ \text{где } j = 1, 2.$$

(6) Вводится система координат, в которых система (3.2) имеет канонический вид:

$$\begin{cases} \dot{x} &= u_1, \\ \dot{y} &= u_2, \\ \dot{z} &= \frac{1}{2}(xu_2 - yu_1), \\ \dot{v} &= \frac{1}{2}(x^2 + y^2)u_2, \\ \dot{w} &= -\frac{1}{2}(x^2 + y^2)u_1. \end{cases} \tag{3.3}$$

Каноническая система (3.3) является нильпотентной и имеет вектор роста $(2, 3, 5)$.

Известно [15], что любые две пятимерные нильпотентные системы с вектором роста $(2, 3, 5)$ диффеоморфны. Замена переменных, переводящая одну такую систему в другую, строится следующим образом.

Пусть X_1, X_2 — векторные поля первой, а Y_1, Y_2 — векторные поля второй нильпотентной системы с вектором роста $(2, 3, 5)$. Построим диффеоморфизм, переводящий поля X_i в окрестности точки x_0 в поля Y_i в окрестности y_0:

$$\Phi : O(x_0) \to O(y_0), \qquad \Phi_*(X_i) = Y_i.$$

Определим отображения F и G как композицию потоков векторных полей X_i и Y_i соответственно за время t_i:

$$F(t_1,\dots,t_5) = e^{t_5 X_5}\circ\cdots\circ e^{t_1 X_1}(x_0),$$
$$G(t_1,\dots,t_5) = e^{t_5 Y_5}\circ\cdots\circ e^{t_1 Y_1}(y_0).$$

Отображения F, G задают канонические координаты второго рода на нильпотентной группе Ли $\mathbb{R}^5$, поэтому являются диффеоморфизмами (см., например, Теорему 3.18.11 [16]). Тогда искомый диффеоморфизм имеет вид

$$\Phi = G\circ F^{-1}.$$

Итак, мы имеем способ задания диффеоморфизма, переводящего поля исходной системы X_i из окрестности точки x_0 в поля системы Y_i в окрестности начала координат, нильпотентная аппроксимация которой является канонической системой:

$$\tau = \Phi\circ\lambda\circ\alpha. \tag{3.4}$$

4. Алгоритм приближенного решения задачи управления

Предположим, что имеется способ точного решения задачи управления для канонической нильпотентной системы (3.3), а потому и для любой нильпотентной аппроксимации (3.2). Далее в разделах 5, 6 описаны методы вычисления решения этой задачи в классах оптимальных и кусочно-постоянных управлений соответственно.

Опишем итерационный алгоритм приближенного решения задачи управления для системы

$$\dot{x} = u_1 X_1(x) + u_1 X_2(x), \qquad x\in\mathbb{R}^5, \quad (u_1,u_2)\in\mathbb{R}^5, \tag{4.1}$$

с начальной точкой p и конечной точкой q, для достаточно близких точек $p,q\in\mathbb{R}^5$, с заданной погрешностью $\varepsilon>0$, за время $T>0$.

(1) По формуле (3.4) вычисляется замена переменных τ, переводящая координаты x исходной системы (4.1) в координаты z канонической нильпотентной системы (3.3).
(2) Вводится обозначение $p^1 = p$.
(3) Вычисляется управление $u^1(t)$, $t\in[0,T]$, переводящее каноническую нильпотентную систему(3.3) из точки $z^1 = \tau(p^1)$ из точки в точку $0 = \tau(q)$.
(4) Вычисляется траектория $x^1(t)$, $t\in[0,T]$, исходной системы (4.1), соответствующая управлению $u^1(t)$, с начальной точкой $x(0) = p^1$.

(5) Если $\|x^1(T) - q\| < \varepsilon$, то задача решена. Иначе принимается $p^2 = x^1(T)$, выполняется переход в пункт (3), и процесс повторяется до достижения условия $\|x^N(T) - q\| < \varepsilon$ на некотором шаге N.

(6) Вычисленные управления $u^1(t), \dots, u^N(t)$ объединяются (с помощью стандартной процедуры конкатенации) в одно управление $u(t)$, $t \in [0, T]$, которое и является решением поставленной задачи.

5. Точное решение нильпотентной задачи в классе оптимальных управлений

В работах [11–14] исследуется задача оптимального управления для канонической нильпотентной системы (3.3) с краевыми условиями

$$q(0) = 0, \qquad q(T) = q_1, \qquad q = (x, y, z, v, w) \in \mathbb{R}^5, \tag{5.1}$$

и критерием оптимальности

$$l = \int_0^T \sqrt{u_1^2(t) + u_2^2(t)}\, dt \to \min. \tag{5.2}$$

Экстремальные траектории в задаче (3.3), (5.1), (5.2) параметризованы функциями Якоби. Семейство экстремальных траекторий описывается экспоненциальным отображением

$$\begin{aligned} &\operatorname{Exp} : N \to M = \mathbb{R}^5, \\ &N = C \times \mathbb{R}_+, \quad C = \{\lambda = (h_1, h_2, h_3, h_4, h_5) \in \mathbb{R}^5 \mid h_1^2 + h_2^2 = 1\}, \\ &\operatorname{Exp}(\lambda, t) = q(t). \end{aligned}$$

В работах [12–14] получена верхняя оценка времени разреза, т.е. времени, когда экстремальные траектории теряют оптимальность:

$$t_{\text{cut}}(\lambda) \leq \mathbf{t}(\lambda), \qquad \lambda \in C, \tag{5.3}$$

где $\mathbf{t} : C \to (0, +\infty]$ есть некоторая функция, явно описанная в [14]. Из существования оптимальных управлений, принципа максимума Понтрягина, и оценки (5.3) следует, что для исследования оптимальных решений в задаче (3.3), (5.1), (5.2) следует рассмотреть ограничение экспоненциального отображения

$$\begin{aligned} &\operatorname{Exp} : \widehat{N} \to \widehat{M} = M \setminus \{q_0\}, \\ &\widehat{N} = \{(\lambda, t) \in N \mid t \leq \mathbf{t}(\lambda)\}, \end{aligned} \tag{5.4}$$

причем ограничение (5.4) является сюръективным.

Из результатов работ [12–14] следует, что можно выделить открытые всюду плотные подмножества $N' \subset \widehat{N}$, $M' \subset \widehat{M}$, сужение на которые

$$\mathrm{Exp} : N' \to M', \tag{5.5}$$

$$\mathrm{Exp}(u, v, k, \alpha, \beta) = (x, y, z, v, w), \tag{5.6}$$

будет диффеоморфизмом.

Подмножества N', M' описываются следующим образом.

$$M' = \{(x, y, z, v, w) \in \mathbb{R}^5 \mid z \neq 0, \quad V \neq 0\},$$
$$V(x, y, z, v, w) = xv + yw - z\frac{x^2 + y^2}{2},$$

$$N' = \cup_{i=1}^{4} L_i,$$
$$L_i = L_i^1 \cup L_i^2, \qquad i = 1, \ldots, 4,$$

$$L_1^1 = \{(u, v, k, \alpha, \beta) \mid u \in (0, 3\pi/2),\ g_z(u) > 0,\ g_V^1(u) < 0,\ v \in (0, \pi/2)\},$$
$$L_1^2 = \{(u, v, k, \alpha, \beta) \mid u \in (0, \pi),\ g_V^2(u) < 0,\ v \in (0, \pi/2)\},$$
$$L_2^1 = \{(u, v, k, \alpha, \beta) \mid u \in (0, 3\pi/2),\ g_z(u) > 0,\ g_V^1(u) < 0,\ v \in (\pi/2, \pi)\},$$
$$L_2^2 = \{(u, v, k, \alpha, \beta) \mid u \in (0, \pi),\ g_V^2(u) < 0,\ v \in (-\pi/2, 0)\},$$
$$L_3^1 = \{(u, v, k, \alpha, \beta) \mid u \in (0, 3\pi/2),\ g_z(u) > 0,\ g_V^1(u) < 0,$$
$$v \in (\pi, 3\pi/2)\},$$
$$L_3^2 = \{(u, v, k, \alpha, \beta) \mid u \in (0, \pi),\ g_V^2(u) < 0,\ v \in (0, \pi/2)\},$$
$$L_4^1 = \{(u, v, k, \alpha, \beta) \mid u \in (0, 3\pi/2),\ g_z(u) > 0,\ g_V^1(u) < 0,$$
$$v \in (3\pi/2, 2\pi)\},$$
$$L_4^2 = \{(u, v, k, \alpha, \beta) \mid u \in (0, \pi),\ g_V^2(u) < 0,\ v \in (-\pi/2, 0)\},$$
$$\text{для всех } L_i^j : \qquad k \in (0, 1),\ \alpha > 0,\ \beta \in [0, 2\pi),$$

где

$$g_z(u, k) = f_z(p, k),\ p = \mathrm{am}(u, k),$$
$$f_z(p, k) = \mathrm{sn}\, p\, \mathrm{dn}\, p - (2\mathrm{E}(p) - p)\, \mathrm{cn}\, p,$$
$$g_V^1(u, k) = f_V^1(p, k),\ p = \mathrm{am}(u, k),$$

$$
\begin{aligned}
f_V^1(p,k) &= \frac{4}{3}\,\mathrm{sn}\,p\,\mathrm{dn}\,p\,(-p-2(1-2k^2+6k^2\,\mathrm{cn}^2\,p)(2\mathrm{E}(p)-p)+\\
&\quad +(2\mathrm{E}(p)-p)^3+8k^2\,\mathrm{cn}\,p\,\mathrm{sn}\,p\,\mathrm{dn}\,p)+\\
&\quad +4\,\mathrm{cn}\,p\,(1-2k^2\,\mathrm{sn}^2\,p)(2\mathrm{E}(p)-p)^2,\\
g_V^2(u,k) &= f_V^2(p,k),\ p=\mathrm{am}(u,k),\\
f_V^2(p,k) &= \frac{4}{3}\{3\,\mathrm{dn}\,p\,(2\mathrm{E}(p)-(2-k^2)p)^2+\mathrm{cn}\,p\,[8\mathrm{E}(p)^3-\\
&\quad -4\mathrm{E}(p)(4+k^2)-12\mathrm{E}(p)^2(2-k^2)p+6\mathrm{E}(p)(2-k^2)^2p^2+\\
&\quad +p(16-4k^2-3k^4-(2-k^2)^3p^2)]\,\mathrm{sn}\,p-\\
&\quad -2\,\mathrm{dn}\,p\,(-4k^2+3(2\mathrm{E}(p)-(2-k^2)p)^2)\,\mathrm{sn}^2\,p+\\
&\quad +12k^2\mathrm{cn}\,p(2\mathrm{E}(p)-(2-k^2)p)\mathrm{sn}^3\,p-8k^2\,\mathrm{sn}^4\,p\,\mathrm{dn}\,p\},
\end{aligned}
$$

и

$$
M' = \cup_{i=1}^4 M_i,
$$

$$
\begin{aligned}
M_1 &= \{(x,y,z,v,w)\in\mathbb{R}^5 \mid z>0, \quad V<0\},\\
M_2 &= \{(x,y,z,v,w)\in\mathbb{R}^5 \mid z<0, \quad V<0\},\\
M_3 &= \{(x,y,z,v,w)\in\mathbb{R}^5 \mid z<0, \quad V>0\},\\
M_4 &= \{(x,y,z,v,w)\in\mathbb{R}^5 \mid z>0, \quad V>0\}.
\end{aligned}
$$

Более того, каждое из отображений

$$
\mathrm{Exp}(L_i)\subset M_i, \qquad i=1,\dots,4. \tag{5.7}
$$

является диффеоморфизмом (в частности, биекцией).

Таким образом, для нахождения оптимального решения задачи (3.3), (5.1), (5.2) для открытого всюду плотного множества терминальных точек $q_1\in M'$ достаточно научиться обращать отображения (5.7), т.е. решать системы из пяти уравнений вида (5.6) в функциях Якоби от пяти неизвестных (u,v,k,α,β).

Подобная проблема была успешно решена для задачи Эйлера об эластиках [17], однако там возникают аналогичные системы из трех уравнений с тремя неизвестными. Для понижения размерности систем (5.6) для (3.3), (5.1), (5.2) будет использована двухпараметрическая группа симметрий этой задачи (вращения и дилатации), описанная в работе [15]. Факторизуя системы уравнений (5.6) по действию

этой двумерной группы, получаем системы трех уравнений с тремя неизвестными вида

$$P(u,v,k) = P_1, \tag{5.8}$$
$$Q(u,v,k) = Q_1, \tag{5.9}$$
$$R(u,v,k) = R_1, \tag{5.10}$$

где

$$P = z/(2r^2), \qquad r^2 = x^2 + y^2,$$
$$Q = (xv + yw - zr^2/2)/r^4,$$
$$R = (xw - yv)/r^2,$$

суть координаты в факторе пространства M по действию двумерной группы симметрий.

Система уравнений (5.8)–(5.10) аналогична системе, возникающей в задаче Эйлера об эластиках [17]. В системе Mathematica [18] разрабатывается программа решения этой системы.

6. Точное решение нильпотентной задачи в классе кусочно-постоянных управлений

Другой естественный класс управлений, который можно использовать для точного решения нильпотентной задачи (3.3), — кусочно постоянные управления. Из подсчета параметров ясно, что достаточно рассматривать управления тремя переключениями:

$$u_i = \begin{cases} \alpha_i, \text{ при } t \in [0, \frac{T}{4}], \\ \beta_i, \text{ при } t \in (\frac{T}{4}, \frac{T}{2}], \\ \gamma_i, \text{ при } t \in (\frac{T}{2}, \frac{3T}{4}], \\ \delta_i, \text{ при } t \in (\frac{3T}{4}, T], \end{cases}$$

где $i = 1, 2$.

Для определения коэффициентов управления

$$\alpha_i, \beta_i, \gamma_i, \delta_i$$

необходимо решить систему из пяти уравнений с восемью неизвестными:

$$\begin{cases} x(\alpha_i, \dots, \delta_i) = 0, \\ y(\alpha_i, \dots, \delta_i) = 0, \\ z(\alpha_i, \dots, \delta_i) = 0, \\ v(\alpha_i, \dots, \delta_i) = 0, \\ w(\alpha_i, \dots, \delta_i) = 0, \end{cases}$$

имеющую трехпараметрическое семейство решений. Для любого начального состояния x_0 существует способ зафиксировать свободные параметры так, чтобы получалось решение без особенностей.

В системе Mathematica [18] написана программа, определяющая коэффициенты управления из условия минимальности максимума модуля параметров $\alpha_i, \dots, \delta_i$. Таким образом получено точное решение задачи управления для канонической системы, которое можно использовать для приближенного решения исходной системы.

Эти управления были применены для задач управления качением шара и движением машины с прицепами (примеры 1, 2). Соответствующая программа в системе Mathematica продемонстрировала сходимость алгоритма при достаточно малых расстояниях между начальной и конечной точками. Графики координат решений этих задач (в отклонениях от целевой точки) приведены на рис. 1, 2.

Фильм, визуализирующий качение шара, выложен в файле `ftp://univ.u.pereslavl.ru/upload/masht/SphereMov.avi`.

В большой серии проведенных численных экспериментов алгоритм приближенного решения задачи управления для обеих модельных систем («шар на плоскости» и «машина с прицепами») демонстрирует сходимость при достаточно малых расстояниях между исходной и целевой точками при использовании кусочно-постоянных управлений. Заметим, что теоретически такая сходимость алгоритма гарантирована, если для используемого класса управлений $\mathcal{U}$ выполнено следующее топологическое свойство, которое естественно назвать локальной управляемостью системы (1.1) в классе управлений $\mathcal{U}$:

> для любой точки $q_1 \in M$ и для любой ее окрестности $O_1 \subset M$ существует такая окрестность $O_2 \subset O_1$ точки q_1, что для любой точки $q_0 \in O_2$ существует управление $u(\cdot) \in \mathcal{U}$, переводящее точку q_0 в q_1, причем соответствующая траектория $x(\cdot)$ системы (1.1) содержится в окрестности O_1.

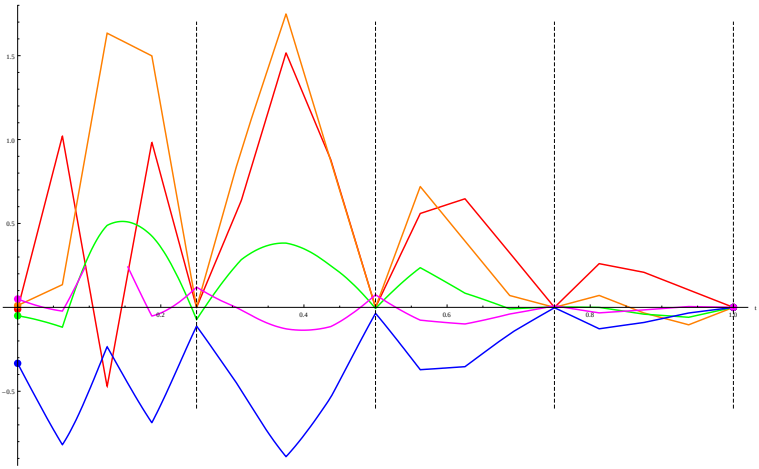


Рис. 1. Траектории системы «шар на плоскости» (в отклонениях)

Из непрерывности субриманова расстояния (1.5) следует, что класс управлений, оптимальных в смысле минимума функционала субримановой длины (1.4), этому свойству удовлетворяет. Для класса кусочно-постоянных управлений, описанных в пункте 6, этот вопрос подлежит изучению.

7. Заключение

В работе рассмотрен конструктивный метод приближенного решения задачи управления на основе нильпотентной аппроксимации. Детально рассмотрен случай пятимерных систем с двумерным управлением, в классах оптимальных и кусочно-постоянных управлений. Эффективность алгоритма и компьютерной программы продемонстрирована на системах, описывающих качение шара по плоскости, и движение машины с двумя прицепами.

Список литературы

[1] Аграчев А. А., Сачков Ю. Л. Геометрическая теория управления. — М.: Физматлит, 2005. — 392 с. ↑1, 1, 1

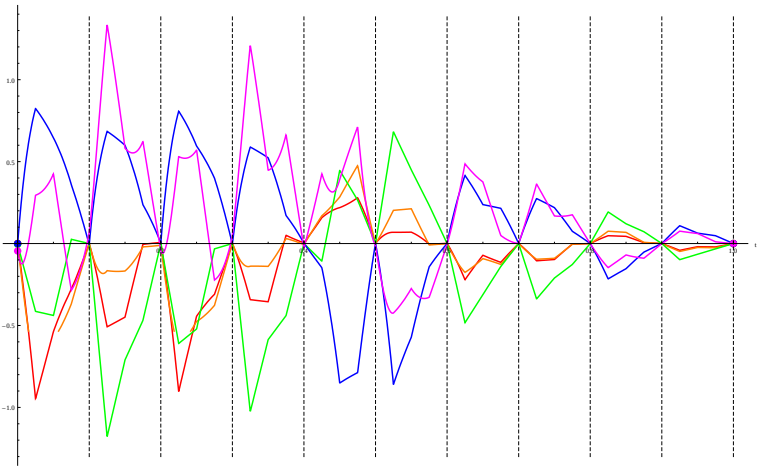


Рис. 2. Траектории системы «машина с прицепами» (в отклонениях)

[2] Jurdjevic V. Geometric control theory. — Cambridge: University Press, 1997. ↑1

[3] Laumond J. P. // Lecture Notes in Control and Information Science, № 229: Springer, 1998. — 343 с. ↑1

[4] Murray R.M., Sastry S.S. *Steering controllable systems* // Proc. 29th IEEE Conf. Dec. and Control. — Honolulu, Hawaii, 1990. ↑1

[5] Fliess M., Levine J., Martin P., Rouchon P. *On differential flat nonlinear systems* // Proc. IFAC NOLCOS Symposium. — Bordeaux, France, 1992, с. 408–412. ↑1

[6] Аграчев А.А., Сарычев А.В. *Фильтрация алгебры Ли векторных полей и нильпотентная аппроксимация управляемых систем* // ДАН СССР. — **295**, 1987, с. 777–781. ↑1, 2

[7] Bellaiche A. *The tangent space in sub-Riemannian geometry.* // Sub-Riemannian Geometry, A. Bellaiche and J. J. Risler, Eds. — Basel, Swizerland: Birkhäuser, 1996, с. 1–78. ↑3

[8] Hermes H. *Nilpotent and high-order approximations of vector fields systems* // SIAM Review. — **33**, 1991, с. 238–264. ↑2

[9] Laferriere G., Sussmann H.J. *A differential geometric approach to motion planning* // Nonholonomic Motion Planning, Zexiang Li and J.F. Canny Eds. — Basel, Swizerland: Kluwer, 1992. ↑

[10] Vendittelli M., Oriolo G., Jean F., Laumond J.-P. *Nonhomogeneous nilpotent approximations for nonholonimic systems with singularities* // IEEE Trans. Automat. Control. — **49**, 2004, с. 261–266. ↑1, 3

[11] Сачков Ю. Л. *Экспоненциальное отображение в обобщенной задаче Дидоны* // Мат. Сборник. — Т. **194**, 2003, с. 63–90. ↑1, 5

[12] Сачков Ю. Л. *Дискретные симметрии в обобщенной задаче Дидоны* // Мат. Сборник. — Т. **197,2**, 2006, с. 95–116. ↑5, 5

[13] Сачков Ю. Л. *Множество Максвелла в обобщенной задаче Дидоны* // Мат. Сборник. — Т. **197,4**, 2006, с. 123–150. ↑

[14] Сачков Ю. Л. *Полное описание стратов Максвелла в обобщенной задаче Дидоны* // Мат. Сборник. — Т. **197,6**, 2006, с. 111–160. ↑1, 5, 5, 5, 5

[15] Sachkov Yu.L. *Symmetries of Flat Rank Two Distributions and Sub-Riemannian Structures* // Transactions of the American Mathematical Society. — Т. **356**, 2004, с. 457–494. ↑3.2, 5

[16] Varadarajan V.S. Lie groups, Lie algebras, and their representations. — New York: Springer-Verlag, 1984. ↑3.2

[17] Ардентов А.А., Сачков Ю. Л. *Решение задачи Эйлера об эластиках* // Автоматика и Телемеханика, 2009, в печати. ↑5, 5

[18] Wolfram S. Mathematica: a system for doing mathematics by computer. — MA: Addison-Wesley, Reading, 1991. ↑5, 6

Исследовательский центр процессов управления ИПС РАН

Yu. L. Sachkov, A. A. Ardentov, A. P. Mashtakov. *Constructive solution to control problem via nilpotent approximation method* // Proceedings of Program Systems institute scientific conference “Program systems: Theory and applications”. — Pereslavl-Zalesskij, v. **1**, 2009. — p. 5–23. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. Nilpotent approximation method for nonlinear systems, linear in controls, is described. An algorithm of constructive solving the control problem for such systems on the basis of nilpotent approximations is presented. A realization of the algorithm in classes of optimal and piecewise constant control is considered.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 517.977:004.415.2

А. О. Блинов, В. И. Гурман, Е. А. Трушкова, В. П. Фраленко

Программный комплекс улучшения и оптимизации законов управления

Аннотация. Разработан программный комплекс улучшения и оптимизации управления для приложений в различных областях. В программном комплексе успешно реализованы на кластерном вычислительном устройстве алгоритмы аппроксимации, оптимизации и улучшения приближенно оптимального управления для различных динамических процессов с управлением. Исследование возможностей и эффективности комплекса проведено на ряде модельных и прикладных задач, включая многокомпонентную модель региона, модель движения вертолета в нештатной ситуации, модель спуска космического челнока в атмосфере с минимальным теплоподводом.

1. Введение

Работа посвящена разработке и реализации методов синтеза оптимальных целевых законов управления, что является кардинальной проблемой теории и практики управления. Ее решение связано с бесконечномерными обратными задачами, аппроксимируемыми при численной реализации многомерными задачами, требующими неограниченно растущих вычислительных ресурсов для приближения к точным решениям. Это приводит к неизбежному выводу о необходимости нового подхода — априорно приближенного, позволяющего связать точность приближения с размерностью аппроксимаций и, соответственно, с мощностью вычислительных средств, и реализовать такой подход на современных высокопроизводительных вычислительных системах параллельной архитектуры. Парадигма параллельных вычислений в высшей степени соответствует самой природе рассматриваемых задач, связанных с множественностью однотипных операций на верхнем уровне, таких как:

– решение обратной задачи через множество решений прямой (прямо или косвенно);

– формирование и численная реализация полей управлений (ситуационных управлений).

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований. Проект №08-01-00274-а.

В статье отражены результаты разработки, реализации и исследования экспериментального варианта комплекса улучшения и оптимизации законов управления для приложений в различных областях ПК ISCON (Improvement and Synthesis of Control) с распараллеливанием и переносом на кластерное вычислительное устройство (КВУ).

В следующем разделе 2 описываются главные компоненты комплекса: графический интерфейс, сервер управления, управляющие модули и набор исполняемых модулей. Разделы 3 и 4 содержат постановки задач, на которых проводятся вычислительные эксперименты в рамках тестирования ПК и основные рекомендации пользователям по работе в ПК.

2. Описание программного комплекса

Программный комплекс предназначен для моделирования сложных динамических процессов, а также решения оптимизационных задач и задач улучшения управления для различных прикладных областей на КВУ. Для этого в нем реализованы алгоритмы аппроксимации, оптимизации и улучшения приближенно оптимального управления. Главными компонентами комплекса являются графический интерфейс, сервер управления, управляющие модули и набор исполняемых модулей.

В графическом интерфейсе происходит ввод начальных данных, постановка задачи, выбор метода решения задачи, управление потоками данных, визуализация и сохранение результатов. Сервер управления участвует в обеспечении пользователям доступа к возможностям комплекса, принимает запросы на решение выбранных задач с выбранными пользователем настройками. Управляющие модули принимают полученную от сервера управления информацию и выполняют развертывание полигона для вычислений, запуская в дальнейшем либо локально, либо удаленно исполняемый модуль решаемой задачи. Так же они обеспечивают сбор выходных данных и их передачу обратно серверу управления или пользователю.

Основной идеей при разработке архитектуры системы было обеспечение гибкости и расширяемости. Выбранная модульная схема программного комплекса обеспечивает расширяемость и масштабируемость. Гетерогенность вычислительной среды поддержана включением в архитектуру ПК управляющих модулей, связывающих физически разделенные компоненты ПК. В зависимости от набора исполняемых модулей, ПК может использоваться при создании систем,

относящихся к различным задачам оптимизации и управления. Гибкость системы обеспечивается возможностью подключения доступных вычислительных устройств засчет конфигурирования управляющих модулей.

Из особенностей ПК можно выделить:

– Использование параллелизма на различных уровнях: параллельное выполнение решаемых задач, внутренний параллелизм модулей.

– Модули системы реализуются в виде исполняемых файлов и могут содержать как последовательную, так и параллельную реализацию алгоритма.

Управление ПК поддерживается графическим интерфейсом, интегрированным с сервером управления.

Область применения программного комплекса определяется реализованными методами (исполняемыми модулями), предназначенными для построения аналитического описания модели по имеющимся статистическим данным (алгоритм аппроксимации по методу наименьших квадратов (МНК)), улучшения приближенно оптимальной программы управления динамической системой, моделирования и исследования упругих систем. Указанные методы могут быть применены, например, к задачам управления экономическими системами, движением летательных аппаратов (вертолеты, космические челноки).

3. Описание задач, решаемых с помощью ПК

3.1. Программа аппроксимации моделей динамических систем

При работе с моделями реальных динамических систем

$$\dot{x}(t) = f\left(t, x(t), u(t)\right), \quad x \in \mathbb{R}^n, \quad u \in \mathbb{R}^p,$$

типичны ситуации, когда модель имеет структуру, к которой невозможно применить тот или иной метод исследования или алгоритм синтеза управления. В этих случаях, для того, чтобы упростить систему, предлагается применять алгоритм аппроксимации многомерных функций многих переменных $f(t, x, u)$ (при фиксированных моментах времени) по методу наименьших квадратов многомерными

полиномами [1]

$$\varphi(z) = \sum_{j=1}^{m} \psi_j g_j(z), \quad z = \left(z^1, \ldots, z^{n+p}\right) = (x, u) \in \mathbb{R}^{n+p},$$

где $\left\{g_j(z) = \Pi_{i=1}^{n+p} (z^i)^{k_i(j)}\right\}$ — некоторый набор заданных базисных функций, $k_i(j)$ — целые положительные числа, а $\{\psi_j\}$ — соответствующий набор коэффициентов, подлежащих определению. Решается следующая задача минимизации (МНК):

$$S = \sum_{i=1}^{\beta} \left(\varphi(z_i) - f(t, x_i, u_i)\right)^2 \to \min_{\{\psi_i\}},$$

где β — количество узлов аппроксимации. Эта задача сводится к решению системы линейных алгебраических уравнений с постоянными коэффициентами.

Для этого надо:

1) сформировать с помощью узлов аппроксимации и базисных функций приближающего полинома матрицу и столбец свободных членов для системы линейных алгебраических уравнений,

2) решить полученную систему.

Реализована параллельность указанного метода в первой части. Область формирования исходных данных разбивается на части, в каждой из которых строится матрица и столбец свободных членов для системы уравнений с помощью исходных значений в узлах текущей части. Общая матрица получается в этом случае как сумма всех построенных «частичных» матриц (это же справедливо и для свободных членов).

Алгоритм реализован в Т-системе (на языке программирования Т++) на КВУ семейства «СКИФ» [1].

3.1.1. *Входные данные задачи*

Входными данными задачи являются:

1. размерность фазового пространства n (целое число большее нуля);

2. размерность пространства управлений p (целое число большее нуля);

3. правая часть управляемой системы $f(t, x, u)$ — n-мерная функция от $(n + p)$ переменных (при фиксированном значении t);

4. нижние ограничения на фазовую траекторию и управление ($(n+p)$–мерный действительный вектор);

5. верхние ограничения на фазовую траекторию и управление ($(n+p)$–мерный действительный вектор);

6. базисные функции (j векторов размера $(n+p)$, содержащих степени вхождения переменных z^i в базисную функцию g_i, т. е. вектора вида $(k_1(j), k_2(j), \ldots, k_{n+p}(j))$).

3.1.2. *Выходные данные задачи*

Выходными данными задачи являются коэффициенты аппроксимирующих полиномов (n действительных векторов размера j).

3.1.3. *Настройка ПК по решению описанной задачи*

Настройка ПК по решению описанной задачи произведена для правой части динамической системы вида:

$$f\left(x^1, x^2, x^3, x^4, u^1, u^2\right) = \left(f_1(x,u), f_2(x,u), f_3(x,u), f_4(x,u)\right)^T, \tag{1}$$

где

(2)

$$\begin{aligned}
&f_1(x,u) = x^1 - 0.000004413602941\sqrt{(x^1)^2 + (x^2)^2}x^1 - 0.098u^1,\\
&f_2(x,u) = x^2 - 0.000004413602941\sqrt{(x^1)^2 + (x^2)^2}x^2 +\\
&+0.001217794118(x^3)^2\Big(- 0.04504670322 + 0.002934957373x^3 -\\
&-0.00005375103213(x^3)^2 + x^1\left(0.005307639750 - 0.000291577926x^3 +\right.\\
&\left.+0.00000474579887(x^3)^2\right) + x^2\left(0.00586815863 - 0.000587450425x^3 +\right.\\
&\left.+0.00001158950057(x^3)^2\right) + u^1\left(0.07527467316 - 0.003228550181x^3 +\right.\\
&\left.+0.00004177445652(x^3)^2\right) + u^2\left(0.4564552109 - 0.007347112224x^3 +\right.\\
&\left.+0.0001468137088(x^3)^2\right)\Big) - 0.09799999999,\\
&f_3(x,u) = 1.001251011x^3 + 0.02526105823 - 0.00005189698900(x^3)^2 +\\
&+0.01x^1\left(0.5281399869 - 0.03512379548x^3 + 0.0006299973572(x^3)^2\right) +\\
&+0.01x^2\left(-1.228470885 + 0.07975873768x^3 - 0.001487652282(x^3)^2\right) +\\
&+0.01u^1\left(8.462949172 - 0.5230654017x^3 + 0.009933002308(x^3)^2\right)) +\\
&+0.01u^2\left(75.61380577 - 5.458013542x^3 + 0.08086676638(x^3)^2\right) -\\
&-\frac{0.071153846151}{x^3},\\
&f_4(x,u) = x^4 + 0.01x^2.
\end{aligned}$$

Для проведения аппроксимации указанной функции $f(x,u)$, например, линейной функцией

$$\varphi(x,u) = \psi_1 + \psi_2 x^1 + \psi_3 x^2 + \psi_4 x^3 + \psi_5 x^4 + \psi_6 u^1 + \psi_7 u^2$$

в области изменения переменных

$$0.278 \le x^1 \le -3.2, \quad -3.2 \le x^2 \le 0, \quad 24.6 \le x^3 \le 30.8, \quad -6 \le x^4 \le 0,$$
$$-0.349 \le u^1 \le 0.349, \quad -0.349 \le u^2 \le 0.349$$

входные данные следует задать в виде:

1. размерность фазового пространства $n = 4$;

2. размерность пространства управлений $p = 2$;

3. правая часть управляемой системы $f(x, u)$ (см. выше);

4. нижние ограничения на фазовую траекторию и управление (6-мерный действительный вектор): (0.278, -3.2, 24.6, -6, -0.349, 0.07853);

5. верхние ограничения на фазовую траекторию и управление (6-мерный действительный вектор): (7.5, 0, 30.8, 0, 0.349, 0.349);

6. базисные функции (7 векторов размерности 6, содержащих степени вхождения переменных в текущую базисную функцию), т. е. вектора:

$$(0,0,0,0,0,0),$$
$$(1,0,0,0,0,0),$$
$$(0,1,0,0,0,0),$$
$$(0,0,1,0,0,0),$$
$$(0,0,0,1,0,0),$$
$$(0,0,0,0,1,0),$$
$$(0,0,0,0,0,1).$$

Для оценки эффективности распараллеливания программы проведен запуск ПК на различном числе узлов и замер времени работы в каждом случае. Полученные данные представлены в таблице 1.

Таблица 1. Анализ эффективности параллельной версии программы улучшения управления.

Число узлов: n	Время работы программы: t_n	Ускорение: t_1/t_n	Число узлов: n	Время работы программы: t_n	Ускорение: t_1/t_n
1	3338.348	1	9	631.214	5.289
2	1779.791	1.876	10	596.175	5.600
3	1237.142	2.698	11	588.017	5.677
4	880.248	3.793	12	596.195	5.599
5	729.924	4.574	13	589.926	5.659
6	631.720	5.285	14	586.519	5.692
7	632.003	5.282	15	597.649	5.586
8	586.202	5.695	16	579.739	5.758

Эти данные позволяют сделать вывод об эффективном распараллеливании указанного класса алгоритмов лишь для небольшого числа узлов (1–8).

3.2. Программа улучшения управления

Предполагается, что модель движения в общем случае представляет собой дискретную управляемую систему, терминальный функционал качества, ограничения на управления типа неравенств, фазовые ограничения типа неравенств (различные внутри и на правом конце заданного фиксированного промежутка времени):

$$
\begin{array}{l}
x(t+1) = f\left(t, x(t), u(t)\right), \quad t \in T = \{t_I, t_I + 1, \ldots, t_F\}, \\
x(t_I) = x_I, \\
u(t) \in D_u = \left\{u : T\backslash\{t_F\} \to \mathbb{R}^p \middle| u_l \le u(t) \le u_u, t \in T\backslash\{t_F\}\right\}, \\
x_l \le x(t) \le x_u, t \in T\backslash\{t_F\}, \quad x_{lF} \le x(t_F) \le x_{uF}, \\
I = F_0\left(x(t_F)\right) \to \min, \quad u_l, u_u \in \mathbb{R}^p, x_I, x_l, x_u, x_{lF}, x_{uF} \in \mathbb{R}^n.
\end{array}
$$

Производится замена этой задачи оштрафованной, т. е. задачей без фазовых ограничений, с помощью введенных функций штрафа типа срезок:

$$
\begin{array}{l}
x(t+1) = f\left(t, x(t), u(t)\right), \quad t \in T, \quad x(t_I) = x_I, \\
z(t+1) = z(t) + t_F^{-1}\delta\left(x(t)\right), \quad z(t_I) = 0, \\
F\left(x(t_F), z(t_F)\right) = \beta_0 F_0\left(x(t_F)\right) + \beta_1^T z(t_F) + \beta_2^T \delta_F\left(x(t)\right) \to \min,
\end{array}
$$

где

$$
\beta_0 \in \mathbb{R}, \beta_1, \beta_2 \in \mathbb{R}^n, \quad \delta^i(x) = -\min\{0, x^i - x_l^i\} + \max\{0, x^i - x_u^i\},
$$
$$
\delta_F^i(x) = -\min\{0, x^i - x_{lF}^i\} + \max\{0, x^i - x_{uF}^i\}, \quad i = 1, \ldots, n.
$$

Задача улучшения ставится следующим образом: пусть известен допустимый элемент $m^I = \left(u^I(t), x^I(t)\right)$, требуется найти допустимый элемент $m^{II} = \left(u^{II}(t), x^{II}(t)\right)$, такой, что $F\left(x^{II}(t_F), z^{II}(t_F)\right) < F\left(x^I(t_F), z^I(t_F)\right)$.

Итерационное улучшение основано на линейно-квадратических аппроксимациях соотношений Беллмана в среднем в окрестности текущего приближения полиномами второго порядка. Предусмотрены регуляторы, настраиваемые так, чтобы каждая итерация была наиболее эффективной.

Общие конструкции метода улучшения управления приведены в [2, 3], где на основе принципа оптимальности Кротова элемент m^{II}

ищется путем аппроксимации решения следующей задачи:

$$\begin{gathered} y(t+1) = g\left(t, y(t), v(t)\right), \quad t \in T, \quad y(t_I) = 0, \\ s(t+1) = s(t) + t_F^{-1}\delta\left(y(t) + x^I(t)\right) - t_F^{-1}\delta\left(x^I(t)\right), \quad s(t_I) = 0, \\ y^0(t+1) = y^0(t) + \tfrac{1}{2}v^T(t)v(t), \quad y^0(t_I) = 0, \\ G_\alpha\left(y^0(t_F), y(t_F), s(t_F)\right) = \alpha y^0(t_F) + \\ +(1-\alpha)F\left(y(t_F) + x^I(t_F), s(t_F) + z^I(t_F)\right) \to \min, \end{gathered}$$

где α — некоторое число из отрезка [0,1] (регулятор метода), $y = x - x^I$, $s = z - z^I$, $v = u - u^I$, $g(t, y, v) = f(t, y + x^I, v + u^I) - f(t, x^I, u^I)$.

Будем искать функцию Кротова в виде

$$\varphi(t, y^0, y, s) = w(t) + \psi^0(t)y^0 + \psi^T(t)y + \gamma^T(t)s,$$

где значения $w(t)$, ψ^0, ψ, γ находятся из следующих приближенных соотношений Кротова-Беллмана:

$$\varphi(t_F, y^0, y, s) \approx -G_\alpha(y^0, y, s),$$

$$\begin{aligned} \varphi(t, y^0, y, s) \approx \max_v \varphi((t+1, y^0 + \frac{1}{2}v^T v, g(t, y, v), \\ s + t_F^{-1}\delta(y + x^I) - t_F^{-1}\delta(x^I)), \quad t = t_F - 1, \ldots, t_I. \end{aligned}$$

При этом управление (в форме синтеза), на котором достигается максимум, обозначим через $\hat{v}(t, y)$.

Опишем одну итерацию алгоритма метода улучшения. По данному начальному приближению $m^I = \left(u^I(t), x^I(t)\right)$ выбираем весовые коэффициенты функционала $F\left(x(t_F), z(t_F)\right)$ из следующих условий:

$$\begin{array}{ll} \beta_0 = 1, \beta = \left(\beta_1^1, \ldots, \beta_1^n, \beta_2^1, \ldots, \beta_2^n\right)^T = 0, & \text{если} \quad J_0 = \{1, \ldots, 2n\}, \\ \beta_0 = 0, \beta^j = 0, j \in J_0, \beta^j = \frac{P}{Sh^j}, j \in J, & \text{если} \quad J_0 \neq \{1, \ldots, 2n\}, \end{array}$$

где

$$\begin{gathered} h = \left(z^T(t_F), \delta_F^T(t_F)\right)^T, \quad J_0 = \left\{j \in \{1, \ldots, 2n\} \middle| h^j \leq 0.001\right\}, \\ J = \{1, \ldots, 2n\} \backslash J_0, \quad P = \Pi_{j \in J} h^j, \quad S = \textstyle\sum_{j \in J} \frac{P}{h^j}. \end{gathered}$$

Фиксируем набор параметров метода. Разложив правые части соотношений Кротова–Беллмана при фиксированном $t \in T$ в ряд до членов второго порядка в окрестности нуля и заменив производные их разностными аналогами (шаги разностных схем – дополнительные параметры метода), получим управление

$$u^{II}(t, x) = \hat{v}(t, x - x^I(t)) + u^I(t), \quad t \in T \backslash \{t_f\},$$

и соответствующий элемент

$$m^{II} = \left(u^{II}(t) = u^{II}\left(t, x^{II}(t)\right), x^{II}(t)\right).$$

Если улучшение произошло, проводим следующую итерацию, выбрав в качестве начального элемента m^{II}. В противном случае берем другой набор параметров метода или останавливаем итерации.

Построенный алгоритм естественным образом ориентирован на параллельные вычисления. Алгоритм реализован в Т-системе (язык программирования Т++) на КВУ семейства «СКИФ» [4].

3.2.1. *Входные данные задачи*

Входными данными задачи являются:

1. размерность фазового пространства n (целое число большее нуля);

2. размерность пространства управлений p (целое число большее нуля);

3. начальный момент времени t_I (любое действительное число);

4. конечный момент времени t_F (действительное число большее t_I);

5. число отрезков разбиения временного интервала m (целое число большее нуля);

6. правая часть управляемой системы $f(t, x, u)$— n-мерная функция от $(n + p + 1)$ переменной;

7. минимизируемый функционал $F_0\left(t_F, x(t_F)\right)$— функционал от n переменных;

8. начальное значение фазовой траектории x_I (n-мерный действительный вектор);

9. нижние ограничения на фазовую траекторию внутри временного отрезка (n–мерный действительный вектор), вектор–индикатор наличия/отсутствия этих ограничений (n-мерный вектор, состоящий из нулей и единиц);

10. верхние ограничения на фазовую траекторию внутри временного отрезка (n-мерный действительный вектор, каждая компонента которого больше соответствующей компоненты вектора из 9.), вектор-индикатор наличия/отсутствия этих ограничений (n-мерный вектор, состоящий из нулей и единиц);

11. нижние ограничения на фазовую траекторию в момент времени t_F (n-мерный действительный вектор), вектор–индикатор наличия/отсутствия этих ограничений (n-мерный вектор, состоящий из нулей и единиц);

12. верхние ограничения на фазовую траекторию в момент времени t_F (n-мерный действительный вектор, каждая компонента которого больше соответствующей компоненты вектора из 11.), вектор-индикатор наличия/отсутствия этих ограничений (n-мерный вектор, состоящий из нулей и единиц);

13. нижние ограничения на управление (p-мерный действительный вектор);

14. верхние ограничения на управление (p-мерный действительный вектор, каждая компонента которого больше соответствующей компоненты из 13.);

15. минимальное время перекладки управления (p-мерный действительный вектор, каждая компонента которого положительна);

16. начальная программа управления ((pm)-мерный действительный вектор или имя текстового файла).

При этом данные 1–8 являются обязательными.

3.2.2. *Выходные данные задачи*

Вывод результатов работы программы осуществляется в текстовый файл. В нем указывается набор параметров; номер итерации, на которой достигнуто наибольшее улучшение; далее таблицей идут столбцы результатов (число строк равно числу временных моментов): временной момент, значения траектории в этот момент (покоординатно), значения управлений в этот момент (покоординатно), значения отклонений от допустимого множества (покоординатно); в заключение приводится достигнутое значение целевой функции. Предусмотрена также возможность вывода управления в форме синтеза ($u(t,x) = A(t)x + B(t)$): вывод двух текстовых файлов, один из которых содержит матрицу коэффициентов при переменной x (матрицу $A(t)$), другой — матрицу коэффициентов свободных членов (матрицу $B(t)$).

Формат файла выходных результатов позволяет производить построение графиков для наглядного представления произошедшего улучшения.

3.2.3. *Настройка ПК по решению описанной задачи*

Настройка ПК по решению описанной задачи произведена, в частности, для задачи улучшения начального приближенно оптимального

управления для нелинейной системы, полученной при аппроксимации модели движения вертолета в нештатной ситуации [3]:

$$x(t+1) = f\left(x(t), u(t)\right), \quad t \in \{0, 1, \ldots, t_F\}, \quad x \in \mathbb{R}^4, u \in \mathbb{R}^2,$$

где правая часть системы $f(x, u)$ определена согласно формулам 1 и 2.

Заданы начальные значения фазовых переменных, ограничения на фазовые переменные во время и в конце маневра, ограничения на управления:

$$\begin{array}{l} x(0) = (10, 0, 29.6, 0)^T, \\ u_l = (-0.348, 0.08)^T, \quad u_u = (0.348, 0.348)^T, \\ x_l = (0, -5, 24.6, -\infty)^T, \quad x_u = (+\infty, 0, 30.8, +\infty)^T, \\ x_{lF} = (0, -3.2, 24.6, -\infty)^T, \quad x_{uF} = (7.5, 0, 30.8, +\infty)^T. \end{array}$$

Требуется минимизировать конечную высоту $F_0\left(x(t_F)\right) = x^4(t_F)$, что равносильно максимизации нижней границы опасной зоны аварийной посадки.

Для проведения улучшения одного из вариантов начальной программы управления входные данные следует задать, например, в виде:

1. размерность фазового пространства $n = 4$;
2. размерность пространства управлений $p = 2$;
3. начальный момент времени $t_I = 0$;
4. конечный момент времени $t_F = 9.47$;
5. число отрезков разбиения временного интервала $m = 947$;
6. правая часть управляемой системы $f(x, u)$ (см. выше);
7. минимизируемый функционал $F_0\left(x(t_F)\right)$ (см. выше);
8. начальное значение фазовой траектории $x_I = (10, 0, 29.6, 0)$;
9. нижние ограничения на фазовую траекторию внутри временного отрезка $(0, -5, 24.6, 0)$, вектор-индикатор ограничений $(1, 1, 1, 0)$;
10. верхние ограничения на фазовую траекторию внутри временного отрезка $(7.5, 0, 30.8, 0)$, вектор-индикатор ограничений $(0, 1, 1, 0)$;
11. нижние ограничения на фазовую траекторию в момент времени t_F $(0, -3.2, 24.6, 0)$, вектор-индикатор ограничений $(1, 1, 1, 0)$;
12. верхние ограничения на фазовую траекторию в момент времени t_F $(7.5, 0, 30.8, 0)$, вектор-индикатор ограничений $(1, 1, 1, 0)$;
13. нижние ограничения на управление $(-0.348, 0.08)$;
14. верхние ограничения на управление $(0.348, 0.348)$;

15. минимальное время перекладки управления $(0.7, 0.35)$;

16. начальная программа управления берется из файла с именем upr_nach.txt.

Описанная программа улучшения управления для дискретных динамических систем была успешно применена к исследованию имитационной модели маневров безопасной нештатной посадки вертолета с определением границы безопасной зоны. Результаты позволили сделать вывод о повышении границы опасной зоны на 15% против начального приближения при сохранении качественного характера динамики управлений и состояния [3]. Результаты работы программного комплекса для входных числовых данных, описанных выше, представлены на рис. 1. Вычисления проводились для 256 различных наборов параметров метода улучшения. В результате работы программы удалось уменьшить значение целевого функционала, удовлетворив при этом всем ограничениям (см. рис. 1).

Для оценки эффективности распараллеливания программы проведен запуск ПК на различном числе узлов и замер времени работы в каждом случае. Полученные данные представлены в таблице 2.

Таблица 2. Анализ эффективности параллельной версии программы улучшения управления.

Число узлов: n	Время работы программы: t_n	Ускорение: t_1/t_n
1	1029.85	1
3	351.99	2.93
5	218.83	4.71
7	159.60	6.45
9	130.71	7.88
11	110.29	9.34
13	93.69	10.99
15	90.10	11.43

Эти данные позволяют сделать вывод об эффективном распараллеливании указанного класса алгоритмов.

4. Работа с программным комплексом

Графический интерфейс запускается из командной строки (файл gui.exe) или с помощью манипулятора типа «мышь». После этого пользователю становится доступна его базовая форма. Созданную задачу можно сразу же запустить на КВУ.

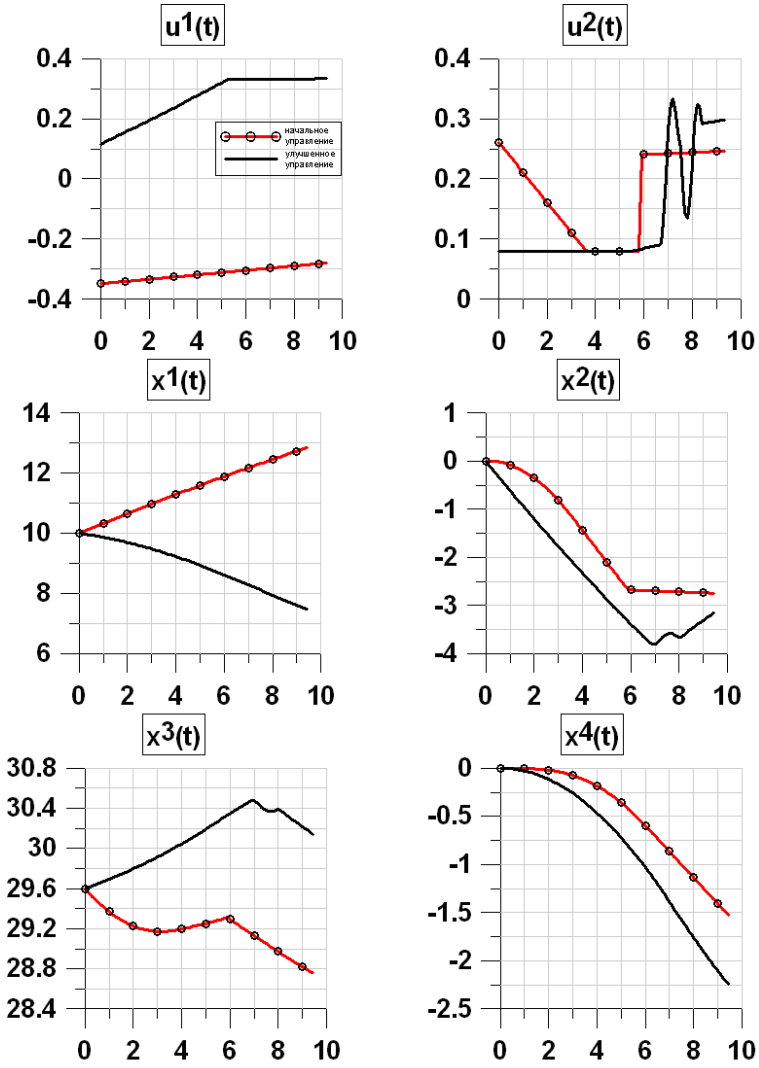


Рис. 1. Начальные и улучшенные значения управлений и соответствующих траекторий.

Входные данные представлены на вкладке «Параметры» области D и списком внешних файлов в форме «Отправить файлы» в *области E* (см. рис. 2).

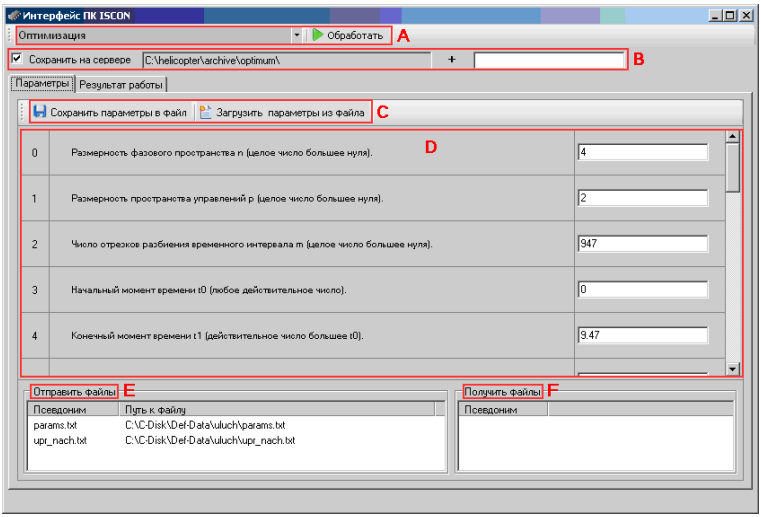


Рис. 2. Основная форма графического интерфейса.

На рис. 2 выделены шесть областей:

1. *область A*, в которой находится инструмент выбора решаемой задачи, кнопка запуска ее обработки;

2. *область B* с необходимыми настройками — включение/выключение архивации результатов обработки созданных задач на сервере с указанием поддиректории;

3. *область C* с кнопками сохранения параметров решаемой задачи в файл и загрузки из файла;

4. *область D*, содержащая необходимые параметры решаемой задачи с их расшифровкой;

5. *область E* с управляемой группой файлов, необходимых для решения задачи (загружаются с компьютера пользователя графического интерфейса на управляющий модуль);

6. *область F* с управляемой группой файлов, получаемых после решения задачи (загружаются на компьютер пользователя графического интерфеса от управляющего модуля).

При использовании графического интерфейса параметризация запуска вычислительных модулей решается автоматически согласно выбранным настройкам в *области D* и загружаемым файлам из *области E*. Другие настройки задаются при конфигурировании сервера управления и при создании управляющего модуля.

В случае успешного завершения вычислений появится окно с текстом «Работа над задачей успешно завершена». Если же во время работы счет задачи был остановлен пользователем, появится первое сообщение об остановке схемы обработки. Второе сообщение говорит о принудительном разрыве связи с сервером управления. Оно может появиться и в случае физического обрыва канала связи. Для контроля за активностью сервера управления можно обратиться к log-файлу сервера управления runtime.log.

По завершении работы исполняемого модуля, графический интерфейс самостоятельно загрузит с сервера управления файлы из *области F* при условии их физического существования и ненулевого размера (сохранение происходит в папку data). Произойдет автоматическое переключение на вкладку результатов работы, представляющей собой текстовый редактор: в случае успешного завершения вычислений появится окно с текстом «Работа над задачей успешно завершена».

5. Заключение

Разработан программный комплекс, в котором реализованы методы и алгоритмы решения оптимизационных задач и задач улучшения управления для различных прикладных областей на кластерном вычислительном устройстве. Парадигма параллельных вычислений в высшей степени соответствует самой природе рассматриваемых задач, связанных с множественностью однотипных операций на верхнем уровне. В программном комплексе успешно реализованы алгоритмы аппроксимации, оптимизации и улучшения приближенно оптимального управления.

Программный комплекс содержит: сервер управления (средство управления и контроля комплексом), управляющие модули (инструменты формирования среды для решения поставленных задач), исполняемые модули (выполняют счет поставленных задач) и интерфейс пользователя (средство запуска счета параметризованных задач).

Для параллельной реализации программного комплекса была использована гетерогенная аппаратная среда. Компоненты программного комплекса физически разделены. Графический интерфейс, сервер управления и управляющие модули работают на платформе IBM PC, а аппаратная платформа для исполняемых модулей вообще не фиксируется. В составе программного комплекса в частности есть исполняемые модули, работающие на аппаратной платформе IBM PC, модули, выполняющиеся на аппаратной платформе суперкомпьютеров «СКИФ» кластерного уровня. Аппаратная платформа суперкомпьютеров «СКИФ» включает: управляющую ЭВМ (фронтенд), вычислительные узлы кластерного уровня; системную сеть КВУ (SCI), объединяющую вычислительные узлы; вспомогательную сеть (семейства Ethernet, с поддержкой TCP/IP), объединяющую управляющую ЭВМ и вычислительные узлы.

Такая гибкость при работе с исполняемыми модулями возможна из-за активного использования протокола SSH (Secure Shell) при построении управляющих модулей, сетевого протокола прикладного уровня, позволяющего производить удаленное управление операционной системой и передачу файлов.

Исследование возможностей и эффективности ПК ISCON проведено на ряде модельных и прикладных задач, включая многокомпонентную модель региона, модель движения вертолета в нештатной ситуации, модель спуска космического челнока в атмосфере с минимальным теплоподводом, модели упругих систем. Результаты позволяют сделать вывод о значительном ускорении вычислительного процесса, близкого к линейному до определенного (оптимального) количества вычислительных узлов. Это подтверждает теоретический прогноз о высокой эффективности распараллеливания.

Список литературы

[1] Белышев Д. В., Блинов А. О., Фраленко В. П. *Параллельный алгоритм аппроксимации моделей управляемых систем.* // Труды Четвертой Международной конференции «Параллельные вычисления и задачи управления» PACO'2008. Москва, 27-29 октября 2008 г. — Москва: Институт проблем управления им. В. А. Трапезникова РАН., 2008. — ISBN 978-5-91450-016-7, с. 968–978. ↑3.1

[2] Гурман В. И. Принцип расширения в задачах управления. — Москва: Наука. Физматлит, 1997. — 288 с. ↑3.2

[3] Гурман В. И., Квоков В. Н., Ухин М. Ю. *Приближенные методы оптимизации управления летательным аппаратом.* // Автоматика и телемеханика, № 3, 2008, с. 191–201. ↑3.2, 3.2.3

[4] Коваленко М. Р., Матвеев Г. А., Осипов В. И., Трушкова Е. А. *Параллельный алгоритм улучшения управления.* // Труды Четвертой Международной конференции «Параллельные вычисления и задачи управления» PACO'2008. Москва, 27-29 октября 2008 г. — Москва: Институт проблем управления им. В. А. Трапезникова РАН., 2008. — ISBN 978-5-91450-016-7, с. 979–984. ↑3.2

Исследовательский центр процессов управления ИПС РАН

Исследовательский центр искусственного интеллекта ИПС РАН

A. O. Blinov, V. I. Gurman, E. A. Trushkova, V. P. Fralenko. *Software package of improvement and optimization of control lows* // Proceedings of Program Systems institute scientific conference "Program systems: Theory and applications". — Pereslavl-Zalesskij, v. **1**, 2009. — p. 25–41. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. A software package of improving and optimization of control processes for applications in various fields is developed. Its algorithms are successfully realized on the cluster computing device. The capacity and effectiveness of the package are investigated in a number of model examples and applications such as multicomponent model of the region, model of the helicopter maneuvers in emergencies, model of the space shuttle descending in the atmosphere with a minimum of heat supply.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 517.977

О. В. Моржин

Нелокальная оптимизация позиционных управлений для дифференциальных систем в границах трубок достижимости и разрешимости

Аннотация. Разработан метод оптимизации позиционных управлений для нелинейных дифференциальных систем в границах трубок достижимости и разрешимости (управляемости). Метод базируется на реализации принципа оптимальности Р. Беллмана на дискретных множествах как аппроксимациях этих трубок. Для аппроксимации траекторных множеств развит метод сечений. Проведены численные эксперименты, иллюстрирующие эффективность предлагаемой методики.

Введение

Для решения задач оптимального позиционного управления (далее: ЗОПзУ) нелинейными дифференциальными системами известны различные методики [1–6]. Специфической чертой подхода Н.Н. Моисеева [4] является возможность декомпозиции ЗОПзУ с аддитивным целевым функционалом на «элементарные задачи» за счет реализации принципа оптимальности Р. Беллмана [1] на априорных «шкалах состояний» в пространстве «время – состояния». Известный метод «блуждающих трубок» [4], призванный дать оценку области, в которой требуется введение «шкал состояний», при данных начальном и/или целевом множествах, обеспечивает, вообще говоря, локальное решение задачи, будучи определенным на некоторых подмножествах трубок достижимости и/или разрешимости (управляемости) [5].

Эффективность подхода определяется суммарной трудоемкостью значительного числа «элементарных операций» по вычислению траекторий системы для перехода из каждого узла на дискретном множестве состояний, введенном для одного узлового момента времени, в каждый узел на аналогичном множестве для последующего узла по

Работа выполнена при поддержке РФФИ, грант 08-01-00945-а.

времени. Иными словами, требуется найти (по возможности глобальное) решение задачи оптимального программного управления (ЗОПрУ) для рассматриваемой динамической системы при терминальных ограничениях, где программными управлениями рассматривают функции или параметры.

Конструктивными направлениями в развитии подхода Н.Н. Моисеева представляются: 1) предварительная аппроксимация траекторных трубок (трубки достижимости при заданном начальном множестве или трубки разрешимости при данном целевом множестве); 2) решение ЗОПрУ на основе современных алгоритмических и программных средств.

1. Формулировка ЗОПрУ и ЗОПзУ

Рассматривается *управляемая система*, описываемая векторным обыкновенным дифференциальным уравнением в форме Коши:

$$\dot{x}(t) = f(x(t), u, t), \tag{1}$$

где $T = [t_S, t_F] \ni t$ – заданный отрезок; $x(t) = \big(x_1(t), ..., x_{d(x)}(t)\big) \in E^{d(x)}$ – состояние системы в момент $t \in T$; управление

$$u \in U \subset \mathrm{comp}\big(E^{d(u)}\big), \tag{2}$$

где $\mathrm{comp}\big(E^{d(u)}\big)$ — множество всех компактов евклидова пространства $E^{d(u)}$.

На функцию $f(x, u, t)$ накладываются условия:

1) *условие Липшица* по переменной состояния:

$$\begin{gathered}\exists L^f(D) \in (0, \infty): \ \big\|f(x^1, u, t) - f(x^2, u, t)\big\| \le L^f(D)\,\big\|x^1 - x^2\big\| \\ \forall\ \big(x^j, t\big) \in D\ (j = \overline{1,2}),\ \forall\ u \in U,\ D \subset \mathrm{comp}\left(E^{d(x)} \times T\right);\end{gathered} \tag{3}$$

2) *условие подлинейного роста*:

$$\begin{gathered}\exists\ M^f(D) \in (0, \infty): \\ \|f(x, u, t)\| \le M^f(D)\,(1 + \|x\|)\ \forall\ (x, u, t) \in E^{d(x)} \times U \times T;\end{gathered} \tag{4}$$

3) *непрерывности* по совокупности аргументов (x, u):

$$f_i(x, u, t) \in C\left(E^{d(x)} \times U \times T\right)\ (i = \overline{1, d(x)}); \tag{5}$$

4) *непрерывной дифференцируемости* по (x, u):

$$\begin{gathered}\frac{\partial f_i(x, u, t)}{\partial x_j},\ \frac{\partial f_i(x, u, t)}{\partial u_k} \in C\left(E^{d(x)} \times U \times T\right) \\ (i, j = \overline{1, d(x)},\ k = \overline{1, d(u)}).\end{gathered} \tag{6}$$

Функция $u = u(t)$ $(t \in T)$ называется *программным* управлением, функция $u = u(t, x)$ $(t \in T,\ x \in E^{d(x)})$ – *позиционным* управлением.

Классы *доступных* программных и позиционных управлений:

$$\mathcal{U}_{\text{прогр}} = \left\{ u \in PC\left(T, E^{d(u)}\right) :\ u(t) \in U\ (t \in T) \right\}, \tag{7}$$

$$\mathcal{U}_{\text{позиц}} = \left\{ u \in PC\left(T \times X, E^{d(u)}\right) :\ u(t,x) \in U\ (t \in T,\ x \in X) \right\}, \tag{8}$$

где компакт X определяется в контексте конкретной задачи управления.

Декартово произведение $T \times E^{d(x)}$ назовем *пространством позиций*, его точки (t, x) – *позициями*. Пусть задана позиция (t^I, x^I) $(t^I \in [t_S, t_F))$, из которой стартуют траектории системы (1) – (6) при различных доступных программных управлениях, образуя пучок траекторий. Согласно теореме существования и единственности для любой функции $u(\cdot) \in \mathcal{U}_{\text{прогр}}$ существует единственное решение $x[t] = x(t|u, (t^I, x^I))$ $(t \in [t^I, t^{II}],\ t^{II} \le t_F)$ в виде кусочно-дифференцируемой функции. Условие подлинейного роста, как известно, является достаточным условием ограниченности пучка траекторий при $t \in [t^I, t^{II}]$.

Для системы (1) задается начальное условие:

$$x(t^I) = x^I(a),\ a = \left(a_1, ..., a_{d(a)}\right) \in A,$$

где a – вектор *управляющих параметров*,

$$A = \left[a_1^-, a_1^+\right] \times ... \times \left[a_{d(a)}^-, a_{d(a)}^+\right] \subset \text{comp}\left(E^{d(a)}\right)\ (d(a) \le d(x)).$$

Поточечные и концевые фазовые ограничения для системы (1):

$$g_k\left(x(t), \{u(t) \wedge u(t,x)\}, t\right) \le 0\ (t \in T,\ k = \overline{1, d(g)}), \tag{9}$$

$$\begin{gathered} h_k\left(x(t^{II})\right) = 0\ \left(k = \overline{1, d(he)}\right), \\ h_k\left(x(t^{II})\right) \le 0\ \left(k = \overline{d(he)+1, d(h)}\right). \end{gathered} \tag{10}$$

На некотором отрезке $[t^I, t^{II}] \subseteq T$ рассматривается *целевой критерий*

$$I\left(u(\cdot), a\right) = F\left(x(t^{II})\right) + \int\limits_{t^I}^{t^{II}} f^0(x(t), u(t), t)dt \to \inf\ (\min).$$

На функции $g_k(x,u,t)$ $(k = \overline{1,d(g)})$, $h_k(x)$ $(k = \overline{1,d(h)})$, $f^0(x,u,t)$, $F(x)$ налагаются стандартные условия по аналогии с (3) – (6).

Управления $u(\cdot) \in \mathcal{U}_{\text{прогр}}$, $a \in A$ называются *допустимыми*, если траектория $x(\cdot|u,a,x^I(a))$ удовлетворяет фазовым ограничениям ((9), (10)) во всей области определения $[t^I, t^{II}]$ $(t_S \le t^I < t^{II} \le t_F)$.

Будем говорить, что задача на отрезке $[t^I, t^{II}]$ имеет *решение* в виде *локально-оптимальных* управлений $u^{\text{л}}(\cdot) \in \mathcal{U}_{\text{прогр}}$ и $a^{\text{л}} \in A$, если $\exists \varepsilon > 0$ такое, что $\forall v(\cdot) \in \mathcal{U}_{\text{прогр}}$ и $\forall c \in A$ таких, что траектория $x(\cdot|v,c)$ удовлетворяет фазовым ограничениям и условию

$$\|x(t|v,c) - x(t|u^{\text{л}}, a^{\text{л}})\| < \varepsilon \ (t \in [t^I, t^{II}]),$$

верно неравенство

$$I\left(v(\cdot),c\right) \ge I\left(u^{\text{л}}(\cdot),a^{\text{л}}\right) = \inf_{u(\cdot)\in\mathcal{U},\ a\in A} I\left(u(\cdot),a\right) > -\infty.$$

Локально-оптимальные управления $u^{\text{л}}(\cdot) \in \mathcal{U}_{\text{прогр}}$ и $a^{\text{л}} \in A$ называются *глобально-оптимальными* и обозначаются $u^{\text{г}}(\cdot)$ и $a^{\text{г}}$, если $\forall v(\cdot) \in \mathcal{U}_{\text{прогр}}$ и $\forall c \in A$ таких, что траектория $x(\cdot|v,c)$ удовлетворяет фазовым ограничениям и не обязана удовлетворять условию близости траекторий $x(\cdot|v,c)$ и $x(\cdot|u^{\text{л}},a^{\text{л}})$, выполнено неравенство $I\left(v(\cdot),c\right) \ge I\left(u^{\text{г}}(\cdot),a^{\text{г}}\right) = \min\limits_j \left\{I\left(u_j^{\text{л}}(\cdot),a_j^{\text{л}}\right)\right\}$, где индекс j пробегает конечное множество локальных минимумов целевого функционала.

Множеством достижимости $\mathcal{R}(t^I,x^I,t^{II})$ системы (1) – (7) из позиции $\{t^I,x^I\}$ $(t_S \le t^I < t^{II} \le t_F)$ называется множество, состоящее из всевозможных состояний системы в момент t^{II} на любых доступных управлениях $u(\cdot) \in \mathcal{U}_{\text{прогр}}$.

Трубкой достижимости, обозначаемой $\mathcal{R}(t^I,x^I,(t^I,t^{II}])$, системы (1) – (7) из позиции $\{t^I,x^I\}$ на полуотрезке $(t^I,t^{II}]$ $(t_S \le t^I < t^{II} \le t_F)$ будем называть объединение $\bigcup\limits_{t\in(t^I,t^{II}]} \mathcal{R}(t^I,x^I,t)$.

Аналогично определяются множества и трубки достижимости из компакта X^{t^I}, лежащего на гиперплоскости, пересекающей пространство позиций при $t = t^I \in [t_S, t_F)$.

На гиперплоскости, пересекающей пространство позиций в момент $t = t_F$, рассматривается компакт $\mathcal{M}$, называемый *целевым множеством*, на который траектории должны переводить систему.

Множеством разрешимости ($\mathcal{M}$-управляемости), обозначаемым $\mathcal{W}(\tau,t_F,\mathcal{M})$, для системы (1) – (7) в момент $\tau \in [t_S,t_F)$ при заданном

целевом множестве $\mathcal{M}$ называется множество, состоящее из всевозможных состояний в момент $t = \tau$, из которых система переводима на $\mathcal{M}$ при любых управлениях $u(\cdot) \in \mathcal{U}_{\text{прогр}}$.

Трубкой разрешимости, или трубкой $\mathcal{M}$-управляемости, обозначаемой $\mathcal{W}\left([t^I, t^{II}), t_F, \mathcal{M}\right)$ системы (1) – (7) на полуотрезке $[t^I, t^{II})$ $(t_S \le t^I < t^{II} < t_F)$ при заданном множестве $\mathcal{M}$ назовем объединение $\bigcup\limits_{t\in[t^I,t^{II})} \mathcal{W}(t, t_F, \mathcal{M})$.

Ограничения (9), (10) являются дополнительными критериями качества управления. В таких задачах может оказаться, что никакое доступное управление не является допустимым: не позволяет за выделенное время $t^{II} - t^I$ с требуемой точностью соблюсти эти ограничения.

При фазовых ограничениях речь идет об *условных* множествах достижимости и разрешимости, для аппроксимации которых не достаточно «отсечения» частей соответствующих множеств системы без фазовых ограничений. Будем обозначать одинаково условные и безусловные множества, прибегая при необходимости к уточнению.

Необходимое условие перевода на множество $\mathcal{M}$ траекторий системы, стартующих из X^{t^I}:

$$\mathcal{M} \bigcap \mathcal{R}\left(t^I, X^{t^I}, t_F\right) \neq \varnothing, \; X^{t^I} \bigcap \mathcal{W}\left(t^I, t_F, \mathcal{M}\right) \neq \varnothing.$$

Относительно системы (1)–(7) при условиях (9) и (10), с заданным целевым множеством $\mathcal{M}$, множеством начальных состояний X^{t_S}, имеющем непустое пересечение с $\mathcal{W}(t_S, t_F, \mathcal{M})$, рассматривается ЗОПзУ с целевым критерием

$$I(u, x) = \int_{t_S}^{t_F} z\left(x, u, t\right) dt \to \inf,$$

где на функцию $z(x, u, t)$ налагаются условия типа (3)–(6), управление $u(\cdot) \in \mathcal{U}_{\text{позиц}}$.

Аналогично формулируется ЗОПзУ в границах трубки достижимости при заданном множестве начальных состояний. Кроме того, может быть рассмотрена ЗОПзУ в границах пересечения трубок достижимости и разрешимости. Для определенности ограничимся задачами, в которых траектории определяются в границах трубок разрешимости.

Решением ЗОПзУ будем называть функцию $u(\cdot) \in \mathcal{U}_{\text{позиц}}$, определяющую управление системой для каждой позиции $\{t, x\}$ из трубки разрешимости.

В плане конструктивной реализации подхода Н.Н. Моисеева необходимо ввести в рассмотрение понятия аппроксимации целевого множества, трубки разрешимости и аппроксимирующего позиционного управления.

2. Схема численной оптимизации позиционного управления и вычислительные эксперименты

Сечение трубки разрешимости – множество разрешимости, является также множеством достижимости системы, получаемой из исходной при ее рассмотрении в «обратном времени» при целевом множестве как множестве начальных состояний. Управляющий параметр a_k $(k \in \overline{1, d(a)})$ при формулировке ЗОПрУ введен для формального задания и идентификации неизвестной k-й координаты вектора начального состояния $x^I(a)$ при аппроксимации таких множеств достижимости, где множество $\mathcal{M}$ играет роль множества начальных состояний. Таким образом, для аппроксимации трубок достижимости и разрешимости необходимо иметь аппарат аппроксимации множеств достижимости.

Пусть $t_S = 0$. На отрезке $[0, t_F]$, рассматриваемом в «прямом времени», вводится равномерная сетка с шагом $\Delta t = t_F/N$: $0 = t^0 < t^1 < ... < t^j < t^{j+1} < ... < t^{N-1} < t^N = t_F \quad (j = \overline{0, N})$. Вводится также сетка, узлы τ^r $(r = \overline{0, N})$ которой следуют по узлам t^j: $t^0 = \tau^N$, ..., $t^{N-r} = \tau^r$, ..., $t^{N-1} = \tau^1$, $t^N = \tau^0$.

Для краткости вместо $\mathcal{W}(t^j, t_F, \mathcal{M})$ будем писать $\mathcal{W}[t^j]$.

Под аппроксимацией (ограниченного) множества разрешимости $\mathcal{W}[t^j]$ системы (1) в момент $t^j \in [0, t_F)$ будем полагать множество $\widehat{\mathcal{W}}[t^j] = \widehat{\mathcal{W}}(t^j, t_F, \mathcal{M}) = \left\{x^i(t^j)\right\}$ $(i = \overline{1, q_{t^j}})$ такое, что

$$\left(\forall^\circ x^i(t^j)\ (i = \overline{1, q_{t^j}})\ x^i(t^j) \cap \mathcal{W}[t^j] \neq \varnothing\right)\ \&\ (\forall^\circ x(t^j) \in \mathcal{W}[t^j]$$

$$\exists i \in \overline{1, q_{t^j}}\ \&\ \exists \varepsilon \downarrow 0\ :\ \|x(t^j) - x^i(t^j)\| \leq \varepsilon\ \&\ x^i(t^j) \in \widehat{\mathcal{W}}[t^j]),$$

где q_{t^j} – количество элементов во множестве $\widehat{\mathcal{W}}[t^j]$.

Аппроксимацией множества $\mathcal{W}([0,t_F),t_F,\mathcal{M}) = \bigcup\limits_{t\in[0,t_F)} \mathcal{W}[t]$ является множество

$$\widehat{\mathcal{W}}([0,t_F),t_F,\mathcal{M}) = \bigcup_{j=1}^{N-1} \widehat{\mathcal{W}}[t^j] = \left\{\{x^i(t^j)\}, i=\overline{1,q_{t^j}},\ j=\overline{1,N-1}\right\}$$

с условием $\Delta t \le \delta \downarrow 0$.

В основу алгоритмов, изложенных в статье [7], положена идея построения контура множества достижимости в некоторый момент t^j по точкам:

1) сначала находятся координаты параллелепипеда, всех граней которого изнутри касается множество достижимости — для этого решается серия ЗОПрУ для поиска экстремальных (по возможности в глобальном смысле) значений каждой фазовой переменной;

2) затем в границах параллелепипеда вводится сетка с разбиением по каждой координате;

3) далее, в результате решения серии ЗОПрУ вычисляются экстремальные (по возможности все локальные) значения некоторой фазовой координаты при фиксированных значениях для всех остальных координат.

В контексте схемы оптимизации позиционного управления узлы, представляющие внутренность множества $\mathcal{W}[t^j]$ могут быть введены условно, так как при работе оптимизационного алгоритма — в случае несвязности множества достижимости — будут удалены такие элементы множества $\widehat{\mathcal{W}}[t^j]$, которые не принадлежат $\mathcal{W}[t^j]$.

Для эффективной реализации схемы необходимо учитывать возможности несвязности, вырождения в многообразие меньшей размерности для множества достижимости. Для аппроксимации, скажем, 3-мерного множества достижимости его 2-мерные сечения не обязательно строить также методом сечений: можно применить для упрощения расчетов, к примеру, метод опорных гиперплоскостей в предположении выпуклости этих плоских сечений. Алгоритмы метода сечений, в т.ч. в комбинации с другими методами, изложены в статьях [7,8]. Рассмотрим пример аппроксимации контура невыпуклого множества разрешимости.

Пример 1. Рассматривается система, описывающая управление с помощью $p(t)$ плоским маятником в среде с неизвестной вязкостью $q(t)$ (управление второго игрока) на отрезке времени $T=[0,2]\ni t$:

$$\dot{x}_1(t) = x_2(t),\ \dot{x}_2(t) = -0.15q(t)x_2(t) - 10.15\sin x_1(t) + p(t).$$

На управления игроков наложены ограничения: $|p(t)| \le 10$, $q(t) \in [0,1]$, $t \in T$. Целевое множество $\mathcal{M} = (0,0)$. Положим функцию $q(t) \equiv 0.5$ $(t \in T)$ и для построения контура множества $\mathcal{W}(0,2,\mathcal{M})$ рассмотрим данную систему в «обратном времени», полагая за начальный момент $t = 0$:

$$\dot{x}_1(t) = -x_2(t),$$
$$\dot{x}_2(t) = 0.15q(t)x_2(t) + 10.15\sin x_1(t) - p(t),\ |p(t)| \le 10.$$

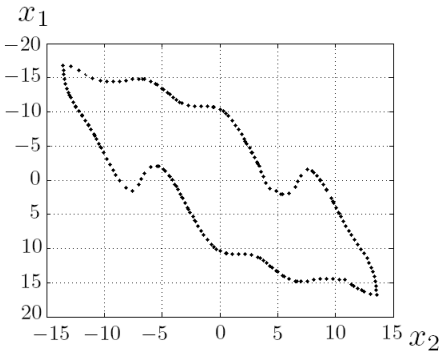


Рис. 1. Аппроксимация границы множества разрешимости в момент $t_S = 0$ (пример 1).

Показанный на рис. 1 результат в целом совпадает, как показал дополнительный расчет, с контуром, получаемым с помощью программы Я. Митчелла [9], которая реализует известный способ оценивания множеств достижимости на основе решения уравнения Беллмана [3]. □

Основным отличием методов сечений и опорных гиперплоскостей от методов эллипсоидов ([10], [11]) и других является получение аппроксимации множества достижимости, исходя непосредственно из определения этого множества.

Для нахождения семейства оптимальных программных управлений, аппроксимирующих оптимальное позиционное управление на частичном временном отрезке $[t^j, t^{j+1}]$ $(j \in \overline{0, N-1})$, проводится решение серии ЗОПрУ с целевым критерием

$$I_j(u) = \int_{t^j}^{t^{j+1}} z(x(t), u(t), t)dt \to \inf\ ,\ I(u) = \sum_{j=0}^{N-1} I_j(u),$$

относительно системы (1) – (7) при поточечных фазовых ограничениях (9) и краевых условиях

$$x(t^j) \in \widehat{\mathcal{W}}[t^j],\ x(t^{j+1}) \in \widehat{\mathcal{W}}[t^{j+1}].$$

На отрезке T вычисление оптимального позиционного управления проводится последовательно, переходя от отрезка $[t^{N-1}, t^N]$ к отрезку $[t^0, t^1]$, на основе принципа оптимальности Р. Беллмана.

Для каждой позиции $\{t^j, x^i(t^j)\}$ ($j \in \overline{0, N-1}$, $i \in \overline{1, q_{t^j}}$) определяется программное управление для движения на текущем частичном временном отрезке $[t^j, t^{j+1}]$. Тем самым проводится аппроксимация позиционного управления программными управлениями: функциями или параметрами. Ключевым отличием от схемы Н.Н. Моисеева является введение в рассмотрение вместо априорных «шкал состояний» аппроксимаций множеств разрешимости системы при заданном целевом множестве. В отличие от метода «блуждающих трубок», описанный метод характеризуется *нелокальностью*: управление рассматривается для всех узлов (позиций) из аппроксимации трубки разрешимости $\mathcal{W}([t_S, t_F), t_F, \mathcal{M})$.

Для простоты изложения ограничимся случаем аппроксимации позиционного управления семействами параметров.

Условие $x(t^{j+1}) \in \widehat{\mathcal{W}}[t^{j+1}]$ записывается посредством терминальных ограничений следующего вида: $x_i(t^j) - \overline{x}_i = 0$, где $\overline{x}$ – заданный числовой вектор, $i \in \overline{1, q_{t^{j+1}}}$, $j \in \overline{0, N-1}$.

В узлах сетки $\widehat{\mathcal{M}}$ функция цены φ позиционного управления $u(t, x)$ имеет только нулевые значения. Рассмотрим функцию

$$y_j\left(t, x^i(t^j), u^m\right) = \int_{t^j}^{t} z(x(\xi), u(\xi), \xi) d\xi,\ t \in [t^j, t^{j+1}].$$

Функция цены управляющего параметра $u^m \in \{u^m\}_{m=0}^M$ для позиции $\{t^{N-1}, x^i\}$:

$$\varphi(t^{N-1}, u^m, x^i(t^{N-1}, u^m)) = y_{N-1}(t^N, x^i(t^{N-1}), u^m),\ i \in \overline{1, q_{t^{N-1}}}.$$

Для позиции $\{t^j, x^i(t^j)\}$ ($i \in \overline{1, q_{t^j}}$, $j = \overline{0, N-1}$) функция цены управления u^m определяется как сумма значения $y_j(t^{j+1}, x^i(t^j), u^m)$ и соответствующих значений функции цены на последующих частичных временных отрезках.

Функция Беллмана на множестве $\mathcal{W}[t^j]$ и ее аппроксимация как объединение наименьших значений функции цены по всем элементам множества $\widehat{\mathcal{W}}[t^j]$:

$$\mathcal{B}(t^j,\mathcal{W}[t^j],\{\overline{u}\}) = \min_{\{u\}}\varphi\left(t^j,\mathcal{W}[t^j],\{u\}\right) \approx \mathcal{B}(t^j,\widehat{\mathcal{W}}[t^j],\{\widetilde{u}\}) =$$

$$= \bigcup_{i=1}^{q_{tj}} \min_{u}\varphi\left(t^j,x^i(t^j),u\right)\ \ (j=\overline{0,N-1}),$$

где $\{\widetilde{u}\}$ – объединение оптимальных программных управлений по всем узлам из $\widehat{\mathcal{W}}[t^j]$. Аналогично,

$$\varphi^{\max}(t^j,\mathcal{W}[t^j],\{\overline{\overline{u}}\}) = \max_{\{u\}}\varphi(t^j,\mathcal{W}[t^j],\{u\}) \approx \varphi^{\max}\left(t^j,\widehat{\mathcal{W}}[t^j],\widetilde{\widetilde{u}}\right) =$$

$$= \bigcup_{i=1}^{q_{tj}} \max_{u}\varphi(t^j,x^i(t^j),u).$$

На каждом отрезке $[t^j,t^{j+1}]$ $(j=\overline{0,N-1})$ известны: а) дискретное множество всех возможных начальных состояний в момент t^j в виде $\widehat{\mathcal{W}}[t^j]$ и дискретное множество всевозможных конечных состояний в момент t^{j+1} в виде $\widehat{\mathcal{W}}[t^{j+1}]$; б) для каждого узла $x^s\in\widehat{\mathcal{W}}[t^{j+1}]$ $(s\in\overline{1,q_{t^{j+1}}})$ минимальное и максимальное значения функции цены.

На отрезке $[t^j,t^{j+1}]$ $(j=\overline{0,N-1})$ проводится выбор управления $\widetilde{u}^i$ для каждого узла $x^i\in\widehat{\mathcal{W}}[t^j]$ $(i\in\overline{1,q_{t^j}})$ из набора $\{u^m\}_{m=0}^{M}$:

$$\widetilde{u}^i = \arg\min_{u^m}\left(y_j(t^{j+1},x^i(t^j),u^m)+\mathcal{B}(t^{j+1},x^s(t^{j+1}),\widetilde{u}^s)\right),$$

$$\mathcal{B}(t^j,x^i(t^j),\widetilde{u}^i) = \min_{u^m}\left(y_j(t^{j+1},x^i(t^j),u^m)+\mathcal{B}(t^{j+1},x^s(t^{j+1}),\widetilde{u}^s)\right),$$

$$i\in\overline{1,q_{tj}},\ \ s\in\overline{1,q_{tj+1}},\ \ j=\overline{0,N-1}.$$

В случае аппроксимации позиционного управления семействами программных управляющих функций требуется привлечение вычислительных средств для оптимизации программных управлений — вместо простой схемы выбора значений параметра, изложенной выше. Случай аппроксимации параметрами более простой и менее трудоемкий, поэтому с точки зрения сравнительной эффективности можно считать его более приемлемым.

По результатам работы алгоритмов, реализующих изложенную схему, строится «композиционное» программное управление, которое обеспечивает кусочно-дифференцируемую траекторию, по которой

производится перевод системы на целевое множество. Если реализуется случай аппроксимации позиционного управления семействами параметров, то может понадобиться «сглаживание» получаемой траектории. С этой целью проводится приближение «композиционного» управления как кусочно–постоянной функции полиномиальной функции достаточно высокой степени. Таким образом, итоговым этапом работы программной системы является применение результатов, насчитанных для всех аппроксимирующих сечений трубки разрешимости, для построения оптимального программного движения из любой позиции, взятой на аппроксимации этой трубки, на целевое множество.

Пример 2. Опишем результаты применения метода к простейшей модельной задаче уклонения движущегося объекта от преследования (летательного аппарата от ракеты) [12].

При исследовании модели ставится вспомогательная ЗОПзУ, которую рассмотрим в безразмерных координатах: найти нелокальное оптимальное управление $u(t,x)$ $((t,x_1,x_2)\in\mathcal{W}(t,t_F,\mathcal{M}),\ t\in[0,t_F))$, доставляющее глобальный минимум целевому функционалу вида (5)

$$I(u,x)=\int\limits_0^{t_F}\left(x_1\sqrt{1-(vu/x_1)^2}-v\sqrt{1-u^2}\right)dt$$

относительно динамической системы

$$\begin{gathered}\dot{x}_1=-x_1x_2-\lambda gvu/x_1,\ \dot{x}_2=-vx_2u,\\ (x_1(0),x_2(0))\in\mathcal{W}(0,t_F,\mathcal{M}),\end{gathered}\tag{11}$$

с ограничениями на управление и (введенными условно) фазовыми ограничениями:

$$\begin{gathered}u(t)\in[-0.985,\ 0.985],\ x_1(t)\in[v,\ 0.1],\\ x_2(t)\in[0.09,\ 0.12],\ t\in[0,t_F],\end{gathered}\tag{12}$$

$$\mathcal{M}=\{(x_1(t_F),x_2(t_F))|x_1(t_F)=v,\ x_2(t_F)\in[0.09,\ 0.12]\}\,.\tag{13}$$

Здесь $x_1(t)$, $x_2(t)$ $(t\in[0,t_F])$ – безразмерные функции скорости ракеты, движущейся на пассивном участке полета, и плотности атмосферы на высоте ее полета; $v=0.018$ – безразмерная постоянная скорость цели, преследуемой ракетой; $\lambda=1.079\cdot10^{-4}$, $g=9.81$, $u(t)$ $(t\in[0,t_F])$ – управление целью (синус угла наклона траектории цели в вертикальной плоскости). Функция цены характеризует значения дальности пуска ракеты для текущей позиции.

Эта задача состоит в поиске оптимального позиционного управления преследуемым объектом с целью уклонения от преследователя, используя при этом текущие координаты (высоту полета и плотность атмосферы) преследователя и расстояние между объектами. В задаче считается, что движение преследователя происходит на пассивном участке полета и при этом выполняется принцип преследования, согласно которому приравниваются управления для цели и преследователя, а также предполагается, что преследование начинается на одной высоте полета с целью.

В работах [13], [14] представлены результаты аппроксимации оптимального позиционного управления цели по описанной выше схеме с использованием приближения на каждом частичном временном отрезке позиционного управления семейством управляющих параметров.

На рис. 2, 3 показаны аппроксимирующие сечения для трубки достижимости системы, получаемой при рассмотрении системы (11) в «обратном времени» (при условиях (12)), где $\mathcal{M}$ из (13) играет роль множества начальных состояний (моменты $\tau^r = 0.5r$, $r = \overline{1,44}$). Рис. 4 представляет семейство оптимальных управляющих параметров, аппроксимирующих оптимальное позиционное управление на отрезке $[t^{N-2}, t^{N-1}]$, определенное на сетке узлов $x^i \in \widehat{\mathcal{W}}[t^{N-2}]$ ($i \in \overline{1, q_{t^{N-2}}}$).

В отличие от случая, соответствующего изображенному на рис. 4 графику, для большинства множеств $\widehat{\mathcal{W}}[t^j]$ в трубке разрешимости получаются разрывные по состоянию функции управления. Так как для каждого узла существует программное управление, то соответствующая траектория для перевода системы на следующее множество разрешимости (вплоть до множества $\mathcal{M}$) получается в виде непрерывной кусочно-дифференцируемой функции, ограниченной трубкой разрешимости. □

Вычислительные эксперименты проводились в системе MS Visual Studio 2005 с интегрированной системой Intel Visual Fortran 9. Важным преимуществом новой системы программирования является автоматическое распараллеливание потоков. Вообще говоря, реализация описываемых вычислительных схем довольно трудоемка, поэтому конструктивным является проведение расчетов на многопроцессорных системах. Распараллеливание алгоритмов не представляет труда, причем может быть осуществлено как на кластерах, так и на суперкомпьютерах различной архитектуры. Простота реализации

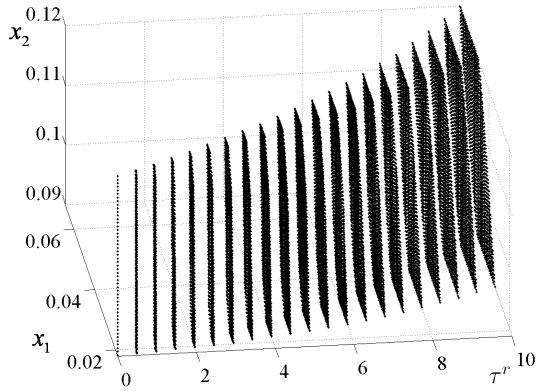


Рис. 2. Аппроксимирующие сечения трубки достижимости

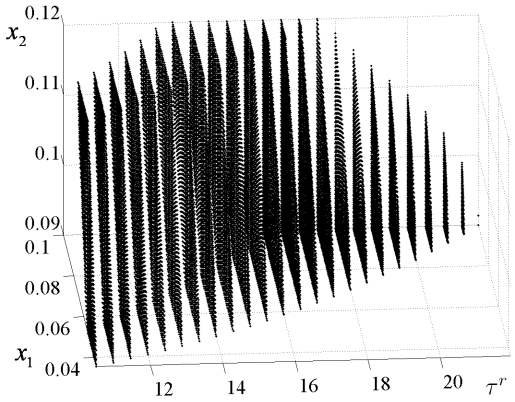


Рис. 3. Аппроксимирующие сечения трубки достижимости

обусловлена отсутствием зависимости между, например, вычислениями по аппроксимации множеств достижимости для различных моментов τ^r.

3. Вопросы численного решения ЗОПрУ

Как указано выше, «элементарной операцией» в алгоритмах аппроксимации множеств разрешимости и оптимизации позиционного

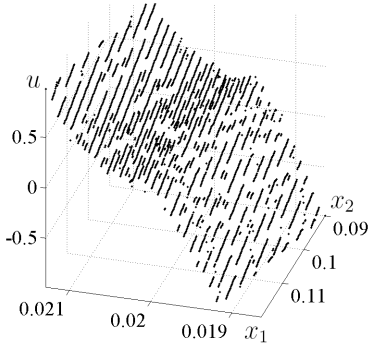


Рис. 4. Множество оптимальных значений параметров

управления является ЗОПрУ, которая может оказаться достаточно трудноразрешимой. Эффективность решения серии ЗОПрУ зависит от уровня надежности (включая уровень автоматизации) программного обеспечения.

Автором проведена реализация на языке Fortran ряда методов улучшения программных управлений [15–18].

На языке Maple разработана программа автоматического вывода конструкций принципа максимума Понтрягина и его линеаризованной версии. Для учета концевых и поточечных фазовых ограничений реализованы методы гладких и недифференцируемых по Фреше штрафных функционалов. Наряду с методами, работающими в функциональных пространствах, эффективно применяются методы, базирующиеся на сведении ЗОПрУ к задачам конечномерной оптимизации большой размерности за счет дискретизации по управлению ([18, 19]). Последний подход реализован, например, в пакете «TOMP» Д. Крафтом [20], причем, на наш взгляд, весьма успешно.

Заключение

Метод решения ЗОПзУ и алгоритм аппроксимации множеств разрешимости не используют дифференциальное уравнение Гамильтона-Якоби-Беллмана, опираются непосредственно на определение множеств разрешимости и принцип оптимальности Р. Беллмана, реализуемый на аппроксимации трубки разрешимости. В подходе «элементарной операцией» является ЗОПрУ, следовательно, эффективность

его зависит от эффективности методов и многометодных схем решения ЗОПрУ. Иными словами, аппарат аппроксимации траекторных трубок и решения ЗОПрУ, ЗОПзУ является единым с точки зрения достаточно полного исследования возможностей управления в нелинейных дифференциальных системах.

Список литературы

[1] Беллман Р., Калаба Р. Динамическое программирование и современные проблемы управления. — Пер. с англ. — М.: Наука, 1968. ↑(document)

[2] Кротов В.Ф., Гурман В.И. Методы и задачи оптимального управления. — М.: Наука, 1973. ↑

[3] Гурман В.И. Принцип расширения в задачах управления. — 2-е изд., перераб. и доп. — М.: Наука. Физматлит, 1997. ↑2

[4] Моисеев Н.Н. Численные методы в теории оптимальных систем. — М.: Наука, 1971. ↑(document)

[5] Куржанский А.Б. *Дифференциальные уравнения в задачах синтеза управлений. I* // Дифференц. уравнения. — **41**, № 1, 2005, с. 12–22. ↑(document)

[6] Bardi M., Capuzzo-Dolcetta I. Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations. — Boston: Birkhauser, 1988. ↑(document)

[7] Моржин О.В., Тятюшкин А.И. *Алгоритм метода сечений и программные средства для построения множеств достижимости* // Известия РАН. Теория и системы управления, № 1, 2008, с. 5–11. ↑2

[8] Тятюшкин А.И., Моржин О.В. *Алгоритм численного синтеза оптимального управления* // Автоматика и телемеханика. — **69**, № 4, 2008, с. 109–118. ↑2

[9] Mitchell I. A Toolbox of Level Set Methods. — http://www.cs.ubc.ca/~mitchell/ToolboxLS/, дата обращения: 16.02.2009. ↑2

[10] Черноусько Ф.Л. Оценивание фазового состояния динамических систем. Метод эллипсоидов. — М.: Наука, 1988. ↑2

[11] Kurzhanskiy A.A. Ellipsoidal Toolbox. — http://code.google.com/p/ellipsoids/, дата обращения: 16.02.2009. ↑2

[12] Тятюшкин А.И., Федунов Б.Е. *Возможности защиты от атакующей ракеты задней полусферы самолета вертикальным маневром* // Известия РАН. Теория и системы управления, № 1, 2006, с. 125-132. ↑2

[13] Моржин О.В., Тятюшкин А.И. *Оптимизация позиционного управления в одной задаче уклонения от преследователя* // Материалы IV междунар. симпозиума «Обобщенные решения в задачах управления» (23 – 28 июня 2008). – Улан-Удэ, 2008, с. 77-85, ISBN 978-5-9793-0067-2. ↑2

[14] Тятюшкин А.И., Моржин О.В. *Конструктивные методы оптимизации управлений в нелинейных системах* // Автоматика и телемеханика. — **70**, 2009, принята к изданию. ↑2

[15] Федоренко Р.П. Приближенное решение задач оптимального управления. — М.: Наука, 1978. ↑3

[16] Срочко В.А. Итерационные методы решения задач оптимального управления. — М.: Наука, 2000. ↑

[17] Булдаев А.С. Методы возмущений в задачах улучшения и оптимизации управляемых систем. — Улан-Удэ: Изд-во Бурятского государственного ун-та, 2008. ↑

[18] Тятюшкин А.И. Численные методы и программные средства оптимизации управляемых систем. — Новосибирск: Наука, 1992. ↑3

[19] Евтушенко Ю.Г. Методы решения экстремальных задач и их применение в системах оптимизации. — М.: Наука, 1982. ↑3

[20] Kraft D. *Algorithm 733: TOMP – Fortran modules for optimal control calculations* // ACM Transactions on Mathematical Software. — **20**, № 3, 1994, pp. 262–281. ↑3

Институт программных систем РАН, Исследовательский центр процессов управления; Бурятский государственный университет, Институт математики и информатики

O. V. Morzhin. *Nonlocal optimization of positional controls for differential systems in borders of the reachable and solvability tubes* // Proceedings of Program Systems institute scientific conference “Program systems: Theory and applications”. — Pereslavl-Zalesskij, v. **1**, 2009. — p. 43–58. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. It’s developed a method for optimization of positional controls for nonlinear differential systems in borders of the reachable and solvability (controllability) tubes. The method is based on realization of the Bellman’s optimality principle on discrete sets as approximations of these tubes. The section method is worked out for approximation of the trajectory sets. The numerical experiments were carried out and allow to illustrate the efficiency of the suggested technique.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 517

Е. Ф. Сачкова

Реализация и анализ работы алгоритмов приближенного решения задачи управления

Аннотация. Рассматривается компьютерная реализация алгоритма приближенного решения задачи управления для трехмерных нелинейных систем, описываемых обыкновенными дифференциальными уравнениями, с двумя линейными управлениями. Алгоритм основан на методе нильпотентной аппроксимации. Он реализован в системе Maple и апробирован на примере управления ориентацией катящейся по плоскости сферы.

1. Введение

Настоящая работа посвящена описанию программной реализации алгоритма перемещения трехмерной нелинейной системы в двумя линейными управлениями из заданного начального состояния в малую окрестность заданного финального состояния. Алгоритм разработан в предыдущей статье [1]. Написанная в системе компьютерной математики Maple [2] программа осуществляет итерационный процесс последовательного приближения системы к цели с помощью управлений другой, более простой, управляемой системы, являющейся аппроксимацией исходной. Широкие возможности программы обусловлены эффективностью как алгоритма, так и численных и символьных расчетов, осуществляемых системой Maple, поэтому она может быть полезна при решении большого числа прикладных задач, возникающих при управлении мобильными роботами [3], спутниками (при неограниченных управлениях) и иными импульсными системами [4, 5], качением твердых тел [6, 7].

Приводится пример апробации программы при решении задачи управления ориентацией сферы, которая катится по плоскости без проскальзывания и прокручивания. Рассмотрены управления, ранее не апробированные: кусочно-постоянные с одним переключением (см. [8]); управления, построенные с помощью линейных векторных полей на плоскости, имеющих особенность типа центр и типа фокус

Работа поддержана Российским Фондом Фундаментальных Исследований, проект No. 09-01-00246.

(см. [9]). Проанализирована эффективность соответствующих алгоритмов.

2. Постановка задачи управления

Рассматривается управляемая система вида

$$\dot{x} = u_1 X_1(x) + u_2 X_2(x), \qquad x \in \mathbb{R}^3, \qquad u = (u_1, u_2) \in \mathbb{R}^2, \tag{1}$$

$$X_1,\ X_2,\ X_3 = [X_1, X_2] = \frac{\partial X_2}{\partial x} X_1 - \frac{\partial X_1}{\partial x} X_2, \tag{2}$$

— линейно независимые гладкие векторные поля в $\mathbb{R}^3$,

с заданными граничными условиями и точностью:

$$x(0) = x^0, \quad x(T) = x^1, \quad x^0,\, x^1 \in \mathbb{R}^3, \quad T > 0, \quad \varepsilon > 0. \tag{3}$$

Требуется переместить систему (1), удовлетворяющую условию (2), из начального состояния x^0 в ε-окрестность конечного состояния x^1 за время $T > 0$.

Задача (1)–(3) разрешима в $\mathbb{R}^3$, так как из условия (2) следует полная управляемость системы (1) в $\mathbb{R}^3$ ([6], с. 73, 78).

В предыдущих работах [1, 8, 9] рассмотрен подход к решению задачи (1)–(3), основанный на методе нильпотентной аппроксимации системы (1), (2) в окрестности точки x^1.

Напомним необходимые сведения. Система вида (1) называется *нильпотентной*, если ее алгебра Ли

$$\begin{aligned}&\mathrm{Lie}(X_1, X_2) = \\ &\quad = \mathrm{span}(X_1, X_2, [X_1, X_2], \dots, [X_{i_1}, [X_{i_2}, \dots, [X_{i_k}, X_{i_{k+1}}] \dots]], \dots)\end{aligned}$$

нильпотентна, то есть для некоторого номера N все скобки Ли длины N равны нулю:

$$[X_{i_1}, [X_{i_2}, \dots, [X_{i_N}, X_{i_{N+1}}] \dots]] = 0 \quad \forall\, i_1, \dots, i_N, i_{N+1} \in \{1, 2\}.$$

Для систем (1), (2) нильпотентные аппроксимации, согласно теореме Белаиша ([1, 10]), имеют $N = 2$. Алгоритм нильпотентизации систем (1), (2) разработан в [1] и реализован в виде процедуры NilpApprox() программы FindControlLoc.

При решении задачи (1)–(3) в алгоритме используются две управляемые системы: исходная система (1), (2) и ее нильпотентная аппроксимация в окрестности точки x^1 (ее коэффициенты c_{ij}, $i, j \in \{1, 2\}$ см. в [1]), вычисленная в некоторых специальных координатах,

связанных с исходной системой (называемых привилегированными координатами):

$$\begin{aligned} &\dot z_1 = u_1, \quad \dot z_2 = u_2, \\ &\dot z_3 = u_1(c_{11}z_1 + c_{12}z_2) + u_2(c_{21}z_1 + c_{22}z_2), \qquad c_{12} \neq c_{21}, \end{aligned} \tag{4}$$

которая сводится заменой переменных

$$G(z) = \left(z_1,\ z_2,\ \frac{1}{c_{21} - c_{12}} \left(z_3 - \frac{c_{21} + c_{12}}{2} z_1 z_2 - \frac{c_{11}}{2} z_1^2 - \frac{c_{22}}{2} z_2^2 \right) \right) \tag{5}$$

к симметричной нильпотентной системе

$$\dot y_1 = u_1, \qquad \dot y_2 = u_2, \qquad \dot y_3 = (u_2 y_1 - u_1 y_2)/2. \tag{6}$$

Заметим, что граничным состояниям $(x^0,\ x^1)$ исходной системы соответствуют граничные состояния $(y^0, 0)$ системы (6).

Метод приближенного решения задачи управления (1)–(3) заключается в последовательном приближении системы (1) к целевой точке x^1 с помощью управлений системы (6), вычисляемых на каждой итерации и точно переводящих систему (6) в целевую точку.

Решения задачи управления

$$y(0) = y^0, \quad y(T) = 0, \quad T > 0 \tag{7}$$

для системы (6) найдены в пяти классах управлений: в тригонометрическом, кусочно-постоянном с одним переключением, оптимальном в смысле минимума функционала субримановой длины ([8]); центральном и фокусном, построенных с помощью линейных векторных полей на плоскости, имеющих соответствующую особенность типа центр и типа фокус (см. [9]). Все эти управления реализованы в виде пяти процедур программы FindControlLoc и составляют библиотеку управлений NilpControls.

3. Вычислительный алгоритм

Основываясь на работе [1], напомним алгоритм приближенного решения задачи управления (1)–(3). Этот итерационный алгоритм основан на методе нильпотентной аппроксимации. Из общей теории [11] следует, что для любой точки x^1 существует радиус сходимости $\delta > 0$ этого алгоритма, то есть такое число $\delta = \delta(x^1) > 0$, что для всех точек x^0, $|x^0 - x^1| < \delta$, построенная далее в алгоритме последовательность

приближений q^n сходится к x^1. Будем далее решать задачу перемещения системы (1) из точки x^0 в точку x^1 при условии $|x^0 - x^1| < \delta$; такую задачу управления будем называть *локальной*.

Обозначим через $\mathcal{U}$ класс управлений, используемых в алгоритме для перемещения системы (например, оптимальных в смысле некоторого функционала, тригонометрических, кусочно-постоянных). Напомним, что используемая далее матрица $F(x)$ имеет вид:

$$F(x) = (X_1 \mid X_2 \mid X_3)\,, \tag{8}$$

т. е. составлена по столбцам из векторов X_1, X_2 правой части системы (1) и их коммутатора X_3.

Алгоритм приближенного решения локальной задачи управления (1)–(3):

(1) **Проверка условия достижения цели:** если $|x^0 - x^1| < \varepsilon$, то цель достигнута и алгоритм останавливается. Далее предполагается, что $|x^0 - x^1| \geqslant \varepsilon$.

(2) **Вычисление нильпотентной аппроксимации** исходной системы (1), (2) в окрестности точки x^1: вычисляются коммутатор $X_3 = [X_1,\, X_2]$, матрица $F(x) = (X_1, X_2, X_3)(x)$, коэффициенты c_{ij}, $i,\, j = 1,\, 2$ по формулам (8)–(11) статьи [1].

(3) **Итерационный процесс.** В качестве начального приближения на первой итерации берется $q^0 = x^0$. Пусть q^{n-1} — приближение к целевой точке x^1, полученное на $(n-1)$-ой итерации.

   (a) Выбирается класс управлений $\mathcal{U}$.

   (b) Вычисляются координаты начальной точки q^{n-1} в привилегированных координатах, центрированных в целевой точке x^1: $z^{n-1} = F^{-1}(q^{n-1})(q^{n-1} - x^1)$.

   (c) Вычисляются координаты y^{n-1} начальной точки q^{n-1} в системе координат (y_1, y_2, y_3) системы (6):

$$y^{n-1} = G(c_{ij}, z^{n-1}),$$

   где отображение G задано формулой (5).

   (d) По формулам работ [8,9] вычисляются управления $\widehat{u}^n \in \mathcal{U}$, переводящие систему (6) из точки y^{n-1} в точку $0 \in \mathbb{R}^3$ за время T.

(e) Решается задача Коши для исходной системы (1) с управлениями $\widehat{u}^n$:

$$\dot{x} = \widehat{u}_1^n(t)X_1(x) + \widehat{u}_2^n(t)X_2(x),$$
$$x(0) = q^{n-1}, \quad t \in [0,\, T];$$

обозначим ее решение через $x^n(t)$.

(f) В качестве следующего приближения берется точка $q^n = x^n(T)$.

(g) Проверяется условие достижения цели: если $|q^n - x^1| < \varepsilon$, то цель достигнута и алгоритм останавливается. Если $|q^n - x^1| \geqslant \varepsilon$, то совершается переход к следующей итерации, к пункту (3a), и в качестве начального приближения берется q^n. Из сходимости алгоритма при условии $|x^0 - x^1| < \delta$ следует, что на некоторой итерации N выполнится условие $|q^N - x^1| < \varepsilon$, и алгоритм остановится.

(4) **Приближенное решение локальной задачи управления** дается последовательным применением управлений $\widehat{u}^1$, …, $\widehat{u}^N$, вычисленных на каждой итерации и перепараметризованных соответствующим образом:

$$u(t) = \begin{cases} N\widehat{u}^1(Nt), & t \in [0, T/N], \\ N\widehat{u}^2(Nt - T), & t \in [T/N, 2T/N], \\ \dots & \\ N\widehat{u}^N(Nt - (N-1)T), & t \in [T(N-1)/N, T]. \end{cases} \tag{9}$$

Смысл этих формул в следующем: если, например, управление $\widehat{u}^1(t)$ переводит точку q^0 в точку q^1 на отрезке времени длины T, то управление $N\widehat{u}^1(Nt)$ переводит точку q^0 в точку q^1 на отрезке времени длины T/N, и т.д.; в результате управление $u(t)$ определено на отрезке $t \in [0, T]$. Управление $u(t) = u(x^0, x^1, T, \varepsilon; t)$, полученное с помощью формул (9), переводит систему (1) за время $T > 0$ из точки x^0 в точку x^1 с заданной точностью $\varepsilon > 0$, следовательно, является приближенным решением локальной задачи управления (1)–(3).

С помощью этого локального алгоритма можно построить и глобальный алгоритм, введя по некоторому правилу промежуточные узлы $x_1^1, \dots, x_k^1$ так, что $x_1^1 = x^0$, $x_k^1 = x^1$ и $|x_{i+1}^1 - x_i^1| < \delta(x_i^1)$.

4. Описание программы FindControlLoc

Локальный алгоритм решения задачи управления (1)–(3) реализован в виде компьютерной программы FindControlLoc, написанной на входном языке системы Maple.

Отметим существенные характеристики программы:

1) возможность рассматривать произвольные системы вида (1), (2);

2) возможность выбирать произвольные граничные условия (3) (близкие в смысле δ);

3) решать задачу управления (1)–(3) в нескольких классах управлений.

Гибкость программы FindControlLoc обусловлена, в частности, тем, что она использует средства Maple-языка — языка процедурного программирования. Фрагменты алгоритма реализованы в нескольких процедурах, которые вызываются из тела программы. Это позволяет использовать один и тот же код для различных классов управлений, экономно проводить вычисления, например, для фиксированного финального состояния вычислять нильпотентную аппроксимацию один раз.

4.1. Описание процедур программы FindControlLoc

Процедуры вычислительного алгоритма. Описываемые ниже две процедуры реализуют п. 2, п. 3, (b),(c) алгоритма, изложенного в разделе 3.

Процедура NilpApprox();

Вычисляется нильпотентная аппроксимация системы (1), (2) в точке $x \in \mathbb{R}^3$.

Входные данные: $X_1,\ X_2 \in \mathrm{Vec}(\mathbb{R}^3)$, $x \in \mathbb{R}^3$;

Выполняемые действия: с помощью операций символьного дифференцирования и функций пакета расширения Maple linalg в символьном виде

1) вычисляется матрица $F^{-1}(x)$, где $F(x)$ — матрица (8);

2) вычисляются коэффициенты $c_{ij}(x)$, $i, j = 1, 2$, (см. [1], формулы (8) – (11)) системы (4) в точке $x \in \mathbb{R}^3$.

Выходные данные: 1) символьная 3×3 матрица $F^{-1}(x)$; 2) символьная 2×2 матрица $C(x)$, $C_{ij} = c_{ij}$, $i, j = 1, 2$.

Процедура ChangeCoords();

Выполняется замена переменных в окрестности точки $q^1 \in \mathbb{R}^3_x$: $\phi(x) = y$, $\phi(q^1) = 0$, где (x_1, x_2, x_3) — координаты исходной системы (1), (y_1, y_2, y_3) — координаты системы (6).

Входные данные: q^0, q^1, $C(q^1)$, $F^{-1}(x)$,

где q^0, q^1 — векторы длины 3; $C(q^1)$ — 2×2 матрица коэффициентов системы (4), вычисленная в точке $x = q^1$; $F^{-1}(x)$ — символьная 3×3 матрица.

Выполняемые действия:

1) вычисляется матрица $F^{-1}(q^0)$ с помощью встроенной функции eval(); затем вычисляются привилегированные координаты z^0 точки q^0 (см. формулу п. 3, (b) алгоритма раздела 3);

2) с помощью формулы (5) вычисляются координаты y^0 точки z^0 в системе координат (y_1, y_2, y_3) системы (6).

Выходные данные: вектор y^0 длины 3.

Встроенные процедуры пакета Maple DEtools. Приведенные ниже две процедуры вызываются из программы FindControlLoc.

Процедура dsolve();

Входные данные: $X_1,\ X_2 \in \mathrm{Vec}(\mathbb{R}^3)$, $u(t)$, $q^0 \in \mathbb{R}^3$;

Выполняемые действия: численно решается задача Коши для системы $\dot{x} = u_1(t)X_1(x) + u_2(t)X_2(x)$ с начальным условием $x(0) = q^0$ с помощью метода Рунге-Кутта 4-5 порядков.

Выходные данные: traj — таблица чисел, задающая значения вычисленной траектории.

Процедура odeplot() ;

Входные данные: traj — таблица чисел, T;

Выполняемые действия: Выводит трехмерное графическое изображение, плоские проекции и графики компонент траектории traj.

Выходные данные: графическое изображение траектории, являющейся решением задачи Коши.

Библиотека NilpControls.
Все процедуры библиотеки NilpControls вычисляют управления, являющиеся решениями задачи (6), (7) ([8, 9]).

Далее везде $r_0 = \sqrt{(y_1^0)^2 + (y_2^0)^2}$ — полярный радиус, φ_0 — полярный угол точки (y_1^0, y_2^0).

Процедура Optimal();

Вычисляются программные управления, оптимальные в смысле минимума функционала субримановой длины $L = \int_0^T \sqrt{u_1^2 + u_2^2}\, dt \to \min$ ([8]).

Входные данные: $y^0 \in \mathbb{R}^3$, T;

Выполняемые действия:

1) Если $y_3^0 \neq 0$, то вычисляются функции

$$u_1(t) = -d/T \cos(\psi - ct), \qquad u_2(t) = -d/T \sin(\psi - ct),$$

где $d = r_0\bar{t}/(2|\sin(\bar{t}/2)|)$, $\psi = \varphi_0 + (\operatorname{sign} y_3^0)\bar{t}/2$, $c = (\operatorname{sign} y_3^0)\bar{t}/T$, $\bar{t} = \bar{t}(y^0) \in (0, 2\pi)$ — численное решение уравнения $\dfrac{\bar{t} - \sin\bar{t}}{\sin^2(\bar{t}/2)} = \dfrac{8|y_3^0|}{r_0^2}$, найденное с помощью встроенной функции fsolve().

2) Если $r_0 \neq 0$, $y_3^0 = 0$, то $u_1(t) = -y_1^0/T$, $u_2(t) = -y_2^0/T$.

Выходные данные: $u(t)$, $t \in [0, T]$.

Процедура Trig();

Вычисляются программные управления в классе тригонометрических функций ([8]).

Входные данные: $y^0 \in \mathbb{R}^3$, T, параметр $\gamma \in \mathbb{R}$, $\gamma \neq -2y_2^0/T$;

Выполняемые действия: вычисляются функции

$u_1 = -y_1^0/T + \beta(\gamma)\sin(2\pi t/T)$, $\quad u_2 = -y_2^0/T + \gamma\cos(2\pi t/T)$,

где $\beta(\gamma) = 4\pi y_3^0/(T(2y_2^0 + \gamma T))$ (см. [8]).

Выходные данные: $u(t, \gamma)$, $t \in [0, T]$.

Процедура PieceConst();

Вычисляются кусочно-постоянные с одним переключением программные управления ([8]).

Входные данные: $y^0 \in \mathbb{R}^3$: $r_0 \neq 0$; T, параметр $\nu \in \mathbb{R}$, $\nu \neq \varphi_0$;

Выполняемые действия:

1) Если $y_3^0 \neq 0$, то вычисляются функции

$$u_1 = \begin{cases} \mu(\nu)\cos(\nu), & t \in [0, T/2), \\ -\mu(\nu)\cos(\nu) - \eta_1, & t \in [T/2, T], \end{cases}$$

$$u_2 = \begin{cases} \mu(\nu)\sin(\nu), & t \in [0, T/2), \\ -\mu(\nu)\sin(\nu) - \eta_1, & t \in [T/2, T], \end{cases}$$

где $\mu(\nu) = 4y_3^0/(T(y_2^0\cos(\nu) - y_1^0\sin(\nu)))$, $\eta_1 = 2y_1^0/T$, $\eta_2 = 2y_2^0/T$.

2) Если $y_3^0 = 0$, то $u_1(t) = -y_1^0/T$, $u_2(t) = -y_2^0/T$.

Выходные данные: 1) $u(t, \nu)$, $t \in [0, T]$; 2) $u(t)$, $t \in [0, T]$.

Процедура Centre();

Вычисляются программные управления, полученные с помощью линейного поля $v = (v_1, v_2) = (ay_1 + by_2, \quad cy_1 - ay_2)$, имеющего особенность в точке $(0, 0)$ типа центр. Управления этого класса имеют одно переключение, постоянны на второй половине временного отрезка ([9]).

Входные данные: $y^0 \in \mathbb{R}^3$: $r_0 \neq 0$; T, параметры $a, b, c \in \mathbb{R}$, удовлетворяющие условиям: $a^2 + b^2 + c^2 \neq 0$, $bc < 0$, $\Delta = \begin{vmatrix} a & b \\ c & -a \end{vmatrix} > 0$;

Выполняемые действия:

1) Если $y_3^0 \neq 0$, то вычисляется проекция траектории системы (6), выходящая из точки (y_1^0, y_2^0):

$$y_1(t) = y_1^0\cos(t\sqrt{\Delta}) + \frac{v_1^0}{\sqrt{\Delta}}\sin(t\sqrt{\Delta}),$$

$$y_2(t) = y_2^0\cos(t\sqrt{\Delta}) + \frac{v_2^0}{\sqrt{\Delta}}\sin(t\sqrt{\Delta}),$$

где $v_1^0 = ay_1^0 + by_2^0$, $v_2^0 = cy_1^0 - ay_2^0$;
вычисляются функции:

$$u_1(t) = \begin{cases} k\,\mathrm{sign}(by_3^0)(ay_1(kt) + by_2(kt)), & t \in [0, T/2], \\ -2p_1/T, & t \in [T/2, T], \end{cases}$$

$$u_2(t) = \begin{cases} k\,\mathrm{sign}(by_3^0)(cy_1(kt) - ay_2(kt)), & t \in [0, T/2], \\ -2p_2/T, & t \in [T/2, T], \end{cases}$$

где точка переключения $p = (y_1(\mathrm{sign}(by_3^0)t_p), y_2(\mathrm{sign}(by_3^0)t_p), 0)$, $t_p = \frac{2|y_3^0|}{|\delta_v^0|}$, $\delta_v^0 = -b(y_2^0)^2 + c(y_1^0)^2 - 2ay_1^0y_2^0$, $k = 2tp/T$.

2) Если $y_3^0 = 0$, то $u_1(t) = -y_1^0/T$, $u_2 = -y_2^0/T$.

Выходные данные: 1) $u(t, a, b, c)$, $t \in [0, T]$; 2) $u(t)$, $t \in [0, T]$.

Процедура Focus();

Вычисляются программные управления, полученные с помощью линейного поля $v = (v_1, v_2) = (ay_1 + by_2, \quad cy_1 + dy_2)$, имеющего особенность в точке $(0,0)$ типа фокус (при $a + d > 0$ — неустойчивый). Управления этого класса имеют одно переключение, постоянны на второй половине временного отрезка ([9]).

Входные данные: $y^0 \in \mathbb{R}^3$: $r_0 \neq 0$; T, параметры $a, b, c, d \in \mathbb{R}$, удовлетворяющие условиям: $a^2 + b^2 + c^2 + d^2 \neq 0$, $\quad bc < 0$, $\quad S = a + d > 0$, $\quad \Delta = \begin{vmatrix} a & b \\ c & d \end{vmatrix} > S^2/4$;

Выполняемые действия:

1)Если $y_3^0 \neq 0$, то вычисляется проекция траектории системы (6), выходящая из точки (y_1^0, y_2^0):

$$y_1(t) = e^{St/2}(y_1^0 \cos(t\sqrt{|D|}/2) + \frac{(a-d)y_1^0 + \operatorname{sign}(by_3^0)2by_2^0}{\sqrt{|D|}} \sin(t\sqrt{|D|}/2),$$

$$y_2(t) = e^{St/2}(y_2^0 \cos(t\sqrt{|D|}/2) - \frac{(a-d)y_2^0 + \operatorname{sign}(by_3^0)2cy_1^0}{\sqrt{|D|}} \sin(t\sqrt{|D|}/2),$$

где $D = S^2 - 4\Delta$;
вычисляются функции:

$$u_1(t) = \begin{cases} k(ay_1(kt) + \operatorname{sign}(y_3^0)|b|y_2(kt)), & t \in [0, T/2], \\ -2p_1/T, & t \in [T/2, T], \end{cases}$$

$$u_2(t) = \begin{cases} k(-\operatorname{sign}(y_3^0)|c|y_1(kt) + dy_2(kt)), & t \in [0, T/2], \\ -2p_2/T, & t \in [T/2, T], \end{cases}$$

где точка переключения $p = (y_1(tp), y_2(tp), 0)$, $t_p = 1/S \ln(1 + 2Sy_3^0/\delta^0)$, $\delta^0 = \operatorname{sign}(by_3^0)(b(y_2^0)^2 - c(y_1^0)^2) + (a-d)y_1^0 y_2^0$, $k = 2t_p/T$.

2) Если $y_3^0 = 0$, то $u_1(t) = -y_1^0/T$, $u_2 = -y_2^0/T$.

Выходные данные: 1) $u(t, a, b, c, d)$, $t \in [0, T]$; 2) $u(t)$, $t \in [0, T]$.

Проанализируем описанную библиотеку. Управления Optimal и Trig решают задачу управления (6), (7) во всем пространстве состояний $\mathbb{R}^3$, а управления PieceConst, Centre, Focus, в отличие от первых двух, только в $\mathbb{R}^3 \setminus \{y_1^2 + y_2^2 = 0\}$, что означает, что они неприменимы в случае, если $y^0 \in \{y_1^2 + y_2^2 = 0\}$, и требуется искать другую стратегию перемещения. Все управления библиотеки, за исключением Trig, порождают управления с обратной связью, что обусловливает их устойчивость к небольшим погрешностям начального положения.

4.2. Схема программы FindControlLoc

Схема основной программы FindControlLoc отражает схему описанного выше алгоритма приближенного решения задачи управления (1)–(3).

Программа FindControlLoc.

Входные данные: $X_1,\ X_2 \in \mathrm{Vec}(\mathbb{R}^3)$, $x^0,\ x^1 \in \mathbb{R}^3$, $T > 0$, $\varepsilon > 0$, nc, par(nc);

Выполняемые действия:

(1) Инициализируются пакеты системы Maple linalg, DEtools, plottools, plots.

(2) Инициализируются процедуры NilpApprox, ChangeCoords, библиотека NilpControls.

(3) Считываются данные: X_1, X_2, x^1.

(4) Вычисляется матрица $C = \mathrm{NilpApprox}(X_1, X_2, x^1)$.

(5) Считываются данные x^0, T, ε, выбирается процедура-управление nc из библиотеки NilpControls, считываются параметры для этой процедуры par(nc).

(6) Входное начальное состояние x^0 запоминается в переменной q^0, счетчику итераций присваивается значение $i := 0$.

(7) Итерационный процесс реализуется с помощью условного цикла While(Dist(q^0, x^1)$\geqslant \varepsilon$), где Dist(q^0, x^1) — евклидово расстояние между точками q^0, x^1.

i- я итерация: пусть q^0 — приближение к x^1 на $(i-1)$-ой итерации (считаем $q^0 = x^0$ приближением к x^1 на нулевой итерации);

(a) счетчик итераций увеличивается на единицу: $i := i + 1$;

(b) вычисляются координаты начального состояния системы (6): $y^0 = \mathrm{ChangeCoords}(q^0, x^1, C)$;

(c) с помощью выбранной процедуры nc вычисляются управления и запоминаются в массиве переменной длины $u[i] = \mathrm{NilpControls(nc)}(y^0, T, \mathrm{par(nc)})$;

(d) численно решается задача Коши: traj = dsolve(X_1, X_2, $u[i](t)$, q^0);

(e) вычисляется следующее приближение к финальной точке x^1: $q^0 = \mathrm{traj(T)}$.

(8) Если Dist(q^0, x^1)$< \varepsilon$, то в переменную N запоминается количество итераций; если $N = 0$, то программа останавливается, если $N > 0$, то переходит к следующему пункту.

(9) Управления $u[i]$, $i = 1, ..., N$, перепараметризуются по правилу (9), умножаются на функции $\delta[i]$ с соответствующими номерами i, где

$$\delta[i] = \begin{cases} 0, & t \in [0, (i-1)T/N], \\ 1, & t \in [(i-1)T/N, iT/N], \\ 0, & t \in [iT/N, T], \end{cases}$$

и полученные функции, определенные на временном отрезке $[0, T]$, суммируются. Получаются искомые управления $u(t)$, соответствующие формулам (9).

(10) Полученный результат проверяется прямой подстановкой управления $u(t)$ в исходную систему (1) и численным решением задачи Коши: traj = dsolve(X_1, X_2, $u(t)$, x^0); вычислением состояния, в которое приходит система: q^1 = traj(T); вычислением Dist(q^1, x^1).

(11) *Вывод результата.*
   (a) Если Dist(q^1, x^1)$< \varepsilon$, то управления $u(t)$ выводятся аналитически и графически. Для визуализации результата с помощью функции odeplot выводится трехмерное изображение траектории движения исходной системы; для анализа результата выводятся ее двумерные проекции и графики компонент.
   (b) Если Dist(q^1, x^1)$\geqslant \varepsilon$, то выводится сообщение об ошибке.

Выходные данные: либо (a) графическое и аналитическое представление управления $u(t)$, $t \in [0, T]$, либо (b) — сообщение об ошибке.

Проанализируем возможные причины ошибки:

(1) недостаточная точность в итерационном процессе;
(2) наличие фазовых ограничений: траектория системы может выходить за пределы области;
(3) задача не локальна для выбранного класса управлений.

5. Управление ориентацией катящейся по плоскости сферы

5.1. Апробация программы FindControlLoc

Управляемая система, описывающая качение сферы по плоскости без прокручивания и проскальзывания, описана в книгах [6, 7].

Рассмотрим подсистему этой системы, описывающую изменение ориентации сферы. Переходя в этой подсистеме от ортогональных 3×3 матриц к кватернионам (см. [12]) и применяя проекцию на трехмерное пространство, получим следующую систему:

(10)
$$\dot{x_1} = -x_3 u_1 + x_2 u_2,$$
$$\dot{x_2} = -\sqrt{1 - x_1^2 - x_2^2 - x_3^2}\, u_1 - x_1 u_2,$$
$$\dot{x_3} = x_1 u_1 - \sqrt{1 - x_1^2 - x_2^2 - x_3^2}\, u_2,$$
$$q = (x_1, x_2, x_3) \in B^3 = \left\{x_1^2 + x_2^2 + x_3^2 < 1\right\}, \qquad u = (u_1, u_2) \in \mathbb{R}^2.$$

С помощью компьютерной программы FindControlLoc задача управления (1)–(3) для системы (10) решена в пяти классах управлений библиотеки NilpControls.

Пример работы программы FindControlLoc.

Входные данные: векторные поля системы (10)

$$X_1 = \left(-x_3,\ -\sqrt{1 - x_1^2 - x_2^2 - x_3^2}, x_1\right)^T,$$
$$X_2 = \left(x_2,\ -x_1,\ -\sqrt{1 - x_1^2 - x_2^2 - x_3^2}\right)^T;$$

граничные условия $x^0 = [.3555263893, .2583050417, .1359392951]$, $x^1 = [.1381591491, 0., .05841275134]$; время $T = 1.$, точность $\varepsilon = 10^{-6}$, nc $\in$ NilpControls, par(nc).

Выполняемые действия:
Dist(x^0, x^1)= .3463818366;
Коэффициенты нильпотентной аппроксимации (4) в точке x^1:
$$C = \begin{pmatrix} -.06987008492 & -0.5 \\ 0.5 & -.0698700849 \end{pmatrix}$$

Процедура nc	*Параметры par(nc)*	*Число итераций*
Optimal	—	$N = 6$
Trig	1.	$N = 5$
PieceConst	1.8849555922	$N = 5$
Centre	$(1, 4, -1)$	$N = 8$
Centre	$(0, 5, -3)$	$N = 4$
Focus	$(0, 5, -2, 0.3)$	$N = 3$

Выходные данные:
1) $u^{Opt}(t)$, $t \in [0,1]$, рис. 1, 2;
2) $u^{Trg}(t)$, $t \in [0,1]$, рис. 3, 4;
3) $u^{PC}(t)$, $t \in [0,1]$, рис. 5, 6;
4) $u^{Centre}(t)$, $t \in [0,1]$, рис. 7, 8;
5) $u^{Focus}(t)$, $t \in [0,1]$, рис. 9, 10.

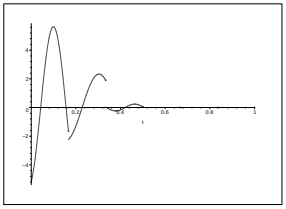


Рис. 1. $u_1^{Opt}(t)$

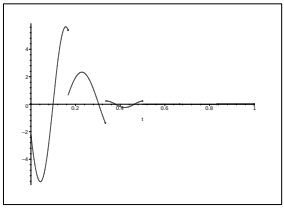


Рис. 2. $u_2^{Opt}(t)$

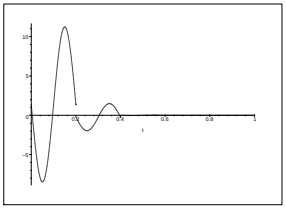


Рис. 3. $u_1^{Trg}(t)$

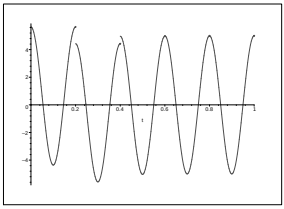


Рис. 4. $u_2^{Trg}(t)$

5.2. Анализ работы программы FindControlLoc

Дадим некоторые рекомендации по использованию библиотеки NilpControls на основании полученных практических наблюдений и анализа графиков управлений, вычисленных с помощью программы FindControlLoc.

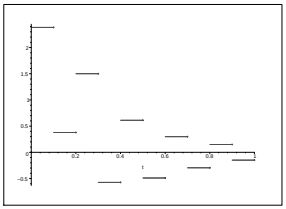


Рис. 5. $u_1^{PC}(t)$

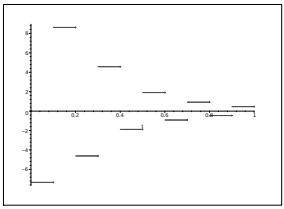


Рис. 6. $u_2^{PC}(t)$

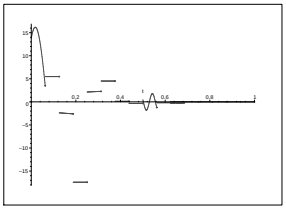


Рис. 7. $u_1^{Centre}(t)$

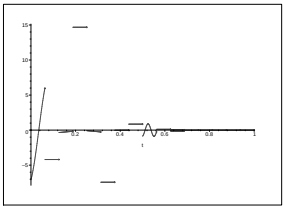


Рис. 8. $u_2^{Centre}(t)$

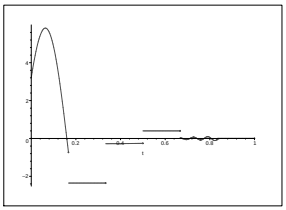


Рис. 9. $u_1^{Focus}(t)$

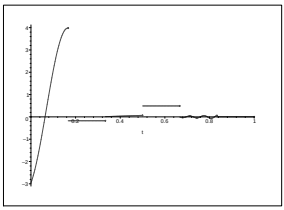


Рис. 10. $u_2^{Focus}(t)$

Если начальное расстояние $\mathrm{Dist}(x^0, x^1)$ мало в смысле константы δ (см. п. 3) и нет ограничений в пространстве состояний, то рекомендуется $\mathcal{U} = \mathrm{Optimal}$ (так как управления этого класса не зависят от параметров, «жесткие»). Из анализа графиков управлений (рис. 1, 2)

видно, что система локализуется в окрестности целевой точки за время $t \approx T/2$.

Если начальное расстояние $\mathrm{Dist}(x^0, x^1)$ достаточно велико и требуется большая скорость перемещения системы, как в задачах управления спутником (требуются неограниченные управления), то рекомендуется $\mathcal{U} = \mathrm{Trig}$. Тригонометрические управления зависят от параметра γ, настройка которого влияет на характер движения системы, например, в разобранном выше случае краевых условий при $\varepsilon = 10^{-3}$ и $\gamma = 5$. количество итераций $N = 9$, максимальная амплитуда управления $u_2(t)$ равна 90, скорость вхождения в ε-окрестность точки x^1 равна 40, в сравнении, при $\varepsilon = 10^{-6}$ и $\gamma = 1$. количество итераций $N = 5$, максимальная амплитуда управления $u_2(t)$ равна 20, скорость вхождения в ε-окрестность точки x^1 равна 5 (см. рис. 3, 4), тогда как во всех остальных случаях, кроме кусочно-постоянного, скорость равна нулю. Заметим, что при $\gamma = 0.5$ и $\varepsilon = 10^{-3}$ траектория системы выходит из области $B^3 = \left\{x_1^2 + x_2^2 + x_3^2 < 1\right\}$, что является одной из причин ошибок программы. В случае $\mathcal{U}=$ PieceConst движение происходит также с большой скоростью, но, несмотря на простоту управлений, характер движения сложен: из-за больших амплитуд и частой смены знака функций $u_1(t)$, $u_2(t)$ (рис. 5, 6) траектория системы игольчатая, поэтому этот тип управлений рекомендуется применять только в специальных случаях локальных задач управления.

Достаточно гибкими показали себя управления синтетического типа — управления классов $\mathcal{U}=$ Centre, Focus. В отличие от оптимальных, которые выражаются через тригонометрические функции и неэлементарную функцию, фокусные и центральные управления проще: они имеют один разрыв при $t = T/2$, на отрезке $[0, T/2]$ выражаются через тригонометрические, экспоненциальные функции, на отрезке $[T/2, T]$ управления постоянны. Амплитуды центрального и фокусного управлений монотонно убывают к нулю и в обоих этих случаях при $t < T$ система практически с нулевой скоростью, как и в оптимальном случае, входит в ε-окрестность точки x^1 (см. рис. 7, 10). Кроме того, управления этих типов зависят от нескольких параметров, что делает их гибкими. На рис. 7, 8 показаны управления класса Centre, вычисленные при двух различных наборах параметров. Управления классов $\mathcal{U}=$ Centre, Focus рекомендуется широко применять при умеренных начальных расстояниях $\mathrm{Dist}(x^0, x^1)$.

6. Заключение

В статье приводится компьютерная реализация вычислительного алгоритма и анализируется его работа. Программа FindControlLoc осуществляет пять стратегий управления системой в малой окрестности целевой точки. Чтобы эффективно решать глобальные задачи управления, потребуется строить эффективные смешанные стратегии, для чего потребуются методы параллельного программирования.

Список литературы

[1] Сачкова Е.Ф. *Приближенное решение задачи управления на основе нильпотентной аппроксимации* // Дифференциальные уравнения, 2009, № 10. ↑1, 2, 3, 2, 4.1

[2] Дьяконов В. Maple 6: учебный курс. — СПб.: Питер, 2001. — 608 с. ↑1

[3] Laumond J. P. // Lecture Notes in Control and Information Science, № 229: Springer, 1998. — 343 с. ↑1

[4] Гурман В. И. Принцип расширения в задачах оптимального управления.-2-ое изд., перераб. и доп. — М.: Наука, 1997. — 288 с. ↑1

[5] Дыхта В. А., Самсонюк О. Н. Оптимальное импульсное управление с приложениями. — М.: Физматлит, 2000. — 256 с. ↑1

[6] Аграчев А. А., Сачков Ю. Л. Геометрическая теория управления. — М.: Физматлит, 2005. — 392 с. ↑1, 2, 5.1

[7] Jurdjevic V. Geometric control theory. — Cambridge: University Press, 1997. ↑1, 5.1

[8] Сачкова Е.Ф. *Решение задачи управления для нильпотентной системы* // Дифференциальные уравнения, № 12, 2008, с. 1704–1707. ↑1, 2, 2, 3d, 4.1

[9] Сачкова Е.Ф. *Синтез управления для трехмерной нильпотентной системы* // Дифференциальные уравнения, принята к публикации. ↑1, 2, 2, 3d, 4.1

[10] Bellaiche A. *The tangent space in sub-Riemannian geometry.* // Sub-Riemannian Geometry, A. Bellaiche and J. J. Risler, Eds. — Basel, Swizerland: Birkhäuser, 1996, с. 1–78. ↑2

[11] Jean F. *Lectures on Dynamical and Contol Systems.* — Trieste, 2003. ↑3

[12] Уиттекер Э.Е. Аналитическая динамика. — М.: УРСС, 2004. ↑5.1

E. F. Sachkova. *Realization and analysis of algorithms for approximate solving the control problem* // Proceedings of Program Systems institute scientific conference "Program systems: Theory and applications". — Pereslavl-Zalesskij, v. **1**, 2009. — p. 59–76. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. We consider a computer realization of an algorithm for approximate solving the control problem for three-dimensional nonlinear systems governed by ordinary differential equations, linear in two controls. The algorithm is based on the method of nilpotent approximation. It is realized in Maple system and tested on the problem of orientation control of a sphere rolling on a plane.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009
УДК 519.87

С. А. Амелькин

Определение максимума термодинамической и экономической эффективности работы предприятия

Аннотация. В работе рассмотрено производственное предприятие, работающей в открытой экономической системе. Технологическое оборудование предприятия представляет собой тепловую машину. Требуется определить максимальное значение термодинамической (код тепловой машины) и экономической (рентабельности предприятия) эффективности производства мощности. Получены условия оптимальности для этой задачи.

1. Введение

Максимальная прибыль производственной фирмы может быть достигнута как за счет выбора цен (или объема выпуска товара) при наличии монополистической власти, так и за счет оптимальной организации технологического процесса. Задачи оптимального выбора технологического процесса сводятся к задаче определения минимальных издержек при заданном объеме производства. Решение этой задачи — кривая развития фирмы — позволяет ставить задачу оптимального ценообразования, позволяющего добиться максимума прибыли или предельного значения любого другого экономического критерия.

Подробно исследован случай, когда технологическая линия представляет собой тепломеханическую систему (например, тепловую машину). Исследования в этой области получили название «термоэкономика» [1]. Кривая производственных возможностей в этом случае — это кривая зависимости максимального КПД тепловой машины от ее мощности. На этой кривой выбираются точки, соответствующие максимуму прибыли при постоянных издержках (Рис. 1). На рисунке показана также ось, соответствующая прибыли от реализации мощности при постоянных ценах.

Такой подход не учитывает взаимосвязи между стоимостью ресурсов, требуемых для технологического процесса и параметрами этого процесса: затраты на потоки теплоты не связаны с параметрами

Работа выполнена при поддержке РФФИ, грант № 08-06-00141.

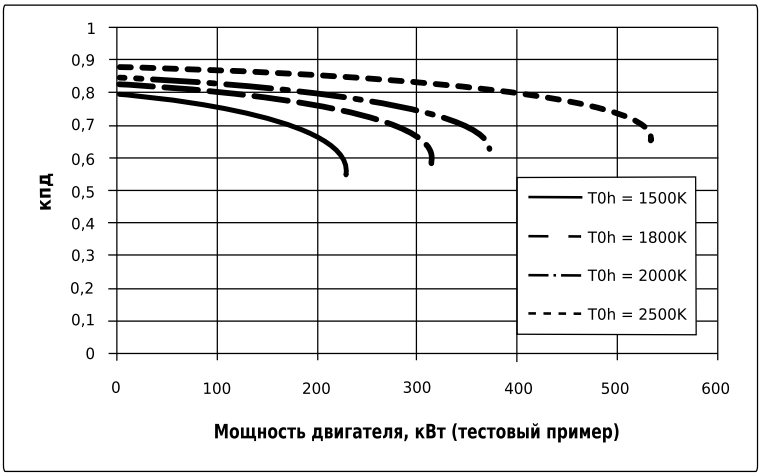


Рис. 1. Зависимость максимального КПД тепловой машины от ее мощности.

этих потоков (температурой источника, площади поверхности контакта рабочего тела с источником), а связаны только с интенсивностью потока, определяемого температурой рабочего тела. Далее рассмотрена задача максимизации термодинамической эффективности (КПД машины) и экономической эффективности (рентабельности производства). Эта задача соответствует микроэкономической задаче определения кривой развития фирмы, то есть зависимости минимальных издержек фирмы от производительности (мощности тепловой машины). Задача рассмотрена для случая, когда

(1) фирма может влиять на цены как целевого потока (мощности), так и потоков теплоты, поступающих и отводимых от рабочего тела;
(2) цены на ресурсы зависят не только от интенсивностей потоков, но и от параметров, входящих в уравнения кинетики.

2. Описание системы

Рассмотрим тепловую машину. Рабочее тело с распределенными параметрами контактирует с двумя источниками с постоянными температурами T_{0h} и T_{0l}. Температуры рабочего тела в контакте с

источниками T_{0h} и T_{0l} соответственно. Будем предполагать законы теплопередачи линейными

$$q_h = \alpha_h(T_{0h} - T_h); \quad q_l = \alpha_l(T_{0l} - T_l). \tag{1}$$

Тепловая машина за счет теплообмена с источниками вырабатывает мощность $n = q_h + q_l$. Предприятие, технологической линией в котором является тепловая машина, покупает теплоту и продает мощность. Предприятие на всех рынках обладает монополистической властью: цены на теплоту p_h, p_l и мощность p_n зависят от интенсивностей потоков. Цены p_h, p_l также зависят от параметров источников тепла, так что $p_h = p_h(q_h, T_{0h})$, $p_l = p_l(q_l, T_{0l})$. Целью предприятия является получить максимальную прибыль π от продажи мощности. Будем рассматривать стационарный режим: интенсивности потоков не изменяются во времени.

3. Алгоритм решения задачи

Задача определения оптимальных технологических параметров решается в три этапа.

На первом этапе для каждого значения n определяются такие величины температур рабочего тела T_{0h}, T_{0l} и потоков теплоты q_h, q_l, которые обеспечивают наибольший КПД тепловой машины. Для этого следует решить задачу минимальной диссипации

$$\sigma = \frac{q_h(T_{0h}, T_h)}{T_{0h}} + \frac{q_l(T_{0l}, T_l)}{T_{0l}} \to \min_{T_h, T_l} \tag{2}$$

при условии, вытекающем из энергетического баланса рабочего тела

$$q_h(T_{0h}, T_h) + q_l(T_{0l}, T_l) = n, \tag{3}$$

и условии энтропийного баланса рабочего тела

$$\frac{q_h(T_{0h}, T_h)}{T_{0h}} + \frac{q_l(T_{0l}, T_l)}{T_{0l}} = 0. \tag{4}$$

На втором этапе требуется найти минимальные издержки предприятия. При заданном значении мощности n и известной зависимости цены готовой продукции $p(n)$ доход предприятия $np(n)$ также задан. Поэтому задача о минимуме издержек идентична задаче о максимальной экономической эффективности — рентабельности предприятия.

$$c = p_h(q_h, T_{0h})q_h(T_{0h}, T_h^*) + p_l(q_l, T_{0l})q_l(T_{0l}, T_l^*) \to \min_{T_{0h}, T_{0l}}, \tag{5}$$

где T_h^*, T_l^* — решение задачи (2)–(4). На этом этапе не требуется учитывать условие (3), т. к. значения T_h^*, T_l^* определены с учетом этого условия и выражения для оптимальных значений этих температур зависят от n.

В задаче (5) учитываются только текущие издержки на приобретение теплоты. В случае если требуется учесть издержки на другие факторы производства (например, на труд, капитал в виде амортизации оборудования и пр.), выражения для этих факторов производства должны быть включены в критерий (5). Впрочем, если эти факторы производства не зависят от температур T_{0h}, T_{0l}, то задача (5) будет сепарабельной. Например, если амортизация оборудования (определяемая капитальными вложениями) c_k зависит от площадей контакта рабочего тела с источниками[1] $c_k = f(\alpha_h, \alpha_l)$, то, наряду с задачей (5), требуется решить задачу

$$f(\alpha_h, \alpha_l) \to \min_{\alpha_h, \alpha_l} \tag{6}$$

при условии либо сохранения общей площади теплообмена

$$\alpha_h + \alpha_l = A, \tag{7}$$

либо сохранения эквивалентного значения коэффициента теплопередачи

$$\frac{\alpha_h \alpha_l}{\alpha_h + \alpha_l} = \alpha_0. \tag{8}$$

Решение этих задач рассмотрено в [2]. Решая задачу минимума издержек при заданной мощности мы получаем решение задачи об эффективности производства — как технологической (КПД машины), так и экономической (рентабельность). В результате решения задач (2)–(4), (5) и (6) получаем зависимость $c(n)$, которая может быть использована на третьем этапе для решения задачи

$$\pi = p(n)n - c(n) \to \max_n . \tag{9}$$

Решение задачи (9) — стандартной для курса микроэкономики — приводит к требованию равенства предельных издержек предельному доходу [3]. Можно решать и другие задачи — например, о предельной мощности. При любом абсолютном критерии на этом этапе решение соответствует максимальным значениям КПД и рентабельности,

[1] Принимаем, что коэффициенты теплопередачи пропорциональны площади контакта с источниками. Тогда можно ограничение на общую площадь теплообмена заменить на требование постоянства суммы коэффициентов теплопередачи.

а, значит, режим работы предприятия относится к классу процессов минимальной диссипации и в термодинамическом, и в экономическом смысле. Рассмотрим задачи каждого этапа подробнее.

4. Задача о минимуме диссипации

Задача о минимуме диссипации при заданном целевом потоке (мощности) (2)—(4) решена в [4]. Там получена зависимость КПД тепловой машины от значения мощности и параметров линейных законов теплопередачи:

$$\eta = 1 - \frac{\alpha(T_{0h} + T_{0l}) - n - \sqrt{(n + \alpha(T_{0h} - T_{0l}))^2 - 4\alpha T_{0h} n}}{2\alpha T_{0h}}, \tag{10}$$

где $\alpha = \alpha_h \alpha_l / (\alpha_h + \alpha_l)$ — эквивалентный коэффициент теплопередачи. В ходе рассматриваемого этапа решения задачи о минимальной диссипации, нас интересуют оптимальные зависимости потоков теплоты от источников к рабочему телу.

Получим эти зависимости. Функция Лагранжа для задачи (2)–(4) имеет вид

$$L = L_h + L_l + \lambda n, \tag{11}$$

где

$$L_i = q_i(T_{0i}, T_i)\left(\frac{1}{T_{0i}} - \lambda + \frac{\mu}{T_i}\right), \quad i \in \{h, l\}. \tag{12}$$

Здесь λ, μ — неопределенные множители Лагранжа, соответствующие ограничениям (3), (4). Необходимые условия оптимальности $\partial L / \partial T_i = 0$ ($i \in \{h, l\}$) приводят к равенствам

$$T_i = T_{0i}\sqrt{\frac{\mu}{T_{0i}\lambda - 1}} = T_{0i}(1 - x_i(T_{0i})). \tag{13}$$

Обозначив $x_i(T_{0i}) = 1 - \sqrt{\dfrac{\mu}{T_{0i}\lambda - 1}}$ и подставив найденные значения T_i в условия (3), (4), получим систему уравнений

$$\begin{cases} \alpha_h T_{0h} x_h(T_{0h}) + \alpha_l T_{0l} x_l(T_{0l}) = n; \\ \alpha_h \dfrac{x_h(T_{0h})}{1 - x_h(T_{0h})} + \alpha_l \dfrac{x_l(T_{0l})}{1 - x_l(T_{0l})} = 0. \end{cases} \tag{14}$$

Выражая из первого уравнения (14) $x_l(x_h)$ и подставляя полученное выражение во второе уравнение (14), получаем квадратное уравнение

$$\alpha_h T_{0h} x_h^2 - \left[n + \frac{\alpha_h \alpha_l}{\alpha_h + \alpha_l}(T_{0h} - T_{0l}) \right] x_h + \frac{\alpha_l}{\alpha_h + \alpha_l} n = 0.$$

Введя эквивалентный коэффициент теплопередачи $\alpha = \dfrac{\alpha_h \alpha_l}{\alpha_h + \alpha_l}$, находим значения x_h и x_l, подставляем их в (13), получаем выражения для T_h^* и T_l^*. Найденные значения подставляем в уравнения кинетики (1) и, наконец, находим искомые выражения для теплоты:

$$q_h = \frac{1}{2}[n + \alpha(T_{0h} - T_{0l}) - \sqrt{D}], \quad q_l = \frac{1}{2}[n + \alpha(T_{0h} - T_{0l}) + \sqrt{D}], \tag{15}$$

где

$$D = (n + \alpha(T_{0h} - T_{0l}))^2 - 4\alpha T_{0h} n. \tag{16}$$

5. Задача о минимуме издержек

Зная зависимость потоков теплоты (15), (16) от значений T_{0h}, T_{0l}, можно выписать условия оптимальности для задачи (5)

$$\frac{\partial c}{\partial T_{0h}} = 0, \quad \frac{\partial c}{\partial T_{0l}} = 0 \Rightarrow$$

$$\begin{cases} \left(\dfrac{\partial p_h}{\partial q_h} q_h + p_h \right) \dfrac{\partial q_h}{\partial T_{0h}} + \left(\dfrac{\partial p_l}{\partial q_l} q_l + p_l \right) \dfrac{\partial q_l}{\partial T_{0h}} + \dfrac{\partial p_h}{\partial T_{0h}} q_h = 0, \\ \left(\dfrac{\partial p_h}{\partial q_h} q_h + p_h \right) \dfrac{\partial q_h}{\partial T_{0l}} + \left(\dfrac{\partial p_l}{\partial q_l} q_l + p_l \right) \dfrac{\partial q_l}{\partial T_{0l}} + \dfrac{\partial p_l}{\partial T_{0l}} q_l = 0. \end{cases} \tag{17}$$

Рассмотрим значения производных $\dfrac{\partial q_i}{\partial T_{0j}}$ $(i, j \in \{h, l\})$. Для этого представим выражения для потоков теплоты в виде

$$q_h = \frac{1}{2}(n + \kappa); \quad q_l = \frac{1}{2}(n - \kappa), \tag{18}$$

где $\kappa = \alpha(T_{0h} - T_{0l}) - \sqrt{D}$. Только κ зависит от значений T_{0h}, T_{0l}. Поэтому

$$\frac{\partial q_h}{\partial T_{0j}} = -\frac{\partial q_l}{\partial T_{0j}} \quad , j \in \{h, l\}, \tag{19}$$

а значит, уравнения (17) могут быть сведены к равенству

$$\frac{\frac{\partial p_h}{\partial T_{0h}} q_h}{\frac{\partial q_h}{\partial T_{0h}}} = -\frac{\frac{\partial p_l}{\partial T_{0l}} q_l}{\frac{\partial q_l}{\partial T_{0l}}}. \tag{20}$$

Надо учесть, что $q_h > 0$, $q_l < 0$, $\dfrac{\partial q_h}{\partial T_{0h}} < 0$, $\dfrac{\partial q_l}{\partial T_{0l}} < 0$. Последние два неравенства требуют пояснений, так как согласно (1) обе эти производные должны быть положительны. Однако, мы рассчитываем эти производные с учетом решения задачи (2)–(4). В этом случае $q_h = q_h(T_{0h}, T_{0l})$; $q_h = q_h(T_{0h}, T_{0l})$, эти зависимости определяются формулами (15)–(16). При увеличении температуры T_{0h} требуется меньший поток теплоты, чтобы обеспечить заданную мощность n; при увеличении температуры T_{0l} требуется большее значение интенсивности отвода теплоты от рабочего тела, а значит меньшее значение q_{0l}.

Из равенства (15) примем, что

$$\operatorname{sign}\left(\frac{\partial p_h}{\partial T_{0h}}\right) = \operatorname{sign}\left(\frac{\partial p_l}{\partial T_{0l}}\right).$$

Это также становится понятным, если учесть, что при $q_l < 0$ требуется, чтобы p_l было также отрицательным. Производные $\dfrac{\partial q_h}{\partial T_{0h}}$ и $\dfrac{\partial q_l}{\partial T_{0l}}$ связаны с интенсивностью потоков q_h и q_l следующим образом

$$\begin{aligned}
\frac{\partial q_h}{\partial T_0 h} &= \frac{1}{2}\alpha - \frac{1}{4\sqrt{D}}\left[2\alpha\left(n + \alpha(T_{0h} - T_{0l}) - 4\alpha n\right)\right] \\
&= \frac{\alpha}{2}\left[1 + \frac{n - \alpha(T_{0h} - T_{0l})}{\sqrt{D}}\right] = \frac{\alpha q_l}{\sqrt{D}}; \\
\frac{\partial q_l}{\partial T_0 l} &= \frac{1}{2}\alpha + \frac{1}{4\sqrt{D}}\left[-2\alpha\left(n + \alpha(T_{0h} - T_{0l})\right)\right] \\
&= \frac{\alpha}{2}\left[1 - \frac{n + \alpha(T_{0h} - T_{0l})}{\sqrt{D}}\right] = -\frac{\alpha q_h}{\sqrt{D}}.
\end{aligned} \tag{21}$$

С учетом (21) равенство (20) примет вид

$$\left(\frac{q_l}{q_h}\right)^2 = \frac{\partial p_l/\partial T_{0l}}{\partial p_h/\partial T_{0h}}, \tag{22}$$

откуда с учетом отрицательности величины q_l:

$$\frac{q_l}{q_h} = -\sqrt{\frac{\partial p_l/\partial T_{0l}}{\partial p_h/\partial T_{0h}}} \quad \text{или} \quad \eta = 1 + \frac{q_l}{q_h} = 1 - \sqrt{\frac{\partial p_l/\partial T_{0l}}{\partial p_h/\partial T_{0h}}}. \tag{23}$$

Уравнение (23) является условием оптимальности для выбора таких значений T_{0h}, T_{0l}, чтобы экономическая эффективность предприятия была бы наибольшей, а, значит, наименьшей — диссипация капитала [2].

Интересно, что для выбора функций спроса и предложения (кинетики ресурсообмена в экономической системе), определяющих зависимость цен от интенсивностей потоков и величин температур источников T_{0j} $j \in \{h, l\}$, удобно воспользоваться следующими зависимостями

$$p_j = v_j - \frac{p_j q_j(T_{0h}, T_{0l})}{T_{0j}}, \quad j \in \{h, l\}. \tag{24}$$

Такая зависимость связывает цену с потоком энтропии, отбираемым от j-го источника. К сожалению, расчет значений T_{0h}, T_{0l} для зависимостей (24), удовлетворяющих условиям (23), а значит и решение задач третьего этапа аналитически невозможно — требуется численный расчет для конкретных значений α, β_h, β_l, v_h, v_l, а также заданной функции спроса $p(n)$.

Список литературы

[1] Sieniutycz S., Salamon P. Finite-time thermodynamics and thermoeconomics. — London: Taylor & Francis, 1990. ↑1

[2] Миронова В. А., Амелькин С. А., Цирлин А. М. Математические методы термодинамики при конечном времени. — М.: Химия, 2000. ↑3, 5

[3] Пиндайк Р., Рубинфельд Д. Микроэкономика. — М.: Экономика. Дело, 1992. ↑3

[4] Цирлин А. М. Оптимальные циклы и циклические режимы. — М.: Энергоатомиздат, 1985. ↑4

Исследовательский центр системного анализа ИПС РАН

S. A. Amelkin. *Maximum of thermodynamic and economic efficiency of an industrial enterprise* // Proceedings of Program Systems institute scientific conference "Program systems: Theory and applications". — Pereslavl-Zalesskij, v. **1**, 2009. — p. 77–84. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. An industrial enterprise operating in an open economic system is considered. Technological equipment of the enterprise ia a heat engine. The problem at issue is to determine both thermodynamic and economic efficiencies of the enterprise. Optimality conditions for the problem are obtained.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 517.977.5

А. А. Ахременков, В. А. Казаков, А. М. Цирлин

Алгоритм оптимизации рынков электроэнергии как макросистем

Аннотация. Рассмотрена задача оптимизации торговли на взаимосвязанных энергетических рынках с учетом особенностей этих рынков. Получены её условия оптимальности и численный метод решения, основанный на их использовании. Показана возможность построения системы автоматической оптимизации.

1. Введение

Рынок электроэнергии включает некоторое число региональных рынков, соединенных между собой линиями межрегиональных передач. Каждый участник рынка (производитель или потребитель) входит в состав одного из региональных рынков, на котором все участники платят или получают плату за энергию по единой региональной цене. Как правило, региональные цены различаются между собой.

Торговый день на рынке энергии разделен на последовательность равных периодов, в каждом из них проводится «аукцион одного периода». Таким образом энергетический рынок представляет собой совокупность проходящих одновременно связанных региональных аукционов. Результаты аукциона зависят от ценовых заявок производителей-поставщиков энергии, состояния рынка перед началом аукциона (объемов региональных поставок и межрегиональных потоков), а также региональных спросов на энергию. Эти результаты определяют объемы поставок на рынок для каждого производителя энергии во всех регионах, объемы межрегиональных потоков, а также региональные цены на данный расчетный период.

Все производители энергии подают оператору свои ценовые заявки («предложения поставок») до начала аукциона. Эти предложения представляют собой ступенчатые возрастающие функции зависимости цены энергии от объемов ее поставки (рис. 1). Ценовые ступеньки на этих функциях могут быть (и часто действительно бывают) отрицательными, что соответствует согласию производителя оплачивать

Работа выполнена при поддержке РФФИ, грант № 08-06-00141.

покупателю поставляемую энергию в связи с тем, что ему невыгодно останавливать генераторы.

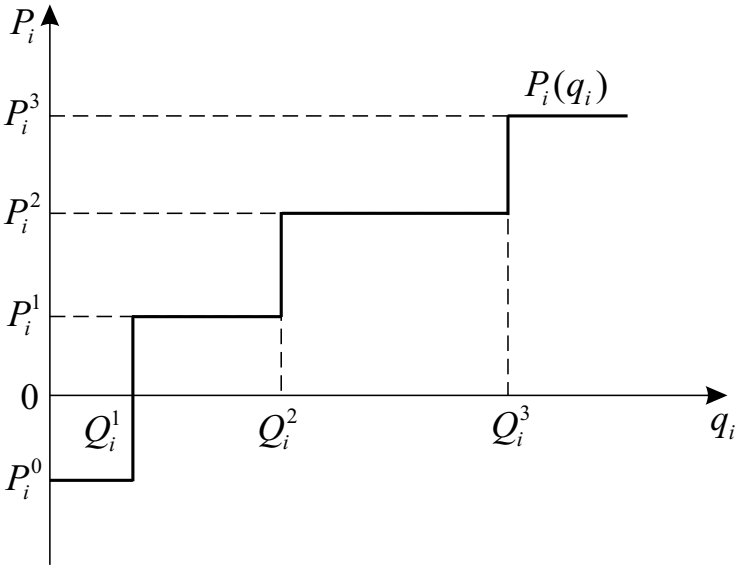


Рис. 1. Вид ценовых заявок производителей.

Региональный аукцион является аукционом единой цены, т. е. каждый из поставщиков в этом регионе получает единую цену за поставляемую энергию. Обычно предполагается, что рассматриваемый рынок является олигополистическим, где ценовые заявки участников определяют распределение поставок, текущие и региональные цены. Спрос предполагают не зависящим от цены (эластичность спроса близка к нулю). Ценовые заявки производителей для всех аукционов одного торгового дня подаются одновременно перед его началом. Количество ступеней заявок не должно превышать заданного числа (так, для Австралии это 10). Эти ценовые заявки используется на всех аукционах одного периода на протяжении данного торгового дня (см. [1, 2]).

Региональные цены и объемы поставок определяются не только спросом на электроэнергию и ценовыми заявками производителей, но и выбором обьемов энергии, поставляемым по линиям электропередач, соединяющим региональные рынки, потерями в этих линиях и

их возможностями. Целью каждого регионального аукциона является минимизация стоимости поставляемой энергии. Ниже рассмотрена задача оптимального распределения поставок между генераторами в рамках рынка электроэнергии.

Задачи оптимального распределения генерации электроэнергии на протяжении десятилетий находились в центре многих исследований, как академических, так и прикладных (см. [3, 4]). При этом стоимость генерации, являвшаяся целевой функцией в классических задачах оптимального распределения, как правило предполагалась непрерывной и непрерывно дифференцируемой. Изложенный ниже алгоритм учитывает влияние ступенчатого характера ценовых заявок на рыночную стоимость генерации и невыпуклость задачи распределения, связанную с тем, что заявки предполагают отрицательные цены.

Используемая модель является существенно упрощенной. В частности, в ней потоки энергии полагают скалярными величинами. Она упрощенно характеризует процессы в цепи электропередач несимметричной функцией потерь, параметры которой периодически уточняют по экспериментальным данным. В то же время эта модель описывает поведение рынка с точностью, необходимой для эффективного управления. В настоящее время задача оптимального распределения поставок и ценообразования на большенстве оптовых рынков электроэнергии решается на основе подобной модели. Как правило её сводят к задаче линейного программирования путем линеаризации целевой функции и ограничений с использованием соответствующих численных методов [1, 2]. Разрывы градиентов и невыпуклость задачи существенно снижают эффективность этих методов, а также могут приводить к остановке алгоритма в локальных экстремумах, не являющихся оптимальными решениями.

В данной работе изложены методы решения задачи оптимального распределения поставок энергии на многорегиональном аукционе, пригодные для преодоления как проблемы разрывов градиента целевой функции, так и невыпуклости задачи. Первоначально приведена математическая модель задачи, для нее получены необходимые условия оптимальности и алгоритм решения системы уравнений, вытекающих из необходимых условий; затем сформулированы достаточные

условия оптимальности, которые позволяют найти оценку снизу глобального экстремума, а также получить сравнительную оценку любого решения задачи оптимального распределния поставок относительно глобального экстремума; предложен способ построения системы с обратной связью для управления в реальном времени.

2. Аукцион одного периода. Задача оптимального распределения поставок

2.1. Постановка задачи

Рассматривается сеть из n взаимосвязанных региональных рынков электроэнергии (Рис. 2). В начале каждого торгового периода оператор получает следующую информацию:
1. Объединенную заявку на потребность в энергии каждого из регионов $d_i, \quad i = 1, ..., n$.
2. Объединенную региональную ценовую заявку $P_i(q_i)$ всех поставщиков энергии данного региона.
3. Объемы выработки в регионах $q_i(0)$.
4. Объемы межрегиональных потоков энергии за предшествующий период $g_{ij}(0)$.

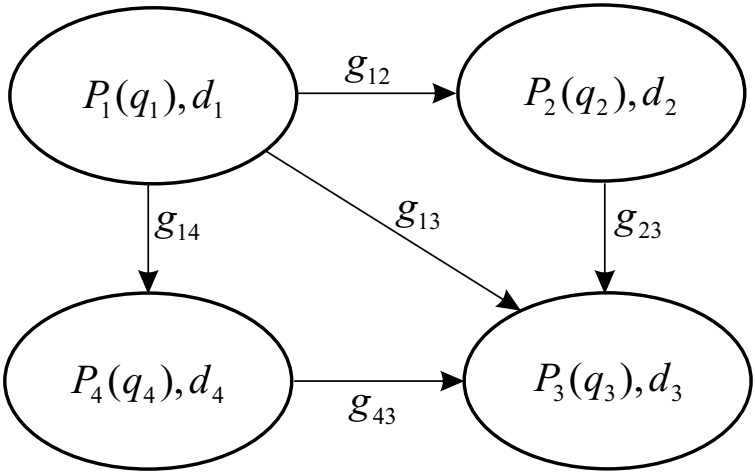


Рис. 2. Фрагмент сети региональных рынков.

Региональная ценовая заявка определяется как зависимость цены P_i от суммарного обьема энергии, который предложен всеми поставщиками данного региона по цене, не превышающей P_i. Так что

$$q_i = \sum_{\gamma=1}^{\gamma=G_i} \Delta_{i\gamma}, \quad \Delta_{i\gamma} = max\Delta / R_{i\gamma}(\Delta) \leq P_i. \tag{1}$$

Последнее равенство позволяет поставить в соответствие выбранному значению энергии, генерируемой поставщиками регионального рынка, объемы, закупаемые у каждого из них. Число региональных поставщиков G_i а $R_{i\gamma}(\Delta)$ – их ценовые заявки.

Функции $P_i(q_i)$ являются непрерывными справа,

$$\lim_{q_i \to Q_i^j} P_i(q_i) = P_i^j.$$

На объемы региональных поставок наложены автономные ограничения $q_i^{min} \leq q_i \leq q_i^{max}$. На межрегиональные объемы поставок g_{ij} из i-го в j-ый регион также наложены ограничения, $g_{ij}^{min} \leq g_{ij} \leq g_{ij}^{max}$. Они учитывают как технологические возможности поставщиков-генераторов по объемам выработки, так и их «динамические» возможности по скорости изменения объемов поставок. Отметим, что $g_{ij} = -g_{ji}$.

Предполагается, что передача объема g_{ij} из i-го в j-ый региональный рынок связана с потерями энергии. Эти потери могут быть охарактеризованы непрерывной монотонно возрастающей функцией $L_{ij}(g_{ij})$, для которой справедливы соотношения:

$$L_{ij}(g_{ij}) = L_{ji}(-g_{ij}) = L_{ji}(g_{ji}), \quad L_{ii}(x) \equiv 0, \quad L_{ij}(0) = 0, \quad L_{ij} > 0$$

для $g_{ij} \neq 0$ и $\frac{d^2 L_{ij}}{d^2 g_{ij}} > 0$ для $\forall g_{ij} \neq 0$. Характер зависимости $L_{ij}(g_{ij})$ от g_{ij} показан на Рис. 3.

Потери L_{ij} делятся между соответствующими региональными рынками в заданном соотношении и создают дополнительный спрос на энергию $\alpha_{ij} L_{ij}$ в i-ом регионе и $\alpha_{ji} L_{ij}$ в j-ом регионе. α_{ij} константы, $0 \leq \alpha_{ij} \leq 1$, $\alpha_{ji} = 1 - \alpha_{ij}$, $\alpha_{ii} = 0$.

Баланс энергии для такой сети описывается уравнением

$$\sum_{i=1}^{n} q_i = \sum_{i=1}^{n} d_i + \frac{1}{2} \sum_{i,j=1, i \neq j}^{n} L_{ij}(g_{ij}). \tag{2}$$

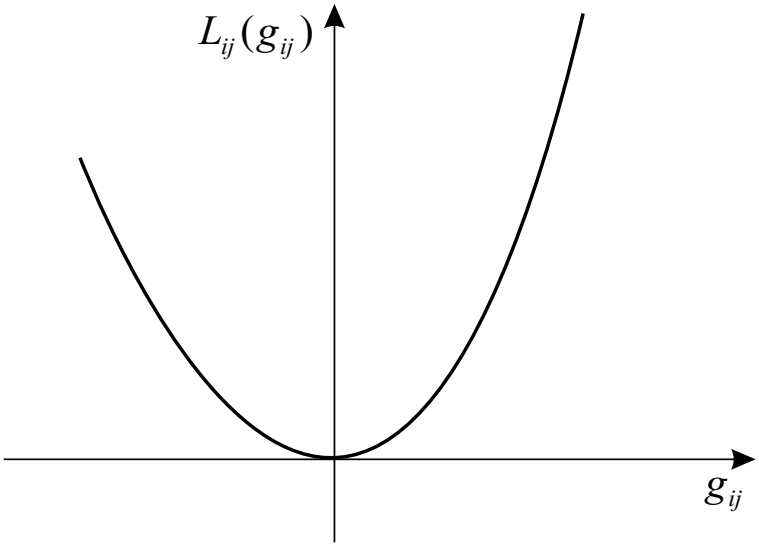


Рис. 3. Зависимость потерь энергии L_{ij} от величины межрегиональных потоков g_{ij}.

Поскольку $g_{ij} = -g_{ji}$ и $\alpha_{ij} + \alpha_{ji} = 1$, $\forall i, j = 1, \ldots, n$ баланс сети выполняется, если выполнены региональные балансы

$$q_i = d_i + \sum_{j=1}^{n} \big(g_{ij} + \alpha_{ij} L_{ij}(g_{ij})\big), \quad i = 1, \ldots, n. \tag{3}$$

Наряду с объединенной ценовой заявкой поставщиков i-го рынка введем объединенную стоимостную заявку $C_i(q_i)$, как зависимость стоимости энергии всех поставщиков от ее объема.

Стоимость генерации на i-ом рынке описывается соотношением

$$C_i(q_i) = \int_0^{q_i} P_i(x)dx. \tag{4}$$

Для непрерывной ценовой заявки $P_i(x)$

$$\frac{dC_i}{dq_i} = P_i(q_i). \tag{5}$$

Так как $P_i(q_i)$ — монотонные, кусочно-постоянные неубывающие функции, непрерывные справа, то $C_i(q_i)$ являются выпуклыми, кусочно-линейными, их производная в точке излома равна ее большему значению. Аналогичная стоимостная заявка может рассматриваться и для каждого поставщика на региональном рынке.

Общая стоимость генерации всеми поставщиками на рынке электроэнергии определяется как

$$I(d_1, \ldots, d_n, q_i, \ldots, q_n) = \sum_{i=1}^{n} C_i(q_i). \tag{6}$$

Оптимальное распределение поставок соответствует минимуму целевой функции I по q_i, g_{ij}

$$I(d_1, \ldots, d_n, q_i, \ldots, q_n) \to \min_{q_i, g_{ij}} \tag{7}$$

с учетом автономных ограничений

$$q_i^{min} \le q_i \le q_i^{max}, \quad i = 1, \ldots, n, \tag{8}$$

$$g_{ij}^{min} \le g_{ij} \le g_{ij}^{max} \quad j = 1, \ldots, n, \qquad i = 1, \ldots, n \tag{9}$$

и балансов региональных рынков (3). Последние соотношения включают нелинейные функции, что приводят к невыпуклости задачи.

В дальнейшем будем называть эту задачу «Задачей оптимального распределения поставок для аукциона одного периода». Обозначим составляющие ее решения как q_i^*, g_{ij}^*, а минимум целевой функции (минимальную общую стоимость поставки) как

$$I^*(d_1, \ldots, d_n) = I(d_1, \ldots, d_n, q_i^*, \ldots, q_n^*, g_{ij}^*). \tag{10}$$

Стоимость поставки I и ее минимальная величина I^* зависят не только от d_i, но и от ценовых заявок $P_i(q_i)$, а также от предаукционного состояния рынка — текущих значений $q_i(0)$ и $g_{ij}(0)$ (через автономные ограничения, наложенные на q_i и g_{ij}).

Неизвестными величинами в задаче оптимального распределения поставок (7), (8), (9), (3) являются региональные объемы поставок и объемы межрегиональных потоков. Если в сети оптового рынка все региональные рынки связаны между собой линиями передач, то общее число неизвестных задачи равно

$$\frac{n^2 - n}{2} + n = \frac{n(n+1)}{2}, \quad n \ge 0.$$

Первое слагаемое этого выражения соответствует числу потоков g_{ij}, а второе — числу региональных поставщиков.

Поскольку эти переменные должны удовлетворять n уравнениям баланса (3), то число свободных переменных в задаче оптимального распределения поставок равно числу межрегиональных потоков энергии $\frac{n(n-1)}{2}$ для сети, где все региональные рынки связаны между собой. Для $n = 1$ получается единственная свободная переменная, для $n = 3$ — три, и т.д.

2.2. Необходимые условия оптимальности

Исключив по условиям (3) региональные объемы поставок энергии, задачу оптимального распределения поставок (7), (8), (3) можно записать как задачу нелинейного программирования

$$\sum_{i=1}^{n} C_i \Big[d_i + \sum_{j=1, i\neq j}^{n} \big(g_{ij} + \alpha_{ij} L_{ij}(g_{ij}) \big) \Big] \to \min_{g_{ij}} \tag{11}$$

при выполнении следующих неравенств

$$q_i^{min} \leq d_i + \sum_{j=1, i\neq j}^{n} \big(g_{ij} + \alpha_{ij} L_{ij}(g_{ij}) \big) \leq q_i^{max} \tag{12}$$

и автономных ограничений

$$g_{ij}^{min} \leq g_{ij} \leq g_{ij}^{max} \quad i, j = 1, \ldots, n. \tag{13}$$

Независимыми переменными задачи являются объемы межрегионального обмена энергией g_{ij}. Региональные выработки (поставки) рассчитывают из уравнений баланса (3). Обозначим ее решение как g_{ij}^*. Это решение является вектор-функцией потребления $d_1, \ldots, d_n$.

Автономные ограничения выполнены, расчет производится в пределах интервала непрерывности ценовой заявки. Начинаем рассмотрение со случая, когда все автономные ограничения не активны, т.е. в точке оптимума

$$q_i^{min} < d_i + \sum_{j=1, i\neq j}^{n} \big(g_{ij} + \alpha_{ij} L_{ij}(g_{ij}) \big) < q_i^{max} \tag{14}$$

и

$$g_{ij}^{min} < g_{ij} < g_{ij}^{max}. \tag{15}$$

Допустим, что q_i - точки, где функции $P_i(q_i)$ непрерывны. Тогда из условия стационарности I по g_{ij} получаем для двух взаимосвязанных регионов:

$$\frac{\partial I}{\partial g_{ij}} = 0 \Rightarrow \frac{dC_i}{dq_i}\big(1 + \alpha_{ij}\frac{dL_{ij}}{dg_{ij}}\big) = -\frac{dC_j}{dq_j}\big(-1 + \alpha_{ji}\frac{dL_{ij}}{dg_{ij}}\big). \tag{16}$$

Здесь принято во внимание, что $g_{ij} = -g_{ji}$. С учетом (4) получаем

$$\big(1 + \alpha_{ij}\frac{dL_{ij}}{dg_{ij}}\big)P_i^* = \big(1 - \alpha_{ji}\frac{dL_{ij}}{dg_{ij}}\big)P_j^*. \tag{17}$$

Здесь $P_i^* = P_i(q_i^*)$ и $P_j^* = P_j(q_j^*)$ — цены, соответствующие выбранным суммарным объемам поставок в ценовых заявках для регионов i и j соответственно.

Назовем в этих уравнениях величины, стоящие в левой и правой частях равенства (17), *скорректированными ценами* в одном регионе относительно другого, связанного с ним прямой межрегиональной линией передачи энергии. Для i-го региона по отношению к j-ому это

$$\overline{P}_{ij} = \big(1 + \alpha_{ij}\frac{dL_{ij}}{dg_{ij}}\big)P_i^*, \tag{18}$$

а для j-го по отношению к i-му соответственно

$$\overline{P}_{ji} = \big(1 - \alpha_{ji}\frac{dL_{ij}}{dg_{ij}}\big)P_j^*.$$

Следовательно, потоки между двумя регионами в режиме, который соответствует неактивным автономным ограничениям и участкам непрерывности функций ценовых заявок, могут быть оптимальны в том и только в том случае, если скорректированные цены в этих регионах равны

$$\overline{P}_{ij} = \overline{P}_{ji}. \tag{19}$$

Генерирование в точках скачка цены на ценовой заявке. Допустим, что генерированию в объеме q_j соответствует точка на ценовой заявке, где цена изменяется скачком. Тогда пределы скорректированных цен слева и справа от нее равны $\overline{P}_{ji} = \lim_{\epsilon\to 0} P_j(q_j + \epsilon)\big(1 - \alpha_{ji}L'_{ij}\big)$, $\overline{P}^-_{ji} = \lim_{\epsilon\to 0} P_j(q_j - \epsilon)\big(1 - \alpha_{ji}L'_{ij}\big)$, и $\overline{P}_{ij} < \overline{P}_{ji}$ (см. Рис. 4). Таким образом, отрицательная вариация $\delta g_{ij} < 0$ увеличивает цену поставки $\delta I = (\overline{P}_{ij} - \overline{P}_{ji})\delta g_{ij} > 0$.

Бесконечно малая правая вариация $\delta g_{ij} > 0$ может привести к двоякому результату. Если предел слева для приведенной цены на j-ом рынке все еще выше, чем на i-ом, $\overline{P}_{ij} < \overline{P}^{-}_{ji}$, то цена поставки испытает скачок, который приведет к положительному изменению $\delta I = (\overline{P}_{ji} - \overline{P}^{-}_{ji})\delta g_{ij}$. После чего можно продолжать улучшение критерия I увеличением потока g_{ij} (см. выше).

Если $\overline{P}_{ij} > \overline{P}^{-}_{ji}$, то никакие изменения g_{ij} не могут улучшить I. Таким образом, условие оптимальности для случая, когда автономные ограничения не активны, а один из генераторов работает при нагрузке, соответствующей скачку на функции ценовой заявки, имеет вид

$$\overline{P}^{-}_{ji} < \overline{P}_{ij} < \overline{P}_{ji}. \tag{20}$$

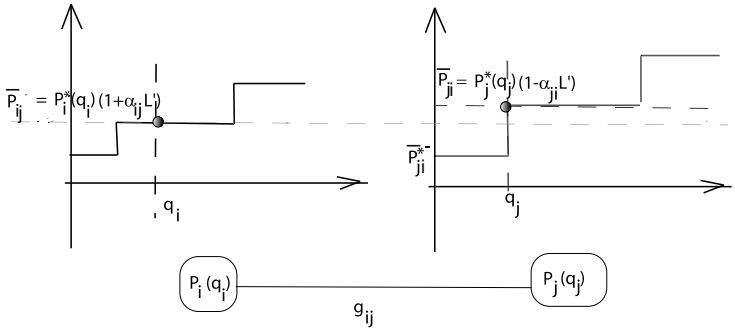


Рис. 4. Вид распределения, которое невозможно улучшить изменением межрегиональных потоков g_{ij}.

Общие условия оптимальности для неактивных автономных ограничений. Аналогичный анализ условий, при которых гарантировано неулучшение целевой функции I путем вариации межрегиональных потоков, приводит к следующим общим условиям оптимальности задачи (11)–(13) при неактивных автономных ограничениях

$$\begin{aligned} &\overline{P}^{+}_{ij} \le \overline{P}^{-}_{ji} \quad \overline{P}^{-}_{ij} \ge \overline{P}^{+}_{ji}, \quad g^{min}_{ij} < g_{ij} < g^{max}_{ij}, \\ &q^{min}_{i} < d_i + \textstyle\sum_{j=1, i\neq j}^{n} \big(g_{ij} + \alpha_{ij} L_{ij}(g_{ij})\big) < q^{max}_{i}. \end{aligned} \tag{21}$$

Здесь предполагается, что левый и правый пределы приведенных цен описываются как

$$\overline{P}^{+}_{ab} = \lim_{\epsilon \to 0} P_{ab}(q_i + \epsilon), \quad \overline{P}^{-}_{ab} = \lim_{\epsilon \to 0} P_{ab}(q_i - \epsilon).$$

Необходимые условия оптимальности для случая активных автономных ограничений. Если $g_{ij} = g^{max}_{ij}$, или $q_i = q^{max}_i$ или $q_j = q^{min}_j$, то допустимы только отрицательные изменения $\delta g_{ij} < 0$, и условия неулучшаемости I при таких изменениях имеют вид

$$\overline{P}_{ij} \geq \overline{P}_{ji}. \tag{22}$$

Аналогично, если $g_{ij} = g^{min}_{ij}$, или $q_i = q^{max}_i$ или $q_j = q^{min}_j$, то распределение является оптимальным только в случае, когда

$$\overline{P}_{ij} \leq \overline{P}_{ji}. \tag{23}$$

Это позволяет сформулировать следующее У т в е р ж д е н и е:

Если $\{g^*_{ij}\}$ оптимальное решение задачи (11)–(13), то выполнены условия

Нестрогие двусторонние ограничения

$$\begin{gathered} g^{min}_{ij} < g_{ij} < g^{max}_{ij}, \\ q^{min}_i < d_i + \textstyle\sum_j \big(g_{ij} + \alpha_{ij} L_{ij}(g_{ij})\big) < q^{max}_i, \\ q^{min}_j < d_j + \textstyle\sum_i \big(-g_{ij} + (1-\alpha_{ij}) L_{ij}(g_{ij})\big) < q^{max}_j, \\ \overline{P}^{+}_{ij} \geq \overline{P}^{-}_{ji}, \qquad \overline{P}^{-}_{ij} \leq \overline{P}^{+}_{ji}. \end{gathered} \tag{24}$$

Ограничения на потоки

$$\begin{gathered} g_{ij} = g^{max}_{ij}, \\ q^{min}_i < d_i + \textstyle\sum_j \big(g_{ij} + \alpha_{ij} L_{ij}(g_{ij})\big) < q^{max}_i, \\ q^{min}_j < d_j + \textstyle\sum_i \big(-g_{ij} + (1-\alpha_{ij}) L_{ij}(g_{ij})\big) < q^{max}_j, \\ \overline{P}^{-}_{ij} \leq \overline{P}^{+}_{ji}, \end{gathered} \tag{25}$$

или

$$\begin{gathered} g_{ij} = g^{min}_{ij}, \\ q^{min}_i < d_i + \textstyle\sum_j \big(g_{ij} + \alpha_{ij} L_{ij}(g_{ij})\big) < q^{max}_i, \\ q^{min}_j < d_j + \textstyle\sum_i \big(-g_{ij} + (1-\alpha_{ij}) L_{ij}(g_{ij})\big) < q^{max}_j, \\ \overline{P}^{+}_{ij} \geq \overline{P}^{-}_{ji}. \end{gathered} \tag{26}$$

Строгие ограничения (А)

$$g_{ij}^{min} < g_{ij} < g_{ij}^{max}, \quad \overline{P}_{ij}^{-} \le \overline{P}_{ji}^{+}, \tag{27}$$

$$q_i^{min} = d_i + \sum_j \big(g_{ij} + \alpha_{ij} L_{ij}(g_{ij})\big), \quad \text{или}$$
$$q_j^{max} = d_j + \sum_j \big(-g_{ij} + (1-\alpha_{ij}) L_{ij}(g_{ij})\big)$$

или (В)

$$g_{ij}^{min} < g_{ij} < g_{ij}^{max}, \qquad \overline{P}_{ij}^{+} \ge \overline{P}_{ji}^{-}, \tag{28}$$

$$q_i^{max} = d_i + \sum_j \big(g_{ij} + \alpha_{ij} L_{ij}(g_{ij})\big), \quad \text{или}$$
$$q_j^{min} = d_j + \sum_j \big(-g_{ij} + (1-\alpha_{ij}) L_{ij}(g_{ij})\big).$$

Многорегиональные условия оптимальности. Рассмотрим линейную часть сети, состоящую из трех региональных рынков. Допустим, что

$$\overline{P}_{ij} < \overline{P}_{ji}, \qquad \overline{P}_{jk} < \overline{P}_{kj}, \qquad q_j^{*} = q_j^{max}.$$

Поток энергии направлен из i-го в j-ый и далее из j-го в k-ый рынок. Автономное ограничения на объем генерации на j-ом рынке активно, что исключает увеличение g_{ij}. Однако две одновременные положительные вариации $\delta g_{ij} = \delta g_{jk} > 0$ допустимы. Анализ, аналогичный проведенному выше, для фрагмента сети, состоящего из соединения двух региональных рынков, приводит к следующим условиям оптимальности для участков непрерывности функций $P_i(.)$ и $P_k(.)$ при неактивных ограничениях на q_i и q_k:

$$P_i\big(1-\alpha_{ij}L'_{ij}\big)\big(1-\alpha_{jk}L'_{jk}\big) = P_k\big(1+\alpha_{ij}L'_{ij}\big)\big(1+\alpha_{jk}L'_{jk}\big) \tag{29}$$

и к общим условиям оптимальности для точек скачкообразного изменения цены на функциях $P_i(.)$ и $P_k(.)$

$$\begin{aligned} P_i^{+}\big(1-\alpha_{ij}L'_{ij}\big)\big(1-\alpha_{jk}L'_{jk}\big) &\le P_k^{-}\big(1+\alpha_{ij}L'_{ij}\big)\big(1+\alpha_{jk}L'_{jk}\big), \\ P_i^{-}\big(1-\alpha_{ij}L'_{ij}\big)\big(1-\alpha_{jk}L'_{jk}\big) &\ge P_k^{+}\big(1+\alpha_{ij}L'_{ij}\big)\big(1+\alpha_{jk}L'_{jk}\big). \end{aligned} \tag{30}$$

Для активных автономных ограничений эти локальные условия оптимальности имеют тот же вид, что и условия оптимальности для двухрегиональной системы, полученные выше, однако в этих условиях фигурируют многорегиональные приведенные цены (которые определены левыми и правыми частями равенств (29)) вместо локальных приведенных цен (18). Эти условия устанавливают, что в задаче об оптимальном распределении два региональных рынка, связанных между собой в сети через несколько региональных рынков с активными ограничениями на генерирование, могут иметь различные

многорегиональные приведенные цены только в том случае, если по меньшей мере один из потоков между ними, либо объем генерации на одном из рынков вышел на активное ограничение.

3. Проблема определения глобального минимума в задаче оптимального распределения поставок энергии

Одной из главных особенностей задачи оптимального распределения поставок (7), (8), (9), (3) является невыпуклость ее целевой функции из-за (возможного) наличия отрицательных ступеней на функциях ценовых заявок и как следствие ее многоэкстремальность. Таким образом, при использовании полученных выше необходимых условий оптимальности, или при использовании прямого поиска минимума стоимости поставки из произвольного начального приближения не гарантируется определение глобального минимума целевой функции. Следовательно, в этих случаях необходимо проверять, является ли полученное решение глобальным или локальным экстремумом, и если минимум оказывается локальным, то сильно ли он отличается от глобального и какое улучшение целевой функции в единицах стоимости поставки возможно при дальнейшем продолжении поиска.

Рассмотрим расширение задачи оптимального распределения (7), (8), (9), (3) за счет исключения автономных ограничений (9) и уравнений связи (3). Она приобретает вид

$$I(d_1,\dots,d_n,q_i,\dots,q_n)=\sum_{i=1}^{n}C_i(q_i)\to\min_{q_i} \tag{31}$$

при условиях

$$\sum_i q_i = M, \tag{32}$$

$$q_i^{min}\le q_i\le q_i^{max},\quad i=1,\dots,n. \tag{33}$$

Здесь M — параметр, равный суммарным потерям энергии при межрегиональных передачах, на который наложены автономные ограничения

$$\sum_i q_i^{min}\le\sum_i d_i\le M\le\sum_i q_i^{max}. \tag{34}$$

Введем в рассмотрение функцию Беллмана (см. [6]), используя следующее рекуррентное соотношение

$$\phi_1(x_1) = C_1(x_1), \quad \phi_2(x_2) = \min_{q_2^{min} \le q_2 \le q_2^{max}} \big[C_2(q_2) + \phi_1(x_2 - q_2)\big], \tag{35}$$

$$\phi_\nu(x_\nu) = \min_{q_\nu^{min} \le q_\nu \le q_\nu^{max}} \big[C_2(q_2) + \phi_\nu(x_\nu - q_\nu)\big].$$

Соотношение

$$\phi_n(M) = \min_{q_1, q_2, \ldots, q_n} \sum_i C_i(q_i^*(M)) \tag{36}$$

позволяет определить глобальный минимум целевой функции и соответствующие ему потери в сети, отвечающие уравнениям энергетического баланса

$$\sum_i d_i + \frac{1}{2} \sum_{ij} L_{ij}(g_{ij}) = M. \tag{37}$$

Полученное решение определяет нижнюю достижимую границу стоимости поставки для заданных потерь энергии в сети (37). Пусть на основе полученных выше условий оптимальности рассчитано оптимальное решение и получены значения g_{ij}^*, q_i^* и I^*. В этом случае соотношения (37) позволяют рассчитать M^* и затем решить задачу (31), (32), (33) для полученного M^*.

Если окажется, что решение $I^* = \phi_n(M^*)$, то g_{ij}^* и q_i^* является глобальным минимумом. Если величина $|I^* - \phi_n(M^*)|$ является достаточно малой, то можно закончить поиск и рассматривать полученный минимум как удовлетворительное решение.

4. Оптимизация поставок энергии с использованием автоматической системы управления

На практике оператор рынка использует предсказывающее управление, т.е. он предсказывает спрос, рассчитывает соответствующие ему распределения поставок и затем информирует поставщиков об их квотах. В результате в системе возникают значительные помехи, вызванные разницей между текущими и расчетными квотами, которые не только приводят к ошибкам при управлении рынком, но при определенных условиях могут привести к его неустойчивости. На некоторых рынках эта проблема усугубляется тем, что контроль за состоянием рынка производится значительно чаще, чем аукционы.

Так, в Австралии контроль осуществляют каждые 5 минут, а аукционы, на которых определяют цены на очередной расчетный период и платежи за поставки энергии, проводят с интервалом 30 минут. В итоге текущие цены и объемы поставок на рынке определяются с большими погрешностями, что отрицательно сказывается на экономической эффективности управления рынком.

Полученные выше условия оптимальности могут быть использованы для создания непрерывных автоматических систем управления рынком электроэнергии, в которых последовательность периодических аукционов можно заменить одним непрерывным аукционом. Это позволит значительно снизить уровень ошибок при расчете оптимального распределения поставок.

Рассмотрим сеть региональных рынков, как экономическую макросистему [7], состоящую из подсистем (региональных рынков), которые обмениваются ресурсом — энергией q_i. Система является открытой, т.к. она получает внешние потоки заявок от потребителей и производителей энергии. В каждой подсистеме имеется своя оценка энергии P_j , которая в свою очередь зависит от объемов обмена между подсистемами g_{ij}. Чем больше g_{ij} (g_{ij} считается положительным, если энергия передается из i-го в j-ый региональный рынок), тем меньше эта оценка. Величина объема g_{ij} зависит от разницы оценок энергии в i-ом и j-ом региональных рынках таким образом, что она равна нулю, если $P_i = P_j$ и

$$\operatorname{sign}[g_{ij}(P_i, P_j)] = \operatorname{sign}[P_i - P_j]. \tag{38}$$

Потоки энергии в такой системе направлены в сторону регионов с более высокой энергетической оценкой, что приводит к снижению последних и сближению оценок на всех рынках в установившемся режиме. Когда энергия передается в j-ый региональный рынок ($q_{ij} > 0$), то количество энергии q_j, генерируемое на этом рынке, уменьшается и соответственно уменьшается стоимость генерации C_j и наивысшая ценовая ступенька P_j на ценовой заявке. Если $q_{ij} < 0$, то C_i и P_i возрастают. Нулевые потери при передаче энергии соответствуют случаю бесконечно большой «проводимости» (отсутствии потерь и ограничений на объемы поставок) линии. Например, если передача энергии описывается линейным законом

$$g_{ij}(P_i, P_j) = \beta_{ij}(P_i - P_j), \tag{39}$$

то нулевые потери соответствуют $\beta_{ij} = \infty$. В этом случае макросистема превращается в гомогенную и в равновесии $P_i = P_j$.

Отметим, что (17) можно записать в виде

$$\phi_{ij}(g_{ij}) = \frac{P_i}{P_j}, \quad i,j = 1,\dots,n, \quad i \neq j, \quad g_{ij} = -g_{ji}. \tag{40}$$

Таким образом, состояние системы, в которой потоки энергии описываются уравнением (39) и условием $\beta_{ij} \to \infty$, q_i определяется соотношением (17) и автономными ограничениями (12), а установившееся значение P_i^s совпадает с решением задачи оптимизации (7), (8).

Видоизменим кинетические функции $\tilde{g}_{i,j}(P_i, P_j)$, так, чтобы для них стационарное состояние системы совпадало с решением задачи (7), (8). Из (40) следует, что задача определения такой кинетической функции сводится к решению следующего функционального уравнения

$$\phi_{ij}(\tilde{g}_{ij}(P_i, P_j)) = \frac{P_i}{P_j} \tag{41}$$

относительно зависимостей $\tilde{g}_{ij}$.

Если обозначить

$$L'(g_{ij}) \equiv \frac{dL_{ij}}{dg_{ij}},$$

то уравнение (41) можно записать в следующей форме

$$\frac{1-(1-\alpha_{ij})L'}{1+\alpha_{ij}L'} = \frac{P_i}{P_j}. \tag{42}$$

Получим

$$L'(g_{ij}) = \frac{P_j - P_i}{\overline{P}_{ij}}, \quad \overline{P}_{ij} \equiv \alpha_{ij}P_i + \alpha_{ji}P_j. \tag{43}$$

Функцию потерь часто аппроксимируют линейной или квадратичной функцией. Рассмотрим их в качестве примера.

1. Предположим

$$L_{ij}(g_{ij}) = \beta_{ij} g_{ij}^2,$$

тогда из (43) следует

$$g_{ij}(P_i, P_j) = \frac{1}{2\beta_{ij}\overline{P}_{ij}}\big(P_j - P_i\big). \tag{44}$$

2. Предположим

$$L_{ij}(g_{ij}) = \begin{cases} r_{ij}^{+} g_{ij} + \beta_{ij}^{+} g_{ij}^2 & \text{если} \quad g_{ij} > 0, \\ L_{ij}(g_{ij}) = 0 & \text{если} \quad g_{ij} = 0, \\ r_{ij}^{-} g_{ij} + \beta_{ij}^{-} g_{ij}^2 & \text{если} \quad g_{ij} < 0, \end{cases} \tag{45}$$

здесь $r_{ij}^{+} > 0$ и $r_{ij}^{-} < 0$. Тогда из (43) следует, что (Рис. 5)

$$g_{ij}(P_i, P_j) = \frac{P_j - P_i - r_{ij}^{+}\overline{P}_{ij}}{2\beta_{ij}\overline{P}_{ij}} \qquad \text{для} \quad P_j - P_i > r_{ij}^{+}\overline{P}_{ij}, \tag{46}$$

$$g_{ij}(P_i, P_j) = \frac{P_j - P_i - r_{ij}^{-}\overline{P}_{ij}}{2\beta_{ij}\overline{P}_{ij}} \qquad \text{для} \quad P_j - P_i < -|r_{ij}^{-}|\overline{P}_{ij}, \tag{47}$$

$$g_{ij}(P_i, P_j) = 0, \qquad \text{для} \quad r_{ij}^{+}\overline{P}_{ij} < P_j - P_i < -|r_{ij}^{-}|\overline{P}_{ij}. \tag{48}$$

Таким образом, если разница в ценах на двух рынках мала, то нецелесообразно передавать энергию между ними.

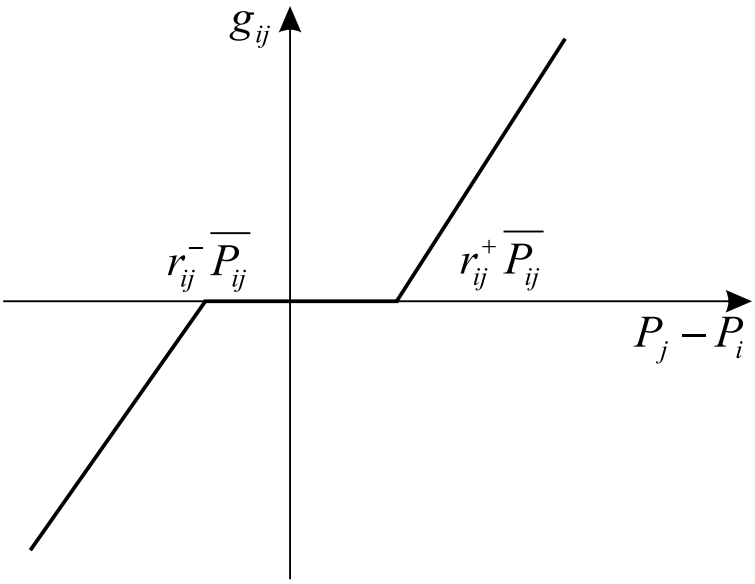


Рис. 5. Характер зависимости оптимального потока g_{ij} от разности наивысших цен поставки на двух региональных рынках.

Если получена зависимость $\tilde{g}_{ij}(P_i, P_j)$ или ее аппроксимация для всех i, j, то решение задачи распределения поставок значительно

упрощается, поскольку она распадается на отдельные задачи управления потоками в реальном времени:

— измерение заявок $\{d_1(t), d_2(t), \ldots, d_n(t)\}$ в реальном времени,

— пересчет региональных цен $P_i(q_i(t))$,

— изменение потоков в соответствии с правилом $\tilde{g}_{ij}(P_i, P_j)$.

Такой подход позволяет получить решение задачи распределения поставок в форме оптимального синтеза, реализацией которого является автоматическая система управления с обратной связью.

Список литературы

[1] Chong-White C. Investigation into aspects of Australian national electricity market central dispatch algorithm: MS Thesis, UTS, 2001. ↑1

[2] Gillett R. Pre-dispatch process description: NEMMCO, 1998. ↑1

[3] Schweppe F. C., Caramanis M. C., Tabors R. D., Bohn R. E. Spot Pricing of Electricity. — Norwell, MA: Kluwer Academic Publishers, 1988. ↑1

[4] Chowdhury B. H., Rahman S. A review of recent advances in economic dispatch. — T. **5**, № 4: IEEE Trans. Pow. Syst., 1990. — 1248 с. ↑1

[5] Guan X., Ni E., Luh P. B., Ho Y. Optimization-based bidding strategies for deregulated electricity power markets in Hobbs, B.F. et al. (eds.): The Next Generation Of Electric Power Unit Commitment Models. — Boston, MA: Kluwer Academic, 2001. ↑

[6] Беллман Р. Динамическое программирование. — М.: ИЛ, 1960. ↑3

[7] Цирлин А. М. Математические модели и оптимальные процессы в макросистемах. — М.: Наука, 2003. ↑4

5. Приложение

Вычислительный алгоритм.

Алгоритм начинает работу с определения любого допустимого решения, т.е. потоков g_{ij}, при которых выполняются все автономные ограничения — на объемы потоков (34) и региональных генераций (12). Затем это решение последовательно улучшается с помощью следующей вычислительной процедуры:

(1) рассчитывают разницу приведенных цен $\Delta\overline{P}_{ij} = \overline{P}_{ij} - \overline{P}_{ji}$ для каждой пары регионов, связанных между собой;

(2) отмечают все те пары, для которых выполнено одно из условий оптимальности ;

(3) для каждой не отмеченной пары, если $\overline{P}^{-}_{ij} > \overline{P}^{+}_{ji}$ то $g^{min}_{ij} < g_{ij} < g^{max}_{ij}$, $q^{max}_i < q_i(g_{ij}) < q^{max}_i$, $q^{max}_j < q_j(g_{ij}) < q^{max}_j$, то g_{ij} увеличивается до тех пор, пока одно из этих неравенств не обращается в равенство. При таком увеличении происходит переход через ступени

ценовой заявки, в результате которого гарантируется, что приведенная i-я цена остается меньше, чем приведенная цена $\overline{P}_{ji}$. Аналогично, если $\overline{P}^{-}_{ij} < \overline{P}^{+}_{ji}$, то g_{ij} уменьшают до тех пор, пока приведенные цены не сравняются, либо одно из автономных ограничений не выйдет на границу допустимых значений. Улучшенное таким образом решение рассматривают как исходное для следующего шага (2).

После завершения итераций решение, полученное для всех фрагментов сети, состоящих из двух рынков, используют как исходное для аналогичной процедуры применительно к фрагментам сети, состоящим из трех связанных локальных рынков и т.д.

Центральной частью алгоритма является нахождение первого приближения, т.е. набора объемов передаваемой энергии между региональными рынками, при котором удовлетворяются автономные ограничения на региональные генерации. Когда допустимое начальное приближение получено, алгоритм гарантирует определение глобального минимума благодаря монотонной зависимости цен от объемов поставки в ценовых заявках.

Отметим, что не обязательно увеличивать/уменьшать g_{ij} монотонно на стадии (2) алгоритма. Поскольку фактически это дискретный поиск минимума монотонной функции (на каждой ценовой ступени минимум может быть лишь в трех точках – на концах и в любой промежуточной), экстремум может быть найден с помощью любого метода одномерного поиска.

Исследовательский центр системного анализа ИПС РАН

A. A. Akhremenkov, V. A. Kazakov, A. M. Tsirlin. *Optimization algorithm for energy markets as macrosystems* // Proceedings of Program Systems institute scientific conference “Program systems: Theory and applications”. — Pereslavl-Zalesskij, v. **1**, 2009. — p. 85–103. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. Considered optimization problem of trade on interconnected energy markets taking into account market's peculiarity. Obtained problem's optimal conditions and solving algorithm. Ability to construct system with automatic optimization is shown.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 517.957

В. А. Юмагужин, В. Н. Юмагужина

Скалярные дифференциальные инварианты уравнений $y'' = a^3(x,y)y'^3 + a^2(x,y)y'^2 + a^1(x,y)y' + a^0(x,y)$

Аннотация. Статья посвящена дифференциальным инвариантам обыкновенных дифференциальных уравнений вида

$$y'' = a^3(x,y)y'^3 + a^2(x,y)y'^2 + a^1(x,y)y' + a^0(x,y)$$

относительно точечных преобразований. Исследуется действие псевдогруппы всех таких преобразований на естественном расслоении этих уравнений. На этом пути строятся тензорные и скалярные дифференциальные инварианты рассматриваемых уравнений. Получена полная система образующих и дифференциальных соотношений между ними для алгебры всех скалярных дифференциальных инвариантов широкого класса рассматриваемых уравнений, в частности, для уравнения общего положения.

1. Введение

Эта статья посвящена дифференциальным инвариантам обыкновенных дифференциальных уравнений вида

$$y'' = a^3(x,y)y'^3 + a^2(x,y)y'^2 + a^1(x,y)y' + a^0(x,y)\,. \tag{1.1}$$

Существуют различные подходы к построению дифференциальных инвариантов этих уравнений, см., например, R. Liouville [1], S. Lie [2, 3], A. Tresse [4,5], E. Cartan [6], G. Thomsen [7], and R. B. Gardner [8].

В работах [9,10] мы представили подход к этой задаче, отличный от известных ранее. Этот подход является результатом нашей попытки перенести конструкцию структурной функции G-структуры, см. [11, 12], в расслоения джетов сечений естественного расслоения соответствующих геометрических объектов. Другой общий подход к конструированию дифференциальных инвариантов имеется в работах В. Лычагина и Б. Кругликова [13], [14].

Мы рассматриваем каждое уравнение $\mathcal{E}$ вида (1.1) как геометрическую структуру. С этой целью уравнение $\mathcal{E}$ отождествляется с сечением

$$S_{\mathcal{E}} : (x,y) \mapsto (\,x,\,y,\,a^0(x,y),\,a^1(x,y),\,a^2(x,y),\,a^3(x,y)\,)$$

тривиального расслоения

$$\pi : \mathbb{R}^2 \times \mathbb{R}^4 \longrightarrow \mathbb{R}^2,$$

где $\mathbb{R}^n$ – n-мерное арифметическое пространство. Это отождествление является биекцией множества всех рассматриваемых уравнений на множество всех сечений расслоения π. Хорошо известно, см. [15], что всякое точечное преобразование переменных x и y преобразует каждое уравнение (1.1) в уравнение того же вида [1]. Отсюда, в силу нашего отождествления, всякое точечное преобразование порождает преобразование сечений расслоения π. Иными словами, всякое точечное преобразование f базы расслоения π естественным образом поднимается до диффеоморфизма $f^{(0)}$ тотального пространства расслоения π. Таким образом, расслоение π уравнений (1.1) является естественным, и уравнение (1.1), рассматриваемое как сечение π, является геометрической структурой, см. [16]. Диффеоморфизм $f^{(0)}$ естественно поднимается до диффеоморфизма $f^{(k)}$ расслоения $J^k\pi$ k-джетов сечений расслоения π, $k = 0, 1, ..., \infty$. Т.е. точечные преобразования естественным образом действуют на всех расслоениях $J^k\pi$. В результате эти расслоения разбиваются на орбиты:

(1) Расслоения $J^0\pi$ и $J^1\pi$ являются орбитами.
(2) $J^2\pi$ является объединением двух орбит, одна из которых является орбитой общего положения, вторая – вырожденная орбита.
(3) $J^3\pi$ является объединением четырех орбит, одна из которых – орбита общего положения, остальные – вырожденные орбиты.
(4) $J^k\pi$, $k \geq 4$, является объединением непрерывного семейства вырожденных орбит.

Мы доказываем, что всякий k-джет $\theta_k \in J^k\pi$, $k \geq 1$, естественным образом порождает некоторый геометрический объект ω_{θ_k} на касательном пространстве к базе расслоения π в точке $p = \pi_k(\theta_k)$. В результате, мы получаем поле этих объектов на $J^k\pi$

$$\theta_k \longmapsto \omega_{\theta_k}.$$

[1] Обыкновенные уравнения 2-го порядка допускают контактные преобразования. Известно, см. [17], что любые два таких уравнения контактно эквивалентны.

Эти поля являются инвариантами действия псевдогруппы всех точечных преобразований базы на $J^k\pi$. Их ограничения на сечение расслоения $J^k\pi$, порожденное сечением $S_\mathcal{E}$, являются дифференциальными инвариантами порядка k уравнения $\mathcal{E}$.

В этой статье мы строим тензорные дифференциальные инварианты, различающие орбиты в расслоениях $J^2\pi$ и $J^3\pi$. Это описание орбит с помощью тензорных инвариантов является новым результатом, хотя большинство из полученных инвариантов, по существу, известны специалистам. Основным новым результатом нашей статьи является описание алгебры A всех скалярных дифференциальных инвариантов, определенных в прообразе $(\pi_{\infty,3})^{-1}(\mathrm{Orb}_0^3)$, где Orb_0^3 – орбита общего положения в $J^3\pi$, а отображение $\pi_{\infty,3} : J^\infty\pi \to J^3\pi$ – естественная проекция, сопоставляющая ∞-джету его 3-джет. Мы находим полный набор образующих алгебры A и полный набор дифференциальных соотношений между этими образующими.

В этой статье все вычисления выполнены с помощью REDUCE - программы [18].

Все многообразия и отображения в статье предполагаются гладкими. Через $j^k_p f$, $k = 0, 1, 2, \ldots$, обозначается k-джет отображения f в точке p, через $\mathbb{R}$ обозначается поле действительных чисел и через $\mathbb{R}^n$ — n-мерное арифметическое пространство. Предполагается суммирование по повторяющимся индексам во всех формулах.

2. Расслоение уравнений

Пусть

$$\pi : E = \mathbb{R}^2 \times \mathbb{R}^4 \longrightarrow \mathbb{R}^2$$

— тривиальное расслоение. Через x^1, x^2 обозначим стандартные координаты на базе этого расслоения, а через u^1, u^2, u^3, u^4 — обозначим стандартные координаты на его слое. Пусть $\mathcal{E}$ — произвольное уравнение (1.1). Мы отождествляем его с сечением $S_\mathcal{E}$ расслоения π, определенным формулой

$$S_\mathcal{E}(p) = \big(p,\, a^0(p),\, a^1(p),\, a^2(p),\, a^3(p)\,\big),$$

где $p = (x^1, x^2)$. Ясно, что это отождествление является биекцией между множеством всех уравнений (1.1) и множеством всех сечений расслоения π. Напомним, что *точечное преобразование* пространства $\mathbb{R}^2$ есть диффеоморфизм f открытой области из $\mathbb{R}^2$ в $\mathbb{R}^2$

$$f(x, y) = (\tilde{x}, \tilde{y}). \tag{2.1}$$

Оно преобразует уравнение $\mathcal{E}$ вида (1.1) в уравнение $\tilde{\mathcal{E}}$ того же вида, см. [15]. Коэффициенты $\tilde{a}$, $\tilde{b}$, $\tilde{c}$, $\tilde{d}$ уравнения $\tilde{\mathcal{E}}$ выражаются через коэффициенты уравнения $\mathcal{E}$ и 2–джет обратного преобразования f^{-1}:

$$\tilde{a}^i(\tilde{p}) = \Phi^i\big(a^0(f^{-1}(\tilde{p})), \dots, a^3(f^{-1}(\tilde{p})), j^2_{\tilde{p}} f^{-1}\big), \quad i = 0, 1, 2, 3. \tag{2.2}$$

Следовательно уравнения

$$\tilde{p} = f(p), \quad \tilde{u}^i = \Phi^i\big(u^1, \dots, u^4, j^2_{f(p)} f^{-1}\big), \quad i = 1, 2, 3, 4,$$

определяют диффеоморфизм $f^{(0)}$ расслоения π. Он называется *поднятием преобразования f в расслоение π*. Т.о. ясно, что расслоение π является естественным.

Закон преобразования уравнений (1.1) в терминах соответствующих сечений имеет следующий вид

$$S_{\tilde{\mathcal{E}}} = f^{(0)} \circ S_{\mathcal{E}} \circ f^{-1}.$$

Через $j^k_p S$ обозначим k–джет сечения S расслоения π в точке p, $k = 0, 1, 2, \dots, \infty$, через

$$\pi_k : J^k\pi \longrightarrow \mathbb{R}^2, \quad \pi_k : j^k_p S \mapsto p$$

обозначим расслоение всех таких k–джетов. Всякое точечное преобразование f формулой

$$f^{(k)}(j^k_p S) = j^k_{f(p)}\big(f^{(0)} \circ S \circ f^{-1}\big) \tag{2.3}$$

определяет диффеоморфизм $f^{(k)}$ расслоения $J^k\pi$. Он называется *поднятием преобразования f в расслоение джетов $J^k\pi$*.

Через Γ обозначим псевдогруппу всех точечных преобразований базы расслоения π. Эта псевдогруппа действует на каждом расслоении $J^k\pi$ посредством поднятых преобразований.

Стандартные координаты в $J^k\pi$ обозначим через x^1, x^2, u^i_σ, где $i = 1, \dots, 4$, σ — мультииндекс $\{j_1 \dots j_r\}$, $j_1, \dots, j_r = 1, 2$, $|\sigma| = r$, $0 \le |\sigma| \le k$. Через σj обозначим мультииндекс $\{j_1 \dots j_r j\}$. Естественная проекция

$$\pi_{k,\,r} : J^k\pi \longrightarrow J^r\pi, \quad \infty \ge k > r,$$

определяется формулой $\pi_{k,\,r}(j^k_p S) = j^r_p S$. Через $J^k_p\pi$ обозначим слой расслоения π_k над точкой p.

3. Алгебры изотропии и орбиты

Пусть X — векторное поле на базе расслоения π и f_t — поток этого поля. Тогда поток $f_t^{(k)}$ на расслоении $J^k\pi$ определяет векторное поле $X^{(k)}$ in $J^k\pi$, которое нзывается *подъемом поля X в расслоение $J^k\pi$*. Очевидно

$$(\pi_{l,\,m})_*\big(\,X^{(l)}\,\big) = X^{(m)}\,,\ \ \infty \geq l > m \geq -1\,.$$

Через $X^{(k)}_{\theta_k}$ обозначим значение поля $X^{(k)}$ в точке θ_k. Ясно, что вектор $X^{(k)}_{\theta_k}$ определяется $k+2$-джетом $j^{k+2}_p X$, где $p = \pi_k(\theta_k)$.

Алгебра изотропии $\mathfrak{g}_{\theta_k}$ точки θ_k определяется формулой

$$\mathfrak{g}_{\theta_k} = \big\{\, j^{2+k}_p X \,\big|\, X^{(k)}_{\theta_k} = 0 \,\big\}.$$

Структура алгебры Ли на $\mathfrak{g}_{\theta_k}$ порождается операцией коммутирования векторных полей, обращающихся в ноль в точке p.

$$[\, j^{2+k}_p X,\, j^{2+k}_p Y\,] = j^{2+k}_p [\,X,\,Y\,]\,.$$

Имеет место следующее утверждение:

Теорема 3.1.

(1) *При $k = 0, 1$ $\dim \mathfrak{g}_{\theta_k} = 6$ для всех $\theta_k \in J^k\pi$.*
(2) *Размерность алгебры изотропии точки $\theta_2 \in J^2\pi$ равна либо 4, либо 6.*
(3) *Размерность алгебры изотропии точки $\theta_3 \in J^3\pi$ равна: либо 0, либо 1, либо 2, либо 6.*

Из этой теоремы вытекает

Теорема 3.2.

(1) *$J^k\pi$, $k = 0, 1$, является орбитой действия Γ,*
(2) *$J^2\pi$ – объединение двух орбит: орбиты общего положения Orb^2_0 и вырожденной орбиты Orb^2_2.*

$$\mathrm{Orb}^2_0 = \{\theta_2 \in J^2\pi |\, \dim \mathfrak{g}_{\theta_2} = 4\}, \quad \mathrm{Orb}^2_2 = \{\theta_2 \in J^2\pi |\, \dim \mathfrak{g}_{\theta_2} = 6\}.$$

(3) *$J^3\pi$ – объединение 4-х орбит: орбиты общего положения Orb^3_0 и вырожденных орбит Orb^3_1, Orb^3_2 и Orb^3_6.*

$$\mathrm{Orb}^3_r = \{\, \theta_3 \in J^2\pi \,|\, \dim \mathfrak{g}_{\theta_3} = r \,\}, \quad r = 0, 1, 2, 6.$$

Ниже нам понадобятся следующие идеалы алгебр $\mathfrak{g}_{\theta_2}$ и $\mathfrak{g}_{\theta_3}$:

$$g_{\theta_1} = \left\{ j_p^3 X \in \mathfrak{g}_{\theta_1} \,\middle|\, j_p^1 X = 0 \right\}, \quad g_{\theta_2} = \left\{ j_p^4 X \in \mathfrak{g}_{\theta_2} \,\middle|\, j_p^1 X = 0 \right\}.$$

Прямым вычислением можно показать, что каждый из этих идеалов естественным образом отождествляется с подпространством

$$g = \langle e_1, e_2 \rangle \subset T_p \otimes (T_p^* \odot T_p^*)\,,$$

где

$$\begin{aligned} e_1 &= 2\frac{\partial}{\partial x^1} \otimes (dx^1 \odot dx^1) + \frac{\partial}{\partial x^2} \otimes (dx^1 \odot dx^2)\,, \\ e_2 &= 2\frac{\partial}{\partial x^2} \otimes (dx^2 \odot dx^2) + \frac{\partial}{\partial x^1} \otimes (dx^1 \odot dx^2)\,, \end{aligned} \tag{3.1}$$

T_p и T_p^* – касательное и кокасательное пространство к базе расслоения π в точке p.

4. Пространства формальных симметрий

Каждое сечение S расслоения π порождает сечение $j_k S$ расслоения π_k

$$j_k S : p \;\mapsto\; j_p^k S.$$

Пусть $\theta_{k+1} \in J^{k+1}\pi$ и $j_p^{k+1} S = \theta_{k+1}$. Тогда θ_{k+1} отождествляется с касательным пространством к образу сечения $j_k S$ в точке $\theta_k = j_p^k S$. Мы обозначим это пространство через $\mathcal{K}_{\theta_{k+1}}$. Рассмотрим векторные поля X на базе расслоения π, проходящие через точку p. Следующее векторное пространство $k+2$-джетов этих полей в точке p играет важную роль в нашем подходе к построению дифференциальных инвариантов.

$$\mathcal{A}_{\theta_{k+1}} = \{\, j_p^{2+k} X \mid X_{\theta_k}^{(k)} \in \mathcal{K}_{\theta_{k+1}} \,\}\,. \tag{4.1}$$

Несложно доказать, что скобка векторных полей порождает билинейное кососимметричное отображение

$$[\ ,\ ] : \mathcal{A}_{\theta_{k+1}} \times \mathcal{A}_{\theta_{k+1}} \to \mathcal{A}_{\theta_k}, \quad [\, j_p^{k+2}\xi_1,\, j_p^{k+2}\xi_2 \,] \mapsto j_p^{k+1}[\xi_1,\, \xi_2].$$

Рассмотрим алгебру изотропии $\mathfrak{g}_{\theta_k}$ точки $\theta_k = \pi_{k+1,k}(\theta_{k+1})$. Ясно, что

$$\mathfrak{g}_{\theta_k} \subset \mathcal{A}_{\theta_{k+1}} \;\; \forall\, \theta_{k+1} \in \pi_{k+1,k}^{-1}(\theta_k) \;\; \forall\, \theta_k \in J^k\pi\,. \tag{4.2}$$

Обозначим через $\rho_{m,n}$, $m > n \geq 0$, естественную проекцию, ставящую в соответствие m-джету веторного поля X его n-джет

$$\rho_{m,n} : j_p^m X \mapsto j_p^n X, \quad m > n \geq 0\,.$$

Легко видеть, что проекция

$$\rho_{k+2,0}\big|_{\mathcal{A}_{\theta_{k+1}}} : \mathcal{A}_{\theta_{k+1}} \to T_p, \quad p = \pi_{k+1}(\theta_{k+1})\,,$$

является сюръекцией.

Подпространство $H \subset \mathcal{A}_{\theta_{k+1}}$ называется *горизонтальным* если $\rho_{k+2,0}$ проектирует H на T_p изоморфно. Легко видеть, что если H – горизонтальное подпространство в $\mathcal{A}_{\theta_{k+1}}$, то

$$\mathcal{A}_{\theta_{k+1}} = H \oplus \mathfrak{g}_{\theta_k}\ .$$

Любые два горизонтальных подпространства $H, \tilde{H} \subset \mathcal{A}_{\theta_{k+1}}$ определяют линейное отображение

$$f_{H,\tilde{H}} : T_p \to \mathfrak{g}_{\theta_k}\,, \quad f_{H,\tilde{H}} : X \mapsto (\rho_{k+2}|_H)^{-1}(X) - (\rho_{k+2}|_{\tilde{H}})^{-1}(X)\,.$$

Пусть $H \subset \mathcal{A}_{\theta_{k+1}}$ – горизонтальное подпространство и пусть $f : T_p \to \mathfrak{g}_{\theta_k}$ – линейное отображение. Тогда существует единственное горизонтальное подпространство $\tilde{H} \subset \mathcal{A}_{\theta_{k+1}}$ такое, что $f = f_{H,\tilde{H}}$. Это подпространство натянуто на джеты $(\rho_{k+2}|_H)^{-1}(X) - f(X)\,,\ X \in T_p$.

5. Тензорные дифференциальные инварианты

5.1. Инварианты на $J^2\pi$

5.1.1. *Препятствие к линеаризуемости*

Пусть θ_2 – произвольная точка в $J^2\pi$, $\theta_0 = \pi_{2,0}(\theta_2)$ и $p = \pi_2(\theta_2)$.

Предложение 5.1. *Существуют горизонтальные подпространства $H \subset \mathcal{A}_{\theta_2}$ удовлетворяющие условию*

$$\rho_{2,1}(\,[\,j^3_pX\,,\,j^3_pY\,]\,) = 0 \quad \forall\, j^3_pX, j^3_pY \in H\,. \tag{5.1}$$

Пусть $H \subset \mathcal{A}_{\theta_2}$ – горизонтальное подпространство, удовлетворяющее (5.1). Тогда, принимая во внимание изоморфизм $\rho_{3,0} : H \to T_p$, $\rho_{3,0} : j^3_pX \mapsto X_p$, мы можем определить 2–форму на T_p

$$\omega_H(X_p, Y_p) = [\,j^3_pX,\, j^3_pY\,] \quad \forall\, X_p, Y_p \in T_p\,.$$

Теорема 5.2. *Форма ω_H не зависит от выбора горизонтального подпространства H, удовлетворяющего* (5.1).

Таким образом, определен тензорный дифференциальный инвариант на $J^2\pi$

$$\omega^2 : \theta_2 \mapsto \omega_H.$$

В стандартных координатах он выглядит следующим образом

$$\omega^2 = (\ L_1 \cdot e_1 + L_2 \cdot e_2\ ) \otimes (dx^1 \wedge dx^2)\,, \tag{5.2}$$

где

$$\begin{aligned} L_1 &= 3u^1_{22} - 2u^2_{12} + u^3_{11} \\ &\quad + 3u^4u^1_1 - 3u^3u^1_2 + 2u^2u^2_2 - u^2u^3_1 - 3u^1u^3_2 + 6u^1u^4_1\,, \\ L_2 &= u^2_{22} - 2u^3_{12} + 3u^4_{11} \\ &\quad - 3u^1u^4_2 + 3u^2u^4_1 - 2u^3u^3_1 + u^3u^2_2 + 3u^4u^2_1 - 6u^4u^1_2\,, \end{aligned}$$

и e_1, e_2 определяются формулами (3.1).

Обозначим через $\omega^2_{\mathcal{E}}$ ограничение ω^2 на образ сечения $j_2S_{\mathcal{E}} : p \mapsto j^2_pS_{\mathcal{E}}$. Т.о. $\omega^2_{\mathcal{E}}$ – дифференциальный инвариант уравнения $\mathcal{E}$.

Предложение 5.3. *Уравнение $\mathcal{E}$ вида (1.1) приводится к линейному виду точечным преобразованием тогда и только тогда, когда $\omega^2_{\mathcal{E}} = 0$.*

Инвариант ω^2 различает орбиты расслоения $J^2\pi$.

Теорема 5.4.

(1) $\theta_2 \in \mathrm{Orb}^2_0$ *тогда и только тогда, когда* $\omega^2(\theta_2) \neq 0$.
(2) $\theta_2 \in \mathrm{Orb}^2_2$ *тогда и только тогда, когда* $\omega^2(\theta_2) = 0$.

5.1.2. *Следующие инварианты*

Применяя к ω^2 операцию свертки тензора по верхнему и нижнему индексу, получим тензорный дифференциальный инвариант

$$\alpha^2 = (L_1dx^1 + L_2dx^2) \otimes (dx^1 \wedge dx^2). \tag{5.3}$$

Т.к. $\dim T_p = 2$, то свертка

$$T_p \otimes (\wedge^2 T^*_p) \otimes (\wedge^2 T^*_p) \longrightarrow T^*_p \otimes (\wedge^2 T^*_p)\,, \quad (t^i_{r_1s_1,r_2s_2}) \mapsto (t^m_{ms_1,r_2s_2})$$

является изоморфизмом. Следовательно прообраз тензора $(1/2)\alpha^2$ при этом изоморфизме приводит к тензорному дифференциальному инварианту

$$\beta^2 : \theta_2 \longmapsto \beta^2_{\theta_2} = (L_2(\theta_2)\frac{\partial}{\partial x^1} - L_1(\theta_2)\frac{\partial}{\partial x^2}) \otimes (dx^1 \wedge dx^2)^2\,. \tag{5.4}$$

5.2. Инварианты на $J^3\pi$

5.2.1. *Первый нетривиальный инвариант*

В этом параграфе мы построим дифференциальные инварианты на $(\pi_{3,2})^{-1}(\mathrm{Orb}_0^2) \subset J^3\pi$.

Пусть $\theta_3 \in (\pi_{3,2})^{-1}(\mathrm{Orb}_0^2) \subset J^3\pi$ и let $p = \pi_3(\theta_3)$.

Предложение 5.5. *Существуют такие горизонтальные подпространства $H \subset \mathcal{A}_{\theta_3}$, что*

$$\rho_{3,1}\big([\,j_p^4X,\, j_p^4Y\,]\big) = 0 \quad \forall\, j_p^4X, j_p^4Y \in H. \tag{5.5}$$

Пусть H – горизонтальное подпространство в $\mathcal{A}_{\theta_3}$, удовлетворяющее (5.5). Рассмотрим произвольные векторы j_p^4X, j_p^4Y, j_p^4U и j_p^4Z в H. Тогда $\Big[j_p^2U, \big[j_p^3Z, [j_p^4X, j_p^4Y]\big]\Big] = j_p^1V$. Существует такой единственный вектор $j_p^4W \in H$, что $W_p = V_p$. Тогда $j_p^1V - j_p^1W \in T_p \otimes T_p^*$. Можно доказать, что формула

$$t_H(X_p, Y_p, Z_p, U_p) = j_p^1V - j_p^1W \quad \forall\, U_p, Z_p, X_p, Y_p \in T_p$$

определяет тензор t_H, принадлежащий пространству $T_p \otimes (T_p^* \odot T_p^* \odot T_p^*) \otimes (T_p^* \wedge T_p^*)$. Существует естественная проекция

$$\mu : T_p \otimes (T_p^* \odot T_p^*) \to g \subset T_p \otimes (T_p^* \odot T_p^*)\,, \quad (X^i_{jk}) \mapsto \big(\frac{1}{3}(\delta^i_j\, X^r_{kr} + \delta^i_k\, X^r_{jr})\big)\,,$$

где δ^i_j – символ Кронекера. Эта проекция порождает естественную проекцию

$$\tilde\mu : T_p \otimes (T_p^* \odot T_p^*) \otimes T_p^* \otimes (T_p^* \wedge T_p^*) \to g \otimes T_p^* \otimes (T_p^* \wedge T_p^*).$$

Принимая во внимание, что

$$T_p \otimes (T_p^* \odot T_p^* \odot T_p^*) \otimes (T_p^* \wedge T_p^*) \subset T_p \otimes (T_p^* \odot T_p^*) \otimes T_p^* \otimes (T_p^* \wedge T_p^*),$$

мы можем рассмотреть тензор $\tilde\mu(t_H) \in g \otimes T_p^* \otimes (T_p^* \wedge T_p^*)$ как 1-форму со значениями в $g \otimes (T_p^* \wedge T_p^*)$. Тогда свертка

$$\big(T_p \otimes (T_p^* \wedge T_p^*)^2\big) \lrcorner \Big(\big(g \otimes (T_p^* \wedge T_p^*)\big) \otimes T_p^*\Big) \subset g \otimes (T_p^* \wedge T_p^*)^3\,,$$

$$(t^i_{r_1s_1r_2s_2}) \lrcorner (p^i_{jk,r_3s_3,l}) = (t^m_{r_1s_1r_2s_2} p^i_{jk,r_3s_3,m})\,,$$

определяет новый тензор

$$\omega_H^3 = \beta^2_{\theta_2} \lrcorner \tilde\mu(t_H) \in g \otimes (T_p^* \wedge T_p^*)^3.$$

Теорема 5.6. *Тензор ω_H^3 не зависит от выбора горизонтального подпространства $H \subset \mathcal{A}_{\theta_3}$, удовлетворяющего (5.5).*

Положим

$$\omega^3_{\theta_3} = \omega^3_H.$$

Следовательно θ_3 порождает естественным образом тензор $\omega^3_{\theta_3} \in g \otimes (\wedge^2 T^*_p)^3$

$$\omega^3_{\theta_3} = \left(\Psi^1(\theta_3)e_1 + \Psi^2(\theta_3)e_2\right) \otimes (dx^1 \wedge dx^2)^3,$$

где

$$\begin{aligned}\Psi^1(\theta_3) = &-(L_1)^2u^2 + 2L_1L_2u^1 - 3(L_2)^2u^0\\ &- L_1D_2(L_1) + 4L_1D_1(L_2) - 3L_2D_1(L_1),\\ \Psi^2(\theta_3) = &-3(L_1)^2u^3 + 2L_1L_2u^2 - (L_2)^2u^1\\ &+ 3L_1D_2(L_2) - 4L_2D_2(L_1) + L_2D_1(L_2).\end{aligned}$$

Т.о.

$$\omega^3 : \theta_3 \mapsto \omega^3_{\theta_3}$$

является дифференциальным инвариантом на $(\pi_{3,2})^{-1}(\mathrm{Orb}^2_0) \subset J^3\pi$.

5.2.2. *Следующие инварианты*

Применяя к ω^3 операцию свертки тензора по верхнему и нижнему индексу, получим тензорный дифференциальный инвариант

$$\alpha^3 : \theta_3 \longmapsto \alpha^3_{\theta_3} = (\Psi^1(\theta_3)dx^1 + \Psi^2(\theta_3)dx^2) \otimes (dx^1 \wedge dx^2)^3. \tag{5.6}$$

Свертка тензоров β^2 и α^3 приводит к тензорному дифференциальному инварианту

$$\nu : \theta_3 \longmapsto \frac{1}{3}(\beta^2_{\theta_2} \lrcorner \alpha^3_{\theta_3}) = L_3(\theta_3)(dx^1 \wedge dx^2)^5, \tag{5.7}$$

где

$$\begin{aligned}L_3(\theta_3) = &L_2(L_1D_1(L_2) - L_2D_1(L_1)) - L_1(L_1D_2(L_2) - L_2D_2(L_1))\\ &+ (L_1)^3u^3 - (L_1)^2L_2u^2 + L_1(L_2)^2u^1 - (L_2)^3u^0.\end{aligned}$$

Операция свертки между β^2 и ω^3 приводит к тензорному дифференциальному инварианту

$$\gamma : \theta_3 \longmapsto \left(\gamma^i_j(\theta_3)\partial_{x^i} \otimes dx^j\right) \otimes (dx^1 \wedge dx^2)^5,$$

где коэффициенты γ^i_j, $i,j=1,2$, определены формулами

$$\begin{aligned}\gamma^1_2 &= L_2(3L_1D_2L_2 - 4L_2D_2L_1 + L_2D_1L_2\\ &\qquad - 3(L_1)^2u^4 + 2L_1L_2u^3 - (L_2)^2u^2),\\ \gamma^2_1 &= L_1(3L_2D_1L_1 - 4L_1D_1L_2 + L_1D_2L_1\\ &\qquad + (L_1)^2u^3 - 2L_1L_2u^2 + 3(L_2)^2u^1),\\ &(L_1)^2\gamma^1_2 + (L_2)^2\gamma^2_1 + 3L_1L_2L_3 = 0,\\ &L_2\gamma^1_1 + L_1\gamma^1_2 - 6L_2L_3 = 0, \quad L_2\gamma^2_2 + L_1\gamma^1_2 - 6L_2L_3 = 0.\end{aligned}$$

Инварианты ω^2, ν и γ определяют орбиты расслоения $J^3\pi$ полностью.

Теорема 5.7.

(1) $\theta_3 \in \mathrm{Orb}^0_3$ *тогда и только тогда, когда* $\omega^2(\pi_{3,2}(\theta_3)) \neq 0$ *и* $\nu(\theta_3)\neq 0$.
(2) $\theta_3 \in \mathrm{Orb}^1_3$ *тогда и только тогда, когда* $\omega^2(\pi_{3,2}(\theta_3)) \neq 0$, $\nu(\theta_3)=0$ *и* $\gamma(\theta_3)\neq 0$.
(3) $\theta_3 \in \mathrm{Orb}^2_3$ *тогда и только тогда, когда* $\omega^2(\pi_{3,2}(\theta_3)) \neq 0$, $\nu(\theta_3)=0$ *и* $\gamma(\theta_3)=0$.
(4) $\theta_3 \in \mathrm{Orb}^6_3$ *тогда и только тогда, когда* $\omega^2(\pi_{3,2}(\theta_3))=0$.

Следующие инварианты понадобятся нам при исследовании алгебры скалярных дифференциальных инвариантов.

Точно так, как был получен β^2 из α^2, мы получаем из α^3 тензорный дифференциальный инвариант

$$\beta^3 : \theta_3 \longmapsto \beta^3_{\theta_3} = (\Psi^2(\theta_3)\frac{\partial}{\partial x^1} - \Psi^1(\theta_3)\frac{\partial}{\partial x^2}) \otimes (dx^1 \wedge dx^2)^4 .$$

Для всякой точки $\theta_3 \in \mathrm{Orb}^3_0$ тензоры ν_{θ_3}, $\beta^2_{\theta_2}$ и $\beta^3_{\theta_3}$, где $\theta_2 = \pi_{3,2}(\theta_3)$, порождают естественным образом векторы $\xi_{1_{\theta_3}}$ and $\xi_{2_{\theta_3}}$:

(5.8)
$$\xi_{1_{\theta_3}} = \frac{1}{(L_3)^{2/5}}(L_2\frac{\partial}{\partial x^1} - L_1\frac{\partial}{\partial x^2}), \quad \xi_{2_{\theta_3}} = \frac{1}{(L_3)^{4/5}}(\Psi^2\frac{\partial}{\partial x^1} - \Psi^1\frac{\partial}{\partial x^2}).$$

Поля на Orb^3_0

$$\xi^3_1 : \theta_3 \longmapsto \xi_{1_{\theta_3}}, \quad \xi^3_2 : \theta_3 \longmapsto \xi_{2_{\theta_3}}, \quad \forall\, \theta_3 \in \mathrm{Orb}^3_0$$

являются дифференциальными инвариантами.

Предложение 5.8. *Вектора* $\xi_{1_{\theta_3}}, \xi_{2_{\theta_3}}$ *из* T_p, $p=\pi_3(\theta_3)$, *являются линейно-независимыми во всякой точки* $\theta_3 \in \mathrm{Orb}^3_0$.

Теперь можно определить векторные поля ξ_1 and ξ_2 на подрасслоении $(\pi_{\infty,3})^{-1}(\mathrm{Orb}_0^3)$.

$$\xi_1 = \frac{1}{(L_3)^{2/5}}(L_2D_1 - L_1D_2), \quad \xi_2 = \frac{1}{(L_3)^{4/5}}(\Psi^2D_1 - \Psi^1D_2), \tag{5.9}$$

где D_1 и D_2 – операторы полных производных по переменным x^1 и x^2 соответственно.

Ясно, что эти поля являются дифференциальными инвариантами.

6. Скалярные дифференциальные инварианты

Впервые скалярные дифференциальные инварианты появляются в расслоении $J^4\pi$.

Через A_k мы обозначим алгебру всех скалярных дифференциальных инвариантов порядка k, определенных в $(\pi_{k,3})^{-1}(\mathrm{Orb}_0^3)$, $k > 3$.

Всякая функция от инвариантов из A_k является инвариантом из A_k. Следовательно максимальный набор $\{ I^1, \ldots, I^{N_k} \}$ функционально независимых инвариантов из A_k порождает A_k, т.е. всякий инвариант $I \in A_k$ является некоторой функцией инвариантов $I^1, \ldots, I^{N_k}$. Подсчет размерностей орбит приводит к следующему утверждению

Предложение 6.1.

(1) *Алгебра A_k, $0 \leq k \leq 3$, тривиальна, т.е. она состоит только из констант.*
(2) *Алгебра A_k, $k \geq 4$, порождена $k^2 - k - 6$ функционально независимыми инвариантами. В частности, A_4 порождена 6-ю независимыми инвариантами.*

Отождествляя инварианты $I \in A_k$ и $(\pi_{k+1,k})_*(I) \in A_{k+1}$, мы получим фильтрацию

$$A_0 \subset A_1 \subset \ldots \subset A_k \subset \ldots$$

Алгебра

$$A = \bigcup_0^\infty A_k$$

называется алгеброй всех скалярных дифференциальных инвариантов в $(\pi_{\infty,3})^{-1}(\mathrm{Orb}_0^3)$.

Ясно, что если $J \in A_k$, то производная Ли $\xi_j(J)$ этого инварианта вдоль инвариантного векторного поля ξ_j, $j = 1, 2$, принадлежит A_{k+1}. Т.о. A имеет два независимых дифференцирования ξ_1 и ξ_2. По этой причине алгебра A имеет конечное множество образующих. Это значит, что всякий инвариант из A является функцией этих образующих и их производных Ли некоторых порядков вдоль векторных полей ξ_1 и ξ_2.

6.1. Образующие и соотношения

Алгебра A_4 порождена 6-ю независимыми инвариантами. Найдем их.

Рассмотрим инварианты ω^2, ω^3 and ν. Они порождают два новых тензорных инварианта

$$\tilde{\omega}^2(\theta_3) = \big(L_1(\theta_2)e_1 + L_2(\theta_2)e_2\big)/L_3^{1/5},$$
$$\tilde{\omega}^3(\theta_3) = \big((\Psi^1(\theta_3)e_1 + (\Psi^2(\theta_3)e_2\big)/L_3^{3/5}.$$

Из конструкции инвариантов ω^2 и ω^3 следует, что

$$\tilde{\omega}^2(\theta_3) \in g, \quad \text{and} \quad \tilde{\omega}^3(\theta_3) \in g.$$

Производные Ли $\xi_i(\tilde{\omega}^j)$, $i = 1, 2$, $j = 2, 3$, этих инвариантов являются новыми тензорными инвариантами и

$$\xi_i(\tilde{\omega}^j) \in g, \quad i = 1, 2, \ j = 2, 3.$$

Сравнивая $\tilde{\omega}^2(\theta_3)$ и $\tilde{\omega}^3(\theta_3)$ с ξ_1 и ξ_2, мы получаем, что $\tilde{\omega}^2(\theta_3)$ и $\tilde{\omega}^3(\theta_3)$ линейно независимы и образуют базис в g. Следовательно разложения

$$\xi_i(\tilde{\omega}^j) = J^j_{i2} \cdot \tilde{\omega}^2 + J^j_{i3} \cdot \tilde{\omega}^3$$

дают 8 скалярных дифференциальных инвариантов J^j_{ir}, $i = 1, 2$, $j, r = 2, 3$. Однако, среди этих инвариантов нет 6-и функционально независимых.

Инвариант ν порождает инвариантную форму объема

$$\tilde{\nu} = (L_3)^{1/5} dx^1 \wedge dx^2.$$

Производные Ли этого инварианта $\xi_1(\tilde{\nu})$ и $\xi_2(\tilde{\nu})$ определяют два новых скалярных инварианта J^1 and J^2:

$$\xi_1(\tilde{\nu}) = J^1 \cdot \tilde{\nu}, \quad \xi_2(\tilde{\nu}) = J^2 \cdot \tilde{\nu}.$$

Вычисления показывают, что семейство $\{J^r_{ij}, J^1, J^2\}$ не содержит 6-и функционально независимых инварианта.

Рассмотрим пространство $\mathcal{A}_{\theta_4}$. Имеем

$$\mathcal{A}_{\theta_4} = H \oplus \mathfrak{g}_{\theta_3} \ .$$

Можно доказать, что $\mathfrak{g}_{\theta_3} = \{0\}$. Т.о. $\mathcal{A}_{\theta_4}$ является горизонтальным пространством. Напомним, что естественная проекция $\rho_{5,0} : \mathcal{A}_{\theta_4} \to T_p$, $j^5_p X \mapsto X_p$ является изоморфизмом. Далее напомним, что вектора

$$\xi_{1_{\theta_3}} = (\pi_\infty)_*(\xi_1), \quad \xi_{2_{\theta_3}} = (\pi_\infty)_*(\xi_2)$$

образуют базис в T_p. Через $j^5_p X_1$ и $j^5_p X_2$ обозначим такие вектора в $\mathcal{A}_{\theta_4}$, что

$$\rho_{5,0}(j^5_p X_1) = \xi_{1_{\theta_3}}, \quad \rho_{5,0}(j^5_p X_2) = \xi_{2_{\theta_3}}.$$

Тогда

$$[j^5_p X_1,\, j^5_p X_2] \in \mathcal{A}_{\theta_3}.$$

Через $j^4_p Y$ обозначим такой вектор горизонтального подпространства $\rho_{5,4}(H)$ в $\mathcal{A}_{\theta_3}$, что

$$\rho_{4,0}([j^5_p X_1,\, j^5_p X_2]) = \rho_{4,0}(j^4_p Y).$$

Тогда

$$\Delta(\theta_4) = \rho_{4,1}(j^4_p Y - [j^5_p X_1,\, j^5_p X_2]) \in g^1_{\theta_2} \subset T_p \otimes T^*_p.$$

Базис $\{\xi_{1_{\theta_3}}, \xi_{2_{\theta_3}}\}$ в T_p порождает базис $\{\, e^1_1,\, e^1_2,\, e^2_1,\, e^2_2 \,\}$ в $T_p \otimes T^*_p$. Следовательно разложение

$$\Delta(\theta_4) = J^3(\theta_4) \cdot e^1_1 + J^4(\theta_4) \cdot e^1_2 + J^5(\theta_4) \cdot e^2_1 + J^6(\theta_4) \cdot e^2_2$$

порождает 4 новых скалярных инварианта J^3, J^4, J^5, J^6.

Теорема 6.2. *Инварианты*

$$J^2_{22},\ J^3_{22},\ J^1,\ J^2,\ J^3,\ J^6$$

образуют максимальную систему функционально независимых инвариантов алгебры A_4.

Ясно, что всякий скалярный инвариант $I \in A_k$ однозначно определяется своим ограничением на всякий слой $(\pi_{k,3})^{-1}(\theta_3)$, где $\theta_3 \in \mathrm{Orb}^3_0$. Выберем точку $\theta_3 \in J^3\pi$ так, что ее координаты $u^1_{y^2}$, $u^2_{y^3}$ являются ненулевыми, а все остальные координаты равны нулю. Тогда $L_1(\theta_3) = 3u^1_{22}$ and $L_3(\theta_3) = -9u^2_{222}(u^1_{22})^2$. Следовательно $\theta_3 \in \mathrm{Orb}^3_0$. Через F^4 обозначим слой $(\pi_{k,3})^{-1}(\theta_3)$.

Выражения ограничений инвариантов J^2_{22}, J^3_{22}, J^1, J^2, J^3, J^6 на слой F^4 весьма громоздки. С целью их упрощения, преобразуем

эти инварианты. В результате получим следующие 6 функционально независимых инвариантов алгебры A_4:

$$
\begin{aligned}
I^1 &= (-4J^2_{22} - 15J^3_{22}J^2 + 30J^1J^2 - 30J^3)/6,\\
I^2 &= (-11J^3_{22} + 4J^1)/18,\\
I^3 &= (5J^2)/3,\\
I^4 &= (20J^2_{22}J^2 - 15(J^3_{22})^2 - 25J^3_{22}J^1 - 10(J^1)^2 - 10J^6)/6,\\
I^5 &= (-5J^2_{22} - 20J^3_{22}J^2 + 40J^1J^2 - 40J^3)/6,\\
I^6 &= (-7J^3_{22} + 3J^1)/9.
\end{aligned}
$$

Их ограничения на слой F^4 максимально просты:

$$
\begin{aligned}
I^1|_{F^4} &= (u^4_{1111} - u^2_{1122} + 2u^1_{1222})/(u^1_{22}L_3^{1/5}),\\
I^2|_{F^4} &= (u^4_{1112} - u^2_{1222} + 2u^1_{2222})u^1_{22}/L_3^{4/5},\\
I^3|_{F^4} &= (3u^4_{1122} - 2u^3_{1222} + u^2_{2222})u^1_{22}/(u^2_{222}L_3^{2/5}),\\
I^4|_{F^4} &= (u^3_{1111} - 2u^2_{1112} + 3u^1_{1122})u^2_{222}/(u^1_{22}L_3^{3/5}),\\
I^5|_{F^4} &= (u^3_{1112} - 2u^2_{1122} + 3u^1_{1222})/(u^1_{22}L_3^{1/5}),\\
I^6|_{F^4} &= (u^3_{1122} - 2u^2_{1222} + 3u^1_{2222})u^1_{22}/L_3^{4/5}.
\end{aligned}
$$

Пусть $\theta_\infty \in (\pi_{\infty,3})^{-1}(\theta_3)$. Тогда

$$
\xi_1\big|_{\theta_\infty} = -3\big(u^1_{22}/L_3^{2/5}\big)D_2, \quad \xi_2\big|_{\theta_\infty} = 9\big(u^2_{222}u^1_{22}/L_3^{4/5}\big)D_1.
$$

Теперь из предложения (6.1) вытекают следующие утверждения.

Теорема 6.3. *Семейство инвариантов*

$$
I^1,\ I^2,\ I^3,\ I^4,\ I^5,\ I^6
$$

является системой образующих алгебры A.

Теорема 6.4.

(1) *Образующие* $I^1, I^2, \ldots, I^6$ *алгебры* A *удовлетворяют 4-м дифференциальным соотношениям:*

$$5\xi_1(I^1) - 15\xi_2(I^2) + 57I^1I^3 + 1836(I^2)^2 - 2484I^2I^6 - 39I^3I^5 + 810(I^6)^2 + 20I^4 = 0,$$

$$135\xi_1(I^2) + 15\xi_2(I^3) - 90\xi_1(I^6) - 297I^2I^3 + 243I^3I^6 + 75I^1 - 50I^5 = 0,$$

$$15\xi_1(I^4) + 5\xi_2(I^5) + 225I^1I^6 - 2286I^2I^5 - 6I^3I^4 + 1539I^5I^6 - 75 = 0,$$

$$5\xi_1(I^5) - 15\xi_2(I^6) + 60I^1I^3 + 1620(I^2)^2 - 2079I^2I^6 - 40I^3I^5 + 621(I^6)^2 + 25I^4 = 0.$$

(2) *Все другие дифференциальные соотношения между образующими* $I^1, I^2, \ldots, I^6$ *алгебры* A *являются следствиями этих 4-х соотношений.*

Список литературы

[1] Liouville R. *Sur les invariantes de certaines equationes differentielles.* // Jour. de l'Ecole Politechnique, Leipzig. — **59**, 1889, c. 7–88. ↑1

[2] Lie S. Vorlesungen über continuierliche gruppen. — Leipzig: Teubner, 1893. ↑1

[3] Lie S. Theorie der transformationsgruppen. — Leipzig: Vol. III, Teubner, 1930. ↑1

[4] Tresse A. *Sur les invariants differentiels des groupes continus de transformations* // Acta Math. — **18**, 1894, c. 1-88. ↑1

[5] Tresse A. Détermination des Invariants ponctuels de l'Equation differentielle ordinaire du second ordre: $y'' = \omega(x, y, y')$. — Leipzig: Bei S. Hirzel, 1896. — 88 c. ↑1

[6] Cartan E. *Sur les varietes a connexion projective.* // Bull. Soc. Math. France. — **52**, 1924, c. 205-241. ↑1

[7] Thomsen G. *Uber die topologischen Invarianten der Differentialgleichung* $y'' = f(x,y)y'^3 + g(x,y)y'^2 + h(x,y)y' + k(x,y)$. // Abh. Math. Sem. Hamburg. Univ. — **7**, 1930, c. 301-328. ↑1

[8] Gardner R. B. *Differential geometric methods interacting control theory.* // in R.W. Brockett et al., Eds., *Differential geometry control theory*, 1983, c. 117-180. ↑1

[9] Yumaguzhin V. A. *On the obstruction to linearizability of 2-order ordinary differential equations* // Acta Applicandae Mathematicae. — **83**, № 1-2, 2004, c. 133-148, arXiv:0804.0306. ↑1

[10] Yumaguzhin V. A. *On the obstruction to integrability of almost-complex structures* // arXiv:0804.0690v1, 2008. ↑1

[11] Singer I. M., Sternberg S. *On the infinite groups of Lie and Cartan,I.* // J. Analyse Math. — **15**, 1965, c. 1-114. ↑1

[12] Sternberg S. Lectures on Differential Geometry. — New Jersy: Prentice-Hall, Inc., 1964. — 1-114 c. ↑1

[13] Lychagin V. *Homogeneous geometric structures and homogeneous differential equations.* // The Interplay between differential Geometry and Differential Equations, AMS Translations, Advances in Math. Sciences,. — **2**, № 167, 1995, c. 143–164. ↑1

[14] Kruglikov B., Lychagin V. V. *On equivalence of differential equations.* // Acta et Comment. Univ. Tartuensis Math. — **3**, 1999, c. 7-29. ↑1

[15] Arnold V. I. Advanced chapters of the theory of ordinary differential equations. — Moscow: Nauka, 1978. — 400 c., (in Russian). ↑1, 2

[16] Alekseevskiy D. V., Lychagin V. V., Vinogradov A. M. Fundamental ideas and conceptions of differential geometry. Sovremennye problemy matematiki. Fundamental'nye napravleniy. Itogi nauki i techniki, Vol. 28. — Moscow: VINITI, AN SSSR, 1988. — 298 c., (in Russian) [English transl.: Encyclopedia of Math. Sciences, Vol.28, pp. 298, Springer, Berlin, (1991)]. ↑1

[17] Chern S. *Projective geometry, contact transformations, and CR-structures.* // Arch. Math. — **38**, 1982, c. 1-5. ↑1

[18] Юмагужин В. А., Юмагужина В. Н. *Алгоритм вычисления алгебр изотропии уравнений* $y'' = a^3(x,y)y'^3 + a^2(x,y)y'^2 + a^1(x,y)y' + a^0(x,y)$ // Труды международной конференции "Программные системы: теория и приложения"ИПС РАН, Переславль-Залесский. — **2**, 2006, с. 365-377. ↑1

Институт программных систем Российской академии наук,
Россия, 152020, г. Переславль-Залесский, м. Ботик

Valeriy A. Yumaguzhin, Valeria N. Yumaguzhina. *Scalar differential invariants of equations* $y'' = a^3(x,y)y'^3 + a^2(x,y)y'^2 + a^1(x,y)y' + a^0(x,y)$ // Proceedings of Program Systems institute scientific conference “Program systems: Theory and applications”. — Pereslavl-Zalesskij, v. **1**, 2009. — p. 105–121. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. This paper is devoted to differential invariants of equations

$$y'' = a^3(x,y)y'^3 + a^2(x,y)y'^2 + a^1(x,y)y' + a^0(x,y)$$

w.r.t. point transformations. The action of the pseudogroup of all point transformations on the natural bundle of these equations is investigated. Tensor and scalar differential invariants of considered equations are constructed in this way. A complete collection of generators and their differential syzygies is obtained for the algebra of all scalar differential invariants of a wide class of considered equations containing generic equations.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 519.68+316.77

С. А. Амелькин, С. В. Знаменский

Информационная поддержка организации сложной совместной деятельности

Аннотация. Проблема организации информационной поддержки сложной совместной деятельности рассматривается на примере экспертизы научных проектов в конкурсах. Формулируются базовые требования, предъявляемые к информационной системе.

Проблема информационной поддержки организации сложной совместной деятельности возникла [1] в ходе реализации информационной системы учебных практик и научно-практических конференций Университета города Переславля имени А. К. Айламазяна. Алгоритмы решения такой проблемы могут использоваться в других видах деятельности, но наиболее рельефно она проявляется на примере экспертизы научных проектов.

1. Принципы эффективной экспертизы проектов

Постановка задачи. Рассмотрим конкурс научных проектов, объявленный некоторым фондом. На конкурс подано n проектов. Пусть целью конкурса является отбор таких $m < n$ проектов, которые наилучшим образом удовлетворяют условиям конкурса. Отбор проектов осуществляется k экспертами, каждому из которых предлагается для оценки l проектов ($kl > n$). Условия конкурса представляют собой набор критериев, степень соответствия которым для каждого из проектов оценивается экспертами.

Требуется определить:

(1) Алгоритм выбора проектов, предлагаемых каждому из экспертов.
(2) Методику проведения экспертизы, которая включает правило измерения соответствия проекта каждому из критериев, условия обмена информацией между экспертами и авторами проекта.
(3) Алгоритм сведения оценок экспертов в общую оценку и правило выбора m лучших проектов.

Работа выполнена при поддержке РФФИ, грант № 09-07-00407.

Процесс проведения экспертизы должен удовлетворять следующим взаимно противоречивым принципам:

1а. Принцип компетентности. Эксперт должен работать только с теми заявками, по которым он может составить компетентное мнение.

1б. Принцип независимости. Эксперт не должен быть связан с авторами проекта деловыми обязательствами (например, работать в одной и той же организации) во избежание конфликта интересов.

2а. Принцип равенства. Окончательное решение должно вырабатываться единовременно назначенными экспертами на равных основаниях.

2б. Принцип репутации. Влиятельность эксперта должна увеличиваться при активном и результативном участии в экспертизе проектов и, наоборот, падать при проявлениях некорректного, непрофессионального или пассивного поведения в ходе экспертизы. Авторы и эксперты с устойчиво низкой репутацией могут подпадать под дополнительные ограничения.

3а. Принцип оценки. Количество отобранных проектов может быть меньше m при низком общем уровне качества заявок, поэтому необходимо учитывать абсолютные показатели качества проектов.

3б. Принцип сравнения. Количество отобранных проектов не может быть больше m, даже при высоком общем уровне качества проектов. Поэтому на множестве всех проектов должно быть на основании оценок экпертов определено отношение полного строгого порядка. Это означает, что один и тот же эксперт не может выставить двум проектам одинаковые оценки.

Многочисленные существующие системы основаны на стратификации данных с использованием балльных шкал оценок, по которым эксперт с обоснованием квалифицирует заявку и которые являются основание для принятия решений руководящим органом. Если балльная шкала достаточно прозрачная и обоснованная, то по ней оказывается слишком много работ с одинаковой интегральной оценкой, что приводит к принятию окончательного решения без учета мнений специалистов, что приводит к произвольному решению задачи отбора проектов, а значит, к неоправданному повышению риска ошибки из-за отсутствия компетентности в далеких областях.

Для предотвращения возможности совпадения результатов экспертизы требуется ввести единую шкалу с достаточно большим (требуются тысячи) числом градаций итогового числового показателя.

Даже если кто-то создаст такую шкалу, наивно рассчитывать на то, что каждый эксперт ее глубоко изучит и безупречно правильно применит к каждой заявке. Получается, что использование такой шкалы резко повышает вероятность неудачного решения. Эксперт должен не только ставить семантически значимую абсолютную оценку проекту, но и сравнивать заявки, и мотивировать результаты сравнения. Поэтому оправдано использование сложных шкал оценки, сочетающих стратификацию и составление линейного порядка в каждой страте на основе заданного отношения предпочтения.

4а. Принцип анонимности. Участие эксперта в оценке должно быть надежно анонимным. Доступная авторам и другим экспертам информация не должна позволять вычислить эксперта.

4б. Принцип открытости. Максимально возможная часть механизма принятия решений должна быть открыта для прозрачного оперативного контроля правильности работы экспертами и заявителями в ходе рассмотрения заявки. После принятия решения корректность работы в целом должна быть доступна контролирующим органам в форме, удобной для проверки правильности учета мнений экспертов.

5а. Принцип согласия. Эксперты должны выработать общее согласованное с автором мнение о каждой заявке.

5б. Ограничение времени экспертизы. Процесс экспертизы проектов должен быть завершен в определенный срок. Продолжительность экспертизы не может быть слишком большой, так как в противном случае возникает неопределенность проведения исследовательских работ (отсутствие финансирования в начале проекта не позволяет начинать работы по проекту, затем, в случае положительного решения, сроки выполнения работ оказываются неоправданно сжатыми).

Экспертиза затрагивает интересы больших коллективов и принимающий решение эксперт должен быть надежно защищен от возможных попыток силового давления со стороны заявителей. Как бы тщательно ни прорабатывались формы заявок, неизбежны ситуации, в которых обмен информацией между экспертами или получение дополнительной информации от автора позволяет избежать грубой ошибки в оценке работы. Отсюда, в частности, вытекает, что традиционные подходы, основанные на сборе экспертных оценок с

последующей обработкой, характеризуются существенной вероятностью ошибки. В любой момент может всплыть информация, требующая немедленной корректировки результата.

Сформулированные принципы трудно совместимы в рамках единой системы. Например, открытость противоречит анонимности. Разумеется, контролирующие органы должны *a posteriori* иметь возможность выборочно раскрывать экспертов, но только оставляя об этом четкий след в системе.

В следующем параграфе будет показано, что очевидные противоречия между принципами могут быть по сути разрешены надлежащей системой информационной поддержки.

2. Схема организации экспертизы

Оставляя за рамками данной публикации подготовку условий конкурса, включающих требования к заявке и формирование корпуса экспертов, сосредоточимся на совместной выработке решения.

Начинается она с распределения работ между экспертами. Это должно делаться непредсказуемо, но с учетом возможностей экспертов, которые должны быть выяснены при получении от них согласия на работу.

Эти возможности должны ограничивать назначения как по количеству, так и по тематике, не допуская перегрузки эксперта и назначения ему выходящих за рамки его компетенции заявок.

2.1. Назначения на экспертизу

Для назначений на экспертизу известны два демократических подхода

- автоматический случайный выбор [2] на основе сопоставления профилей экспертов и заявок (профили могут включать в себя коды рубрик по рубрикатору проектов, ключевые слова и фразы, а также явное перечисление рассмотренных ранее проектов, наиболее близких по тематике);
- выбор заявок самими экспертами с последующим дораспределением неразобранных заявок.

Первый подход часто приводит к неудачным назначениям из-за несовершенства классификаций и неполноты наборов ключевых слов. Второй часто приводит к тому, что мнения эксперта объективно не

принимаются во внимание (например, если все выбрали 5 одинаковых заявок из 100).

(1) Распределение по профилям применяется для предварительного отбора статей с трехкратным (к примеру) перекрытием по количеству и как правило не включающий своих заявок и заявок от (в том числе бывших) соавторов и (в том числе бывших) сотрудников.
(2) Эксперту предлагается ранжировать заявки по уровню компетентности в них.
(3) Назначения выравниваются с максимально возможным учетом мнений эксперта (заявку придется рассмотреть даже если каждый эксперт нашел более близкие ему по тематике).

На сегодня этот подход представляется наиболее эффективным, но задача оптимального распределения заявок на экспертизу слишком сложна для того, чтобы говорить об окончательности решения.

2.2. Фазы экспертизы

Когда работам назначены эксперты, обязанные принять по ним окончательные решения, рассмотрение каждой заявки переходит в ознакомительную фазу, цель которой составить согласованное с автором объективное совместное заключение по каждой заявке, лаконично и полно отражающее как наиболее сильные и наиболее слабые ее стороны. Задача формулирования заключения решается организацией анонимного обмена информацией между экспертами и автором.

Ознакомительная фаза завершается утверждением согласованного заключения и заявка переходит в фазу принятия решений по заявкам.

Интерфейс эксперта должен помочь ему ценой минимальных усилий достичь согласия с другими экспертами и довести разделение своей части заявок до окончательного и согласованного. При этом интерфейс не должен подталкивать эксперта к необдуманным решениям.

2.3. Интерфейс эксперта

В качестве инструмента согласования экспертных мнений по статье предлагается интерфейс анонимной дискуссии, нацеленный на

(1) согласованное со всеми экспертами и автором экспертное заключение по каждой экспертируемой заявки, корректность отражения в котором наиболее сильных и слабых сторон проекта подтверждены автором и каждым экспертом;
(2) совместное решение экспертов по принятию или отказе.

Эксперту для этого должны быть доступны:

(1) Предложенные экспертами варианты заключений на те заявки, на которые данный дал свой вариант заключения.
(2) Возможность пометки заведомо некорректного (пристрастного, непрофессионального) заключения ("желтая карточка"; эксперт, получивший такую от половины других экспертов заявки отстраняется от экспертизы).
(3) Статистика пометок других экспертов на каждом заключении.
(4) Возможность написания замечаний по заключениям.
(5) Возможность отправить автору запрос на дополнительную информацию.
(6) Возможность просмотра полученной от автора дополнительной информации.
(7) Возможность изменить свое заключение либо выделить чужое заключение как лучшее.

Когда эксперты согласуют проект заключения по заявке, оно становится доступным автору с возможностью мотивированно оспорить (вдруг от экспертов ускользнула существенная особенность) или согласиться.

После получения согласия автора с заключением экспертов у эксперта добавляются возможности:

(8) Видеть текущий конкурс (отношение суммарного требуемого объема финансирования по заявкам, по которым не принято решение, к нераспределенной в данный момент сумме).
(9) Отметить заявку как одну из лучших, достойных поддержки (эксперт готов подписать положительное заключение).
(10) Отметить заявку как одну из худших, не достойных поддержки (эксперт готов подписать отрицательное заключение).

(11) После отметки заявки должно становиться видно наличие противоположных отметок и остаться возможность поменять свое мнение на противоположное. Многократное использование этой возможности должно отрицательно сказываться на репутации эксперта, компенсируя положительный эффект от достижения согласия по статье.

2.4. Сложность реализации

Сочетание перечисленных принципов в одной информационной системе является трудной задачей, не имеющей устойчивого решения. Любое найденное решение проявит свои недостатки и потребует перестройки системы. Поэтому интерфейс эксперта потребует внесения изменений и усовершенствований в ходе эксплуатации.

Требуется разработать устойчивую к таким перестройкам систему. Прежде, чем сформулировать требования к такой системе, рассмотрим общий план ее функционирования.

Неоднозначная формализуемость перечисленных принципов дает основание обобщить постановку задачи и вести речь о системе поддержки сложной совместной деятельности, которая в равной мере может быть ориентирована на решение этой или иных сходных задач.

2.5. Оценка и сравнение проектов

В процессе дискуссии между экспертами и автором проекта вырабатывается заключение о качестве данного проекта. Такое заключение, данное в словесной форме, требуется формализовать для того, чтобы иметь возможность сравнения проектов. Процесс анонимной дискуссии позволяет получить требуемую информацию о качественных характеристиках каждого проекта, возможность организации дискуссии, позволяющей сравнить проекты, сомнительна. Действительно, в процессе дискуссии не предусмотрено согласование позиций авторов нескольких проектов, кроме того, вероятность получения согласованного мнения экспертов убывает с увеличением числа экспертов. Вместе с тем, наличие небольшого числа экспертов, оценивающих один и тот же проект, не позволяет ввести отношения предпочтения. Поэтому, задача формализации оценки и сравнения проектов может быть решена в три этапа:

- Определение оптимального числа экпертов, оценивающих один проект, а также среднего количества проектов, предлагаемых одному эксперту;
- Построение пространства оценок проектов, введение на этом пространстве метрики и отношения предпочтения;
- Определение средней оценки по множеству экспертов для каждого проекта. Заполнение матрицы парных сравнений проектов, проверка на транзитивность. Составление линейного порядка на множестве проектов.

Такой алгоритм аналогичен алгоритму восстановления оценок объектов методами коллаборативной фильтрации [3].

3. Требования к информационной системе

Базовая функциональность

- Персонализация бизнес-процесса: все относительно важные текущие дела каждого сотрудника, вплоть до участия в открытых обсуждениях, должны быть всегда на виду. Индикация важности дела должна немедленно отражать изменения приоритетов руководства, отмены заданий, завершение работ, изменение состава рабочей группы или роли в ней пользователя и изменение приоритетов конкретного контекста для пользователя.
- Стандарты качества ISO 9000-9001 и ГОСТ Р ИСО 15489: ролевое обсуждение и корректный учет всех мнений при принятии решений.Любому решению предшествует протоколируемый обмен открытыми и порой конфиденциальными сообщениями в ходе ролевой дискуссии и прозрачное автоматическое сведение всех выраженных оценок в новый статус документа.
- Трекинг вклада каждого сотрудника: общая динамика активности и результативности участников, а также точный учет и формальная оценку кто и насколько задерживает обсуждение и другие дела.

Фундаментальная проблема

Структура, содержание деятельности и отчасти даже цели организации непредсказуемо изменчивы и потому ценность различных фрагментов текущей информации для будущей деятельности невозможно сравнивать.

Отсюда необратимые потери доступа к неожиданно вновь оказавшейся ценной информации. Масштабы потерь возрастают с усилением защиты от несанкционированного доступа. Особые требования:

(1) Полная ретроспективность: любой запрос может быть дан с дополнительным указанием (прошедшего) момента времени и ответ на него должен быть ровно таким, каким он был бы на указанный момент. Разумеется, в контекстах с ограниченным доступом ретроспективный запрос также ограничивается в соответствии с сегодняшним статусом пользователя. Изменить прошлое невозможно ("парадокс времени").
(2) Контекстная автономность: каждый организационный контекст (т.е. вся организация, любое подразделение, комиссия, совет, рабочая группа, класс проектов, конкретное дело или мероприятие, требующее ролевого взаимодействия различных сотрудников) допускает произвольные изменения структуры и обрабатывающего программного кода. Побочные эффекты на результатах обработки запросов, не связанных с этим контекстом, недопустимы.

В рамках контекстно автономной системы любые подразделения могут при полном сохранении информационной поддержки дробиться, сливаться, переподчиняться, обретать или терять самостоятельность, открываться или засекречиваться и произвольно перепрофилироваться. На поддержке остальных подразделений это никак не будет сказываться.

Система может взаимодействовать с внешними системами и сервисами.

В модульной системе организация или рабочая группа обычно использует различные модули (почта, чат, вики, календарь, расписание, ...), любые изменения в ролевом доступе требуется производить в нескольких модулях. Если система контекстно-автономна, то сложность точного своевременного внесения связанных изменений в каждый модуль концентрируется в рамках контекстов. Администрирование доступа становится прозрачным и действенным.

4. Заключение

Контекстно-автономная система *«edu.botik»* разрабатывается силами студентов УГП имени А. К. Айламазяна [4] и проходит опытную эксплуатацию на базе Переславского научно-образовательного

комплекса. Работающий прототип успешно используется для поддержки перекрестного рецензирования работ, представляемых на научные конференции. [5, 6].

Список литературы

[1] Знаменский С. В. *Хорошо масштабируемое автономное администрирование доступа* // Тр. междунар. конф. "Программные системы: теория и приложения Переславль-Залесский, октябрь 2006. — Т. **1**. — М.: Наука,-Физматлит, 2006, с. 155-169. ↑(document)

[2] Степанов Д. Н. *Алгоритм назначения рецензентов как часть проведения научных конференций при поддержке информационной системы UPIS* // Международная конференция "Программные системы: теория и приложения Переславль-Залесский, апрель 2008. — Т. **1**. — Переславль-Залесский: изд.-во «Университет города Переславля», с. 155-169. ↑2.1

[3] Wang J., de Vries A., Reinders M. J. T. *Unifying Userbased and Itembased Collaborative Filtering Approaches by Similarity Fusion*. — Seattle, Washington, USA, 2006. ↑2.5

[4] Знаменский С. В., Живчикова Н. С., Титова Е. В. Описание системы *«Ботик»*. — Переславль-Залесский, 2009, доступ: http://wiki.botik.ru/IS4UGP/. ↑4

[5] *Сайт Международной конференции «Программные системы: теория и приложения»*. — Переславль-Залесский, 2009, доступ: http://edu.botik.ru/psta2009/. ↑4

[6] *Сайт Молодежной научной конференции «Наукоемкие информационные технологии»*. — Переславль-Залесский, 2009, доступ: http://edu.botik.ru/sit2009/. ↑4

Исследовательский центр системного анализа ИПС РАН

S. A. Amelkin, S. V. Znamenskij. *Informational Support of Complex Collaboration* // Proceedings of Program Systems institute scientific conference “Program systems: Theory and applications”. — Pereslavl-Zalesskij, v. **1**, 2009. — p. 123–132. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. Scientific project expertise activity is an example of complex collaboration activity. The problem of informational support of this activity is considered. The basic requirements for the information system are proposed.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 658.562.6:04.032.26 (06)

Ю. Г. Емельянова, А. А. Талалаев, В. П. Фраленко,
В. М. Хачумов

Нейросетевой метод обнаружения неисправностей в космических подсистемах

Аннотация. Рассмотрены вопросы контроля космических подсистем на основе нейросетевого подхода. Обнаружение неисправности сведено к задаче распознавания двух классов «исправно», «неисправно». Проведен сравнительный анализ качества контроля с применением персептрона и вероятностной нейронной сети (ВНС). Показано, что ВНС достаточно уверенно разделяет ситуацию на два класса и может быть использована для решения задачи.

1. Введение

Вопросам обеспечения работоспособности космических подсистем за счет комплексных средств контроля и диагностики уделяется самое серьезное внимание [1]. Под диагностикой обычно понимают установление технического состояния объектов, выявление дефектов и измерение их параметров, обнаружение и прогнозирование развития нештатных ситуаций (НШС). Естественно, речь идет об измерении параметров способами, не ухудшающими последующую эксплуатационную пригодность и надежность подсистемы. Эффективность методов контроля и диагностики зависит от используемой концептуальной модели предметной области. Применительно к объектам космической отрасли диагностика может рассматриваться, например, как измерение параметров текущего состояния объекта, определение ситуации и прогнозирование надежности работы подсистем на основе знаний экспертов [2].

Целью настоящей работы является обнаружение отклонений в работе бортовых и наземных станциях командно–измерительных систем (НС КИС) на основе искусственных нейронных сетей (ИНС).

Работа выполнена при финансовой поддержке Программы Союзного государства «Космос–НТ» (проект «Нейросеть»).

2. Постановка задачи

Сформулируем задачу обнаружения отклонений в работе технической системы как задачу распознавания. Задача распознавания [3] формулируется следующим образом. Пусть дано множество M объектов $\{\omega_i\}$; на этом множестве имеется разбиение на конечное число подмножеств (классов) Ω_k, $k = \overline{1,m}$, $\bigcup\limits_{k=1}^{m} \Omega_k = M$. Каждый класс Ω_k имеет внутреннюю структуру, например, в виде некоторого множества объектов–эталонов. Объекты задаются значениями некоторых признаков x_j, $j = \overline{1,N}$ (этот набор всегда один и тот же для всех объектов, рассматриваемых при решении задачи). Совокупность значений признаков x_j определяет описание каждого объекта $I(\omega) = \{x_1, x_2, \ldots, x_N\}$. Информация о вхождении некоторого объекта ω в какой-либо класс представляется в виде информационного вектора $I(\omega) = \{I_1(\omega), I_2(\omega), \ldots, I_m(\omega)\}$, где $I_k(\omega)$ несет информацию о принадлежности объекта ω к классу Ω_k:

$$I_k(\omega) = \begin{cases} 1, \omega \in \Omega_k, \\ 0, \omega \notin \Omega_k, \\ -, \text{ если неизвестно } \omega \in \Omega_k \text{ или } \omega \notin \Omega_k. \end{cases} \tag{1}$$

Решение о принадлежности объекта ω классу Ω_k принимается на основе сравнения расстояний между объектом и классами. К мерам расстояний при этом не предъявляется жестких требований. К предложенной Журавлевым Ю.И. [3] схеме распознавания можно свести множество возможных постановок задач контроля и диагностики.

Покажем это на примере постановки задачи контроля сеанса связи центра управления полетом (ЦУП) с Абонентом, в качестве которой выступает НС КИС. При взаимодействии ЦУП с Абонентом происходит обмен директивами. Инициатором выдачи директив может быть только ЦУП. Все директивы и инициативные сообщения (ИС) квитируются получателем. Инициативные сообщения и квитанции выдаются Абонентом на фоне передаваемой информации функционального контроля (ИФКТ) с тактом 1 сек. В поле заголовка квитанции Code вводится код квитанции, в поле NPrm — номер последнего принятого пакета. В структуре информации обратного канала (ИОК) в поле Code содержится одно из трех значений: 1, 2 или 0.

Целью контроля является определение аварийной ситуации при выполнении набора правил. Алгоритм выявления НШС — потери

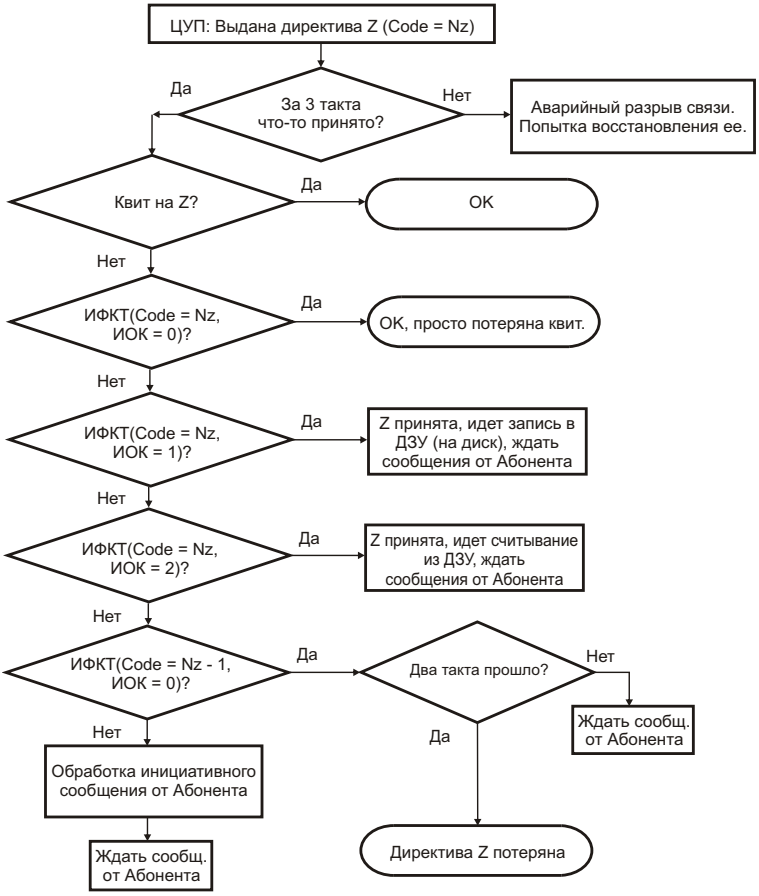


Рис. 1. Алгоритм слежения ЦУП за прохождением данных

связи с ЦУП представлен на рис. 1. Код выданной директивы обозначен как Z, а код квитанции — как N_z.

Представленный алгоритм, предполагает одновременное измерение значений ряда признаков и их проверку на соответствие определенным условиям, выполнение или невыполнение которых кодируется соответственно как «1» и «0». Для дальнейшего анализа ситуаций удобно формализовать алгоритм в виде табл. 1.

Таблица 1. Формализация алгоритма контроля сеанса связи ЦУП с Абонентом (НС КИС)

№ п/п	Наименование ситуации	Признаки								Номер класса
		1	2	3	4	5	6	7	8	
		В течение 3-х тактов принято сообщение	Получена квитанция на Z	Code $= 0$, NPrm $= 0$	Code $= N_Z$, ИОК $= 0$	Code $= N_Z$, ИОК $= 1$	Code $= N_Z$, ИОК $= 2$	Code $= N_Z$-1, ИОК $= 0$	Число пройденных тактов ≥ 2	
1	Связь только что установлена	1	0	1	0	0	0	0	0	1
2	Нормальная передача	1	1	0	0	0	0	0	0	1
3	Z принята, идет запись на диск, ждать сообщения от Абонента	1	0	0	0	1	0	0	0	1
4	Z принята, идет считывание с диска ждать сообщения от Абонента	1	0	0	0	0	1	0	0	1
5	Ожидание сообщения от Абонента	1	0	0	0	0	0	1	0	1
6	Обработка инициативного сообщения от Абонента	1	0	0	0	0	0	0	0	1
7	Аварийный разрыв связи	0	0	0	0	0	0	0	0	0
8	Потеряна квитанция	1	0	0	1	0	0	0	0	0
9	Директива Z потеряна	1	0	0	0	0	0	1	1	0

Каждая строка табл. 1 описывает прецедент с известным исходом. В соответствии с таблицей прецедентов имеем два класса «1» — нет потери связи, «0» — связь потеряна. Заметим что:

- табл. 1 содержит лишь выборочные данные о процессе, т.е. не является полной;
- при распознавании часть признаков может быть неизвестна, т.е. принимать значение «–».

Задача решается при следующих условиях:

- предполагается, что выпадение значений «0» или «1» для всех признаков равновероятно;
- если на входе величина признака неизвестна (не задана) то на вход подается величина равная 0.5;
- если величина признака в эталонном векторе не задана (не существенна), то вероятность совпадения признака на входе и в эталоне принимаем равной 1;
- для удобства работы с ИНС введены два порога: «0» соответствует порог 0.25, а «1» — 0.75.

Таким образом, имеем соответствие условиям задачи (1) и, следовательно, можем формулировать задачу определения неисправности как задачу распознавания. Далее следует проверить, может ли ИНС быть обучена на основе данных таблицы (обучающей выборки) для выявления одного из двух возможных исходов.

3. Исследование возможностей нейронных сетей

Для решения задачи (1) целесообразно использовать принцип логико–вероятностного распознавания, реализуемый на основе соответствующей вероятностной нейронной сети (ВНС) [4]. ВНС требует для своего обучения знания параметров плотности распределения вероятности: математических ожиданий значений признаков и дисперсии. Основные идеи обучения ВНС изложены в работе [5]. С другой стороны задача логического распознавания ситуации может быть решена однослойным или двухслойным персептроном. Возможности обучения ИНС логическим функциям генетическим алгоритмом описаны, например, в работе [6].

3.1. Вероятностная нейронная сеть

Для правильного функционирования вероянтостной нейронной сети (ВНС) необходимо найти статистические параметры распределения вероятности [1]. С этой целью воспользуемся данными табл. 1.

Определяем вектор математических ожиданий M_p признаков для каждого класса P обучающей выборки (средние значения признаков):

$$M_p^i = \frac{\sum\limits_{k=1}^{Q_p} P_k^i}{Q_p}, \tag{2}$$

где: M_p^i — математическое ожидание i-го признака эталонных векторов класса P; Q_p — число эталонных векторов в классе P; P_k^i — i-ый признак эталонного вектора с номером k из класса P.

Находим значения дисперсий D_A:

$$D_A = \frac{\sum\limits_{i=1}^{h} \left(M_p^i - A_i\right)^2}{h}, \tag{3}$$

где: h — число признаков эталонов обучающей выборки; A_i — i-ый признак эталонного вектора A.

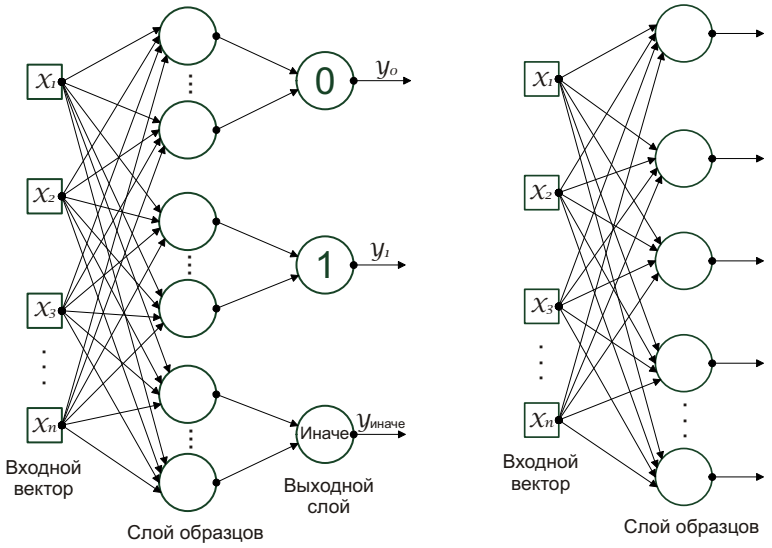
Рассмотрим ВНС «с усреднением» (рис. 2а), у которой расстояние от подаваемого на вход вектора $X\,(x_0, \ldots, x_n)$ до класса P есть среднее расстояние до эталонных векторов этого класса.

Функция активации имеет следующий вид:

$$f_p = \frac{\sum\limits_{k=1}^{Q_p} S_{P_k}}{Q_p}. \tag{4}$$

f_p —вероятность того, что входной вектор A принадлежит к классу P, S_{P_k} вычисляется по формуле $S_{P_k} = \exp\left(\frac{\sum\limits_{i=1}^{h}\left(P_k^i - A_i\right)^2}{-2D_{P_k}h}\right)$, где D_{P_k} — дисперсия эталонного вектора с номером из класса P.

Альтернативный подход к построению ВНС не предполагает измерения среднего расстояния до эталонных векторов. Соответствующая структура сети представлена на рис. 2b.



a) с усреднением b) без усреднения

Рис. 2. Архитектура вероятностной сети

Здесь функция активации f_p вычисляется по следующей формуле:

$$f_p = \max_{k=1\ldots Q_P} \{S_{P_k}\}. \tag{5}$$

На вход нейронной сети подается вектор $X(x_0, \ldots, x_n)$, а на выходе получаем вектор $Y(y_0, \ldots, y_m)$ в соответствии с необходимым преобразованием $X \to Y$. Номер j, для которого выход y_j максимален, соответствует номеру класса.

3.2. Однослойный и двухслойный персептроны

Персептрон работает с входными векторами из чисел 0.25, 0.75 и 0.5, которые соответствуют нулю, единице и ситуации когда величина признака неизвестна. Персептрон представляет собой однослойную нейронную сеть (рис. 3), в которой количество нейронов равно числу классов. Альтернативой служит двухслойная полносвязная ИНС прямого распространения (двухслойный персептрон), представленная на рис. 4.

Функция активации однослойного персептрона имеет вид линейного скачка:

$$f(s) = \begin{cases} 1, \text{ если } s \geq 1, \\ (s+1)/2, \text{ если } -1 < s < 1, \\ 0, \text{ если } s \leq 1. \end{cases} \tag{6}$$

Настройка сети осуществляется по известному методу Видроу–Хоффа [4].

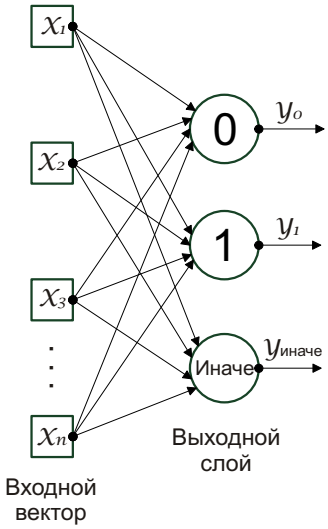


Рис. 3.
Однослойный персептрон

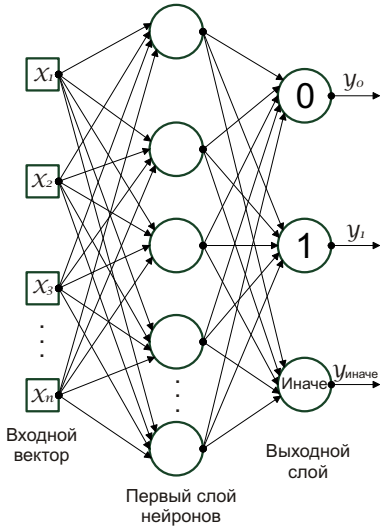


Рис. 4.
Двухслойный персептрон

Обучение двухслойной сети проводилось методом обратного распространения ошибки с активационной функцией вида сигмоид (логистическая функция): $f(s_i) = \frac{1}{1+e^{-s_i}}$, где s_i рассматривается как взвешенная сумма всех входов нейрона i второго слоя. Для всех сетей прямого распространения результат определяют по максимальному выходному сигналу.

3.3. Результаты экспериментов

Помимо данных табл. 1 для обучения были использованы также другие вектора в соответствии с алгоритмом (рис. 1), у которых «–»-значение может быть «0» или «1». Дополнительные вектора приведены в табл. 2.

Таблица 2. Дополнительные обучающие вектора

Наименование ситуации при выполнении обработки инициативного сообщения от Абонента	Признаки	Номер класса: 1 — норма, 0 — авария
Связь только что установлена	1 0 – 0 0 0 0 0	1
Нормальная передача	1 – 0 0 0 0 0 0	1
Z принята, идет запись на диск, ждать сообщения от Абонента	1 0 0 0 – 0 0 0	1
Z принята, идет считывание с диска, ждать сообщения от Абонента	1 0 0 0 0 – 0 0	1
Ожидание сообщения от Абонента	1 0 0 0 0 0 – 0	1

Нейронные сети после обучения на данных табл. 1 и табл. 2 были протестированы как на данных из обучающей выборки, так и на случайных векторах, определенных на полном множестве всех возможных перестановок «0», «1» и «–». Полученные результаты отражены в сводной табл. 3.

Из табл. 3 видно, что однослойный персептрон в целом не справился с поставленной задачей. Двухслойный персептрон, обучаясь лишь на эталонных векторах, однозначно выделяет их из всего множества. Наилучшие показатели у ВНР без усреднения, в то время как сеть с усреднением не справилась со вторым режимом работы.

Таблица 3. Результаты экспериментов по обучению ИНС и распознаванию ситуаций

Тип ИНС	Режим тестирования ИНС			
	Режим 1 (выходы 0, 1)		Режим 2 (выходы 0, 1, «иначе»)	
	Успешное обучение по табл. 1 и табл. 2	Наличие ошибок распознавания эталонных и случайных векторов	Успешное обучение на полном наборе векторов	Наличие ошибок распознавания эталонных и случайных векторов
Вероятностная сеть без усреднения	Да	Нет	Да	Нет
Вероятностная сеть с усреднением	Да	Да	Да	Да
Однослойный персептрон	Да	Да	Нет	Да
Двухслойный персептрон	Да	Нет	Нет	Да

4. Заключение

В настоящей работе рассмотрена задача обнаружения неисправностей в функционировании космических подсистем как составная часть задачи диагностики. Экспериментально показано, что ИНС способны достаточно уверенно разделять ситуацию на два класса («исправно», «неисправно»), т.е. фактически решать задачу контроля. Техническая диагностика подсистем подразумевает выделение существенно большего числа классов. Однако есть уверенность в том, что и она может быть решена по приведенной схеме: построение таблиц прецедентов, обучение ИНС, распознавание.

Список литературы

[1] Большая космическая энциклопедия. — http://kosmos.claw.ru/. ↑1, 3.1

[2] Смирнов С. В. Анализ и прогнозирование надежности аппаратуры наземных станций командно–измерительных систем. Авиакосмическое приборостроение, №1, 2008. — 27-32 с. ↑1

[3] Об алгебраическом подходе к решению задач распознавания или классификации. Проблемы кибернетики. — Т. **33**, 1978. — 5-68 с. ↑2, 2

[4] Каллан Р. Основные концепции нейронных сетей. — М.: Издательский дом «Вильямс», 2001. — 288 с. ↑3, 3.2

[5] Specht D. F. Probabilistic neural networks for classification, mapping or associative memory. In Proceedings of IEEE international conference neural networks (vol. 1), 1988. — 525-532 с. ↑3

[6] Хачумов В. М. Логические элементы на нейронах. Материалы IX Международной конференции «Интеллектуальные системы и компьютерные науки», - том 1, часть 2. — М.: Издательство механико-математического факультета МГУ, 2006. — 297-300 с. ↑3

Исследовательский центр искусственного интеллекта ИПС РАН

Институт системного анализа РАН

J. G. Emelynova, A. A. Talalaev, V. P. Fralenko, V. M. Khachumov. *Failure detection in space subsystems based on artificial neural networks* // Proceedings of Program Systems institute scientific conference “Program systems: Theory and applications”. — Pereslavl-Zalesskij, v. **1**, 2009. — p. 133–143. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. Some issues of space subsystems control based on artificial neural network are considered in the paper. Failure detection is simply a problem of recognition of two classes: "correct" and "failure". Implying perception and a Probabilistic Neural Network (PNN) the comparative analysis of quality of the control is carried out. It has been established that PNN confidently enough divides the situation into two classes and can be used in problem solving.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 004.89

А. Е. Пинжин

Реализация системы логического вывода на основе структурных функциональных моделей для ряда логических исчислений

Аннотация. Рассматривается подход к приведению Хорновских правил, выраженных на языках ряда логических исчислений (логики высказываний, логики первого порядка и дескриптивной логики) к конструкциям теории структурных функциональных моделей. Применение предлагаемого способа интерпретации позволяет использовать алгоритм вывода на структурых функциональных моделях для доказательства ряда логических теорем в перечисленных логиках. Представлены результаты экспериментального сравнения производительности с существующими алгоритмами.

1. Введение

В настоящее время наблюдается значительный интерес к методам интеллектуальной обработки информации. Среди этих методов весомое место занимают алгоритмы, обеспечивающие дедуктивное доказательство логических теорем на основе заданного набора высказваний. Несмотря на длительную историю развития и накопленный опыт в этой области, эффективность алгоритмов логического вывода остается актуальной проблемой. Подход, представленный в [1], предлагает альтернативный алгоритм вывода на основе теории структурных функциональных моделей (С-моделей) [2]. Первоначально этот алгоритм разрабатывался с целью синтеза структурных программ. В [3] показано, каким образом функциональные связи С-модели могут быть интерпретированы в виде дизъюнктов Хорна. Основной целью настоящей статьи является обоснование возможности трансформации задач вывода, сформулированных на базе С-моделей, в логические теоремы классической логики высказываний (ЛВ), логики первого порядка (ЛПП), а также дескриптивной логики (ДЛ). Представлено общее описание реализации машины вывода на С-моделях и результаты опытного сравнения производительности с некоторыми существующими алгоритмами вывода на перечисленных логиках.

2. Основные определения

Исходные данные для машины вывода на С-модели поставляются в виде набора схем. Для упрощения рассуждений будем рассматривать простые схемы с подсхемами без вариантной части и рекурсивных вхождений (более подробная информация представлена в [2]).

Определение 1. *Структурной функциональной моделью* называется конечная совокупность схем вида $M = (T_1, \ldots, T_s)$.

Определение 2. *Простой схемой* T объекта t будем называть выражение вида

$$T(t) = (T_0(a_0), T_1(a_1), \ldots, T_n(a_n)|fset), \tag{1}$$

где T – имя схемы, $a_0, a_1, \ldots, a_n$ – собственные атрибуты (величины) схемы T; $T_0, T_1, \ldots, T_n$ – собственные подсхемы схемы T, определяющие тип соответствующих им атрибутов.

Внутренние атрибуты схемы T, принадлежащие атрибуту $t : T$, выражаются записью $t.a_0, \ldots, t.a_n$. Каждый тип атрибута, встречающийся в определениях схем С-модели M, должен являться примитивным (например, целое число, строка), либо соответствовать схеме, принадлежащей M. Выражение $fset$ в завершающей части описания схемы скрывает множество функциональных связей схемы T.

Определение 3. *Функциональная связь (ФС)* – это выражение вида $f : a_1, \ldots, a_n \to a_0$, где f – имя, $a_1, \ldots, a_n$ – аргументы, a_0 – результат ФС.

Необходимым ограничением для ФС является то, что атрибуты $a_0, a_1, \ldots, a_n$ должны являться собственными атрибутами схемы T. Постановка задачи планирования в С-модели M может быть представлена следующим образом:

$$S = (T(t), A_0, X_0), \tag{2}$$

где t – непервичный атрибут схемы $T \in M$, на которой ставится задача, A_0 и X_0 – наборы имён соответственно исходных и искомых атрибутов, принадлежащих T.

3. Интерпретация С-модели

Традиционная интерпретация I С-модели M, содержащей простые схемы, построенные согласно (1), задает:

для каждой элементарной схемы $T^E \in M$ первичный тип T^E_I;

для каждой ФС $f \in T$ – некоторое отображение $f_I : T_{1I} \times \cdots \times T_{nI} \to T_{0I}$.

Согласно методу интерпретации спецификаций, приведенному в [3], иерархия вложенных атрибутов может быть представлена следующим образом:

$$T(t) \leftarrow t.(T_{01}(a_{01}), \ldots, T_{0n}(a_{0n})); \quad t.T_{01}(a_{01}) \leftarrow t.a_{01}.(T_{11}(a_{11}), \ldots, T_{1m}(a_{1m})); \ldots, \tag{3}$$

где все $T_{ij} \in E$. Переход к примитивным типам позволяет задать интерпретации для различных логических исчислений.

3.1. Логика высказываний

Каждой примитивной величине a_{ij} ставится в соответствие примитивное высказывание A_{ij}, а функциональный символ интерпретируется как знак логического следствия. Тогда множество ФС схемы может быть представлено в виде следующего выражения:

$$A_0 \leftarrow A_{01} \wedge \cdots \wedge A_{0n}, \ldots, A_k \leftarrow A_{k1} \wedge \cdots \wedge A_{kn},$$

т.е. в виде набора Хорновских дизъюнктов:

$$A_0 \vee \neg A_{01} \vee \ldots \vee \neg A_{0n}, \ldots, A_k \vee \neg A_{k1} \vee \ldots \vee \neg A_{kn}. \tag{4}$$

Задача планирования (2), при $A_0 = \{a_1, \ldots, a_i\}$, $X_0 = \{a_{i+1}, \ldots, a_j\}$, для постановки задачи проверки выполнимости набора высказываний (sat-problem) интерпретируется как $X_0 \leftarrow A_0$ и, совместно с (4), соответствует:

$$A_1, \ldots, A_i, \neg A_{i+1}, \ldots, \neg A_j. \tag{5}$$

Приведенная интерпретация позволяет использовать алгоритмы вывода на С-модели [1] для автоматического доказательства весьма обширного множества теорем.

3.2. Логика первого порядка

Во многих классических работах, например [4], описывается метод приведения вывода на логике первого порядка (ЛПП) к выводу в логике высказываний. Приведем некоторый набор высказываний ЛПП, находящихся в стандартной форме и сформулируем задачу вывода:

$$\forall z(A_0(z) \leftarrow A_{01}(z) \wedge \cdots \wedge A_{0n}(z), \ldots, A_k(z) \leftarrow A_{k1}(z) \wedge \cdots \wedge A_{kn}(z), A_1(z), \ldots, A_i(z), \neg A_{i+1}(z), \ldots, \neg A_j(z)). \tag{6}$$

Вполне очевидно, что пропозиционализация этих высказываний приводит теорему к виду (4), (5), а значит и к (2).

3.3. Дескриптивная логика

Интерпретация задачи вывода для дескриптивной логики (ДЛ) представляет интерес, в частности, в связи с интенсивным развитием парадигмы Semantic Web, где дескриптивная логика находит широкое применение. В ряде источников, например [5], приводится сопоставление логики первого порядка и дескриптивной логики (ДЛ). Известно, что выражения ДЛ могут быть приведены к высказываниям ЛППП, если они не вовлекают в предикаты более двух переменных. В таблице 1 приведены некоторые базовые правила трансформации.

Таблица 1. Соответствие выражений ДЛ и ЛППП

ДЛ	ЛППП
$\top$	T
$\bot$	F
R	$R(x)$
$R \subseteq S$	$\forall x(R(x) \leftarrow S(x))$
$R \cup S$	$\forall x(R(x) \vee S(x))$
$R \cap S$	$\forall x(R(x) \wedge S(x))$

Хорновские правила в ДЛ записываются в виде аксиом вида:

$$A_0 \subseteq A_{01} \cap \cdots \cap A_{0n},$$

что эквивалентно следующему выражению ЛППП:

$$A_0(x) \leftarrow A_{01}(x) \wedge \cdots \wedge A_{0n}(x).$$

Задача категоризации в ДЛ: $A_0 \subseteq A_{ij}-?$ аналогичным образом трансформируется в выражение ЛППП. Все это в совокупности позволяет сформулировать следующую постановку проблемы (6) в виде правил ДЛ:

$$A_0 \subseteq A_{01} \cap \cdots \cap A_{0n}, \ldots, Ak \subseteq Ak1 \cap \cdots \cap Akn,$$
$$A_1(z) \equiv \top, \ldots, A_i(z) \equiv \top, A_{i+1}(z) \equiv \bot, \ldots, A_j(z) \equiv \bot. \tag{7}$$

4. Экспериментальные результаты

Реализация машины вывода на С-модели подробно описана в [1]. Приведем здесь лишь краткое описание основных модулей программы (рис. 1).

Для целей сопоставления производительности были разработаны конвертеры формата данных С-модели в различные форматы [6–9], используемые машинами вывода, выбранными для опытного сравнения. В таблице 2 приведен список реализаций машин вывода, вид логики и формат исходных данных.

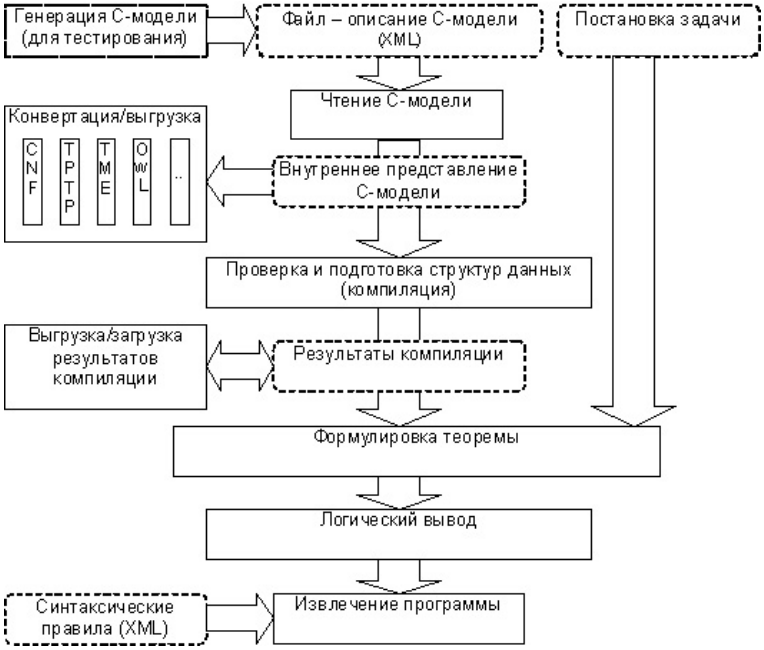


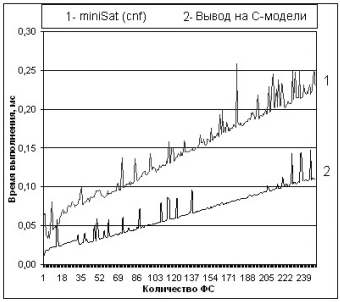
Рис. 1. Структура машины вывода на С-модели

Поясним выбор машин вывода. Пакет Sat4j содержит реализацию алгоритма miniSat, показавшего максимальную эффективность согласно итогам конкурса по решению SAT-проблемы в 2006 г. [10]. Darwin, Paradox, iProver являются номинантами конкурса CADE ATP System Competition (CASC) 2007, проводимого в рамках ежегодной конференции по автоматическому логическому выводу [11]. Отметим, что Vampire 8.0/9.1 не участвует по причине закрытого описания входных параметров. Pellet считается одной из наиболее эффективных свободно распространяемых машин вывода на ДЛ [12].

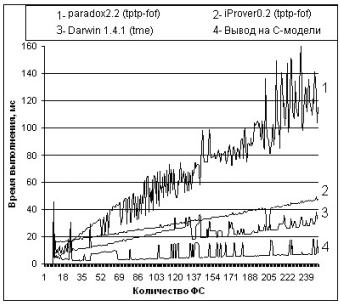
Таблица 2. Форматы данных машин вывода

Машина вывода	Логическое исчисление	Формат данных
miniSat (sat4j)	ЛВ	cnf
Darwin 1.4.1	ЛППП	tme
paradox2.2	ЛППП	tptp-fof
iProver0.2	ЛППП	tptp-fof
Pellet	ДЛ	owl-rdf

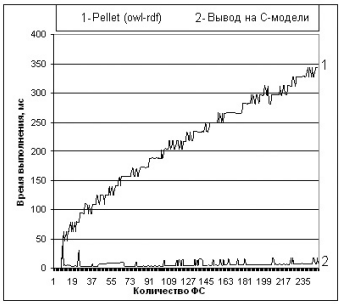
Для целей сравнения выполнялась генерация и конвертация схем с возрастающим от 10 до 250 числом ФС. Аргументы и цели ФС связывались таким образом, чтобы обеспечить необходимость обработки всех элементов при осуществлении вывода. Результаты испытаний приведены на рис. 2.



а)



б)



в)

Рис. 2. Сравнение производительности машин вывода (а – ЛВ, б – ЛППП, в – ДЛ)

Отметим, что при измерении времени работы алгоритмов на ЛПП и ДЛ учитывается время загрузки исходных данных (согласно требованиям CASC [11]), поэтому на рис. 2(а) результаты работы алгоритма вывода на С-модели отличаются от аналогичных результатов рис. 2(б) и рис. 2(в).

5. Заключение

Безусловно, приведенные результаты испытаний нельзя считать точным показателем относительной производительности алгоритмов. Основной целью проведения тестов являлось обоснование того, что предлагаемый алгоритм может быть использован для вывода в различных логиках, показывает сопоставимые показатели эффективности и способен составить конкуренцию существующим реализациям.

Особо следует отметить, что в расчет издержек не включались затраты на представление исходной модели в виде специальных структур данных (компиляция схем), на которых основан алгоритм вывода на С-моделях. Эти структуры подготавливаются однократно и могут быть использованы для постановки любых задач, допустимых в данной модели, однако возможность внесения изменений в подготовленные структуры данных остается предметом дальнейших исследований. Из этого следует, что на текущий момент практическое применение предлагаемого алгоритма структурного вывода обосновано и эффективно для достаточно устойчивых моделей.

Список литературы

[1] Новосельцев В. Б., Пинжин А. Е. *Реализация эффективного алгоритма синтеза линейных функциональных программ* // Известия Томского политехнического университета. — Т. **312**, № 5, 2008, с. 32–35. ↑1, 3.1, 4

[2] Новосельцев В. Б. *Теория структурных функциональных моделей* // Сибирский математический журнал. — Т. **47**, № 5, 2006, с. 1014–1030. ↑1, 2

[3] Новосельцев В. Б. *Эффективный нерезолютивный вывод для ограниченного исчисления хорновских дизъюнктов* // Известия Томского политехнического университета. — Т. **312**, № 5, 2008, с. 94–97. ↑1, 3

[4] Рассел С., Норвиг П. Искусственный интеллект: современный подход (AIMA). — 2-е изд.: Вильямс, 2007. — 381–384 с. ↑3.2

[5] Baader F., Calvaneze D., McGuiness D. L., Nardi D., Patel-Schneider P. F. The Description Logic Handbook: Theory, Implementation, and Applications: Cambridge University Press, 2003. — 154–156 с. ↑3.3

[6] SAT DIMACS Challenge – Satisfiability: Suggested Format: Электронный ресурс, 1993, Режим доступа: ftp://dimacs.rutgers.edu/pub/challenge/satisfiability/doc/satformat.tex. ↑4

[7] TME input specification: Электронный ресурс, Режим доступа: http://www.uni-koblenz.de/ag-ki/Systems/Protein/tme-syntax.txt. ↑

[8] TPTP syntax v3.5.0: Электронный ресурс, Режим доступа: http://www.cs.miami.edu/~tptp/TPTP/SyntaxBNF.html. ↑

[9] OWL Web Ontology LanguageGuide: Электронный ресурс, 2004, Режим доступа: http://www.w3.org/TR/owl-guide/. ↑4

[10] SAT-Race 2006: Runtime comparison of all SAT-Race solvers: Электронный ресурс, 2006, Режим доступа: http://fmv.jku.at/sat-race-2006/analysis.html. ↑4

[11] The CADE ATP System Competition, The 21st International Conference on Automated Deduction: Электронный ресурс. — Bremen, Germany, 17, Режим доступа: http://www.cs.miami.edu/~tptp/CASC/21/. ↑4, 4

[12] Sirin E., Parsia B., Grau B. C., Kalyanpur A., Katz Y. *Pellet: A Practical OWL-DL Reasoner*, № CS 4766. — University of Maryland, College Park, USA, 2005. ↑4

Томский Политехнический Университет

A. E. Pinzhin. *Realization of a reasoner based on structural functional models for several types of logical calculus* // Proceedings of Program Systems institute scientific conference “Program systems: Theory and applications”. — Pereslavl-Zalesskij, v. **1**, 2009. — p. 145–152. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. Presented an approach for transformation of Horn clauses, expressed in several logical calculus (propositional logic, first-order logic and descriptive logic) into structural functional models theory constructions. Proposed interpretation method allows to use inference algorithm based on structural functional models for proving logical theorems expressed in the logics listed above. Experimental results of performance comparison with existing algorithms are shown.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 004.382.2

С. М. Абрамов

Исследования в области суперкомпьютерных технологий ИПС РАН: ретроспектива и перспективы

Аннотация. Данная работа посвящена обзору исследований в области суперкомпьютерных технологий, выполненых и запланированных Институтом программных систем имени А. К. Айламазяна Российской академии наук. Рассмотрены работы и их результаты за период, начиная с 1984 года.

1. Введение

В 1984 году в СССР шла подготовка эффективного ассиметричного ответа на американские планы «Звездных войн». С этой целью необходимо было выполнить ряд фундаментальных исследований в различных областях науки. Для этого решением директивных органов СССР в различных регионах страны было создано несколько исследовательских институтов, одним из которых был и наш институт[1]. При создании руководство страны в качестве основной деятельности определила нашему институту фундаментальные исследования в следующих областях:

- искусственный интеллект;
- высокопроизводительные вычисления — то, что сегодня принято называть суперкомпьютерными технологиями;
- операционные системы, языки и системы программирования, базы данных — то есть, системное программное обеспечение (ПО), информационные технологии.

[1] Создан в 1984 годы как филиал Института проблем кибернетики АН СССР. В 1986 году получил самостоятельный статус с наименованием: Институт программных систем АН СССР.

За прошедшие годы институт не только сохранил эту первоначальную направленность своих исследований, но и добился значительных результатов в этих направлениях. В данной статье рассмотрены выполненные в Институте программных систем Российской академии наук (ИПС РАН) исследования и разработки в области суперкомпьютерных технологий, с 1984 по настоящее время и перспективы их развития.

Эти работы охватывают фундаментальные исследования и инженерные разработки по созданию и применению аппаратных и программных средств мультипроцессорных вычислительных систем, суперЭВМ и grid-сетей. Данные работы в Институте в основном сосредоточены в Исследовательском центре мультипроцессорных систем (ИЦМС ИПС РАН), ряд прикладных работ выполнен в Исследовательском центре искусственного интеллекта. В статье также будут рассмотрены работы и результаты, полученные и в инициативных разработках института, и в рамках крупных (в том числе — международных) научно-технических программ, в которых ИПС РАН играл роль головного исполнителя от России. Некоторые из рассмотренных здесь результатов принадлежат институту, а некоторые — получены широкой научной кооперацией (см. Раздел «Благодарности»), возглавляемой институтом. Мы постараемся по мере изложения корректно освещать данное обстоятельство, хотя в полной мере это сделать будет невозможно из-за ограничений на размер статьи.

2. О сущности суперкомпьютерных технологий

Прежде чем начнем анализировать выполненные в ИПС РАН исследования в области суперкомпьютерных технологий, попытаемся дать короткое описание сущности суперкомпьютерных технологий и их роли.

Сегодня критические (прорывные) технологии в государствах, строящих экономику, основанную на знаниях, исследуются и разрабатываются на базе широкого использования высокопроизводительных вычислений. И другого пути — нет. Без серьезной суперкомпьютерной инфраструктуры:

- невозможно создать современные изделия высокой (аэрокосмическая техника, суда, энергетические блоки электростанций различных типов) и даже средней сложности (автомобили, конкурентоспособная бытовая техника и т. п.);

- невозможно быстрее конкурентов разрабатывать новые лекарства и материалы с заданными свойствами;
- невозможно развивать перспективные технологии (биотехнологии, нанотехнологии, решения для энергетики будущего и т. п.).

Сегодня суперкомпьютерные технологии считаются важнейшим фактором обеспечения конкурентоспособности экономики страны[2], а единственным способом победить конкурентов объявляют возможность обогнать их в расчетах. Здесь характерны слова Президента Совета по конкурентоспособности США:

«Технологии, таланты и деньги доступны многим странам. Поэтому США стоит перед лицом непредсказуемых экономических конкурентов из-за рубежа. Страна, которая желает победить в конкуренции, должна победить в вычислениях»[3].

Не важно, о конкуренции в каком секторе экономики идет речь: сказанное верно для добывающих и перерабатывающих секторов экономики, и особенно это верно при разработке новых технологий. Поэтому в развитых странах мира для перехода к экономике знаний создается новая инфраструктура государства — государственная система из мощных суперкомпьютерных центров, объединенных сверхбыстрыми каналами связи в грид-систему. То есть, по сути, речь идет о национальной научно-исследовательской информационно-вычислительной сети. Для такой системы часто используют термин «киберинфраструктура»[4]. В этих странах на создание национальной киберинфраструктуры выделяются большие финансы из государственных бюджетов: в 2005–2007 гг. США тратили на эти цели от 2 до 4 млрд. долларов в год.

---

[2] Например, см. доклад Президенту США «Вычислительные науки: обеспечение превосходства (конкурентоспособности) Америки»: “Computational Science: Ensuring America’s Competitiveness”, PITAC (President’s Information Technology Advisory Committee), 2005.

[3] “With technology, talent and capital now available globally, the U.S. is facing unprecedented economic competition from abroad. The country that wants to out compete must out-compute”. Deborah Wince-Smith, President of the Council on Competitiveness “US Competitive Council Meets; HPC TOPS Agenda” HPC Wire, 16.07.2004, http://www.taborcommunications.com/archives/108016.html.

[4] В данной работе оба термина — *«национальная научно-исследовательская информационно-вычислительная сеть»* и *«киберинфраструктура»* — используются как синонимы.

Тем самым, краткое определение сегодняшней роли суперкомпьютерных технологий может быть таким: это ключевая критическая технология, единственный инструмент, дающий возможность победить в конкурентной борьбе.

3. Этапы развития работ в ИПС РАН в области суперкомпьютерных технологий

В работах института по суперкомпьютерной тематике можно выделить несколько этапов:

- 1984–1992 гг.: участие ИПС РАН в разработке программного обеспечения (ПО) для мультипроцессора с динамической архитектурой (МДА) ЕС 2704[5];
- 1990–1995 гг.: работы с транспьютерными системами, участие ИПС РАН в Российской транспьютерной ассоциации; начало исследований и первых экспериментов в том направлении, которое в дальнейшем приведет к созданию Т-системы;
- 1994–1998 гг.: поиск и реализация решений для компонент первых версий Т-системы; в качестве аппаратной базы используются различные сети из ПЭВМ — начиная с оригинальных сетей на базе ускоренных (до 1 Mbps) линий RS-232 и собственных коммутирующих устройств для таких связей; заканчивая кластером на базе FastEthernet (100 Mbps);
- 1998–1999 гг.: развитие первой версии Т-системы, налаживание кооперации с коллегами из Минска, формирование суперкомпьютерной программы «СКИФ» Союзного государства;
- 2000–2004 гг.: период исполнения суперкомпьютерной программы «СКИФ» Союзного государства, в которой ИПС РАН определен как головной исполнитель от России; разработка суперЭВМ семейства «СКИФ» Ряда 1 (2000–2002 гг.) и Ряда 2 (2003–2006 гг.), разработка ПО для них;
- 2005–2007 гг.: инициативные работы в области суперЭВМ и grid-систем; развитие новой версии Т-системы (OpenTS), в том числе и в сотрудничестве с корпорацией Microsoft;

[5]Спецпроцессор ЕС 2704 Единого Семейства ЭВМ, оригинальная отечественная разработка ЛИИ АН СССР и НИЦЭВТ [25].

формирование и согласование суперкомпьютерной программы «СКИФ-ГРИД» Союзного государства; создание научно-технического задела для суперЭВМ «СКИФ» Ряда 3;

- 2007–2008 гг.: исполнение работ первого этапа суперкомпьютерной программы «СКИФ-ГРИД» Союзного государства, в которой ИПС РАН определен как головной исполнитель от Российской Федерации; создание Ряда 3 и разработка подходов к реализации Ряда 4 суперЭВМ семейства «СКИФ».

В последующих разделах мы остановимся на основных результатах, полученных в данных работах.

4. 1984–1992 годы: эпоха советских суперкомпьютерных систем

В истории отечественной суперкомпьютерной отрасли весьма существенным периодом являлись последние два десятилетия в истории СССР. В стране в то время одновременно велось несколько серьезных разработок вычислительных систем с высокой производительностью и параллельной архитектурой. То есть, одновременно велось несколько суперкомпьютерных проектов. Упомянем некоторые из них.

(1) Ереванский матричный спецпроцессор ЕС2700;
(2) Киевский макроконвейер ЕС2701;
(3) Ленинградский мультипроцессор с динамической архитектурой ЕС2704;
(4) Таганрогский мультипроцессор ЕС2706;
(5) Семейство мультипроцессорных систем ПС-1000 и ПС-2000, ИПУ АН СССР;
(6) Векторно-конвейерная СуперЭВМ «Электроника СС-БИС», ИПК АН СССР;
(7) Семейство СуперЭВМ Эльбрус-1 и Эльбрус-2, ИТМиВТ АН СССР;
(8) Суперкомпьютерные разработки НИИ «Квант»;
(9) Старшие модели — многомашинные и мультипроцессорные комплексы, ЕС1066 и старше, — НИЦЭВТ.

Десяток одновременных суперкомпьютерных проектов... И политическая воля была, и бюджета на собственные суперкомпьютерные проекты хватало.

Сразу после создания, в 1984–1992 годах, наш институт участвовал в разработке программного обеспечения для мультипроцессора с динамической архитектурой (МДА) ЕС 2704. ЕС2704 поддерживала микропрограммный уровень программирования. Нашему институту головные исполнители проекта ЕС2704 (НИЦЭВТ и ЛИИ АН СССР) поручили разработать ассемблер (как язык и систему программирования) и компилятор с языка С. Обе работы успешно выполнены и сданы заказчику.

Это была интересная машина, содержащая 6 интерфейсных, 12 коммутационных и 24 вычислительных процессорных модулей. К сожалению слишком долгий срок реализации проекта привел к тому, что только небольшое количество экземпляров ЕС2704 было установлено у пользователей. Однако в некоторых местах (ЦУП, Подлипки) эти машины использовались десяток лет.

В ЕС2704 были заложены красивые идеи, например, автоматическое динамическое распараллеливание пользовательских программ, устойчивость процесса выполнения программы к отказам части оборудования и т. п.

Опыт нашей работы с ЕС2704 лег в основу наших дальнейших разработок. В том числе, идея автоматического динамического распараллеливания программ была реализована нами, но уже совсем иным образом, в Т-системе.

5. 1993–1995 годы: транспьютерные системы

В 1993–1995 годы в институте выполнялись исследования с транспьютерными системами. Часть работ поддерживалась в рамках нашего участия в Российской транспьютерной ассоциации, часть исследований была поддержана грантом INTAS, полученным нами вместе с итальянским филиалом компании Inmos и с Университетом города Катанья.

На основе опыта, ранее полученного в работах по ЕС 2704 [1–5] нами велась разработка алгоритмов динамической маршрутизации (с обходом неисправностей) сообщений в транспьютерных сетях и балансировки загрузки процессоров.

Нами развивалась и теория расчета оптимальной конфигурации мультипроцессорной системы по заданному составу вычислительных

модулей. Речь идет о том, чтобы за счет выбора топологии транспьютерной сети минимизировать транзитные передачи в ней, максимизировать устойчивость к отказам и обеспечивать равномерность загрузки каналов сети. Оказывается, что при заданном числе каналов в узле транспьютерной сети[6] (при заданной «валентности» вычислительных узлов), оптимальной конфигурацией являются *графы с минимальным диаметром*. При этом, традиционные архитектуры (многомерные торы, гиперкубы, деревья и т. п.) сильно уступают графам с минимальным диаметром по устойчивости к отказам, поддержке равномерности загрузки каналов и минимизации транзитных передач. Для исследования графов с минимальными диаметрами в данные годы в ИЦМС ИПС РАН был реализован комплекс программных средств для расчета графов и их параметров.

В 1995 году в рамках работ по транспьютерной тематике в институте была выполнена разработка оригинальной интерфейсной платы для ПЭВМ на основе транспьютера T425. Разработанная плата [6] обеспечивала сопряжение ПЭВМ с вычислительной транспьютерной сетью. При этом интерфейс был выполнен на основе разделяемой памяти. Это решение обеспечило на шине ISA высокую скорость передачи данных — до 5 Мбайт/сек, что *на порядок превосходило параметры* всех существовавших в то время транспьютерных плат, использующих метод программного обмена через интерфейсную микросхему C011.

В этом же году нами был завершен перенос [7] свободного компилятора GNU C Compiler на архитектуру транспьютеров семейств T4, T8 и T9. Заключительная отладка и тестирование компилятора была осуществлена в удаленном режиме на установке GCel фирмы Parsytec в High Performance Computing Laboratory в Афинах (Греция), что для того времени[7] было очень серьезным результатом. Тестирование показало, что по качеству генерируемого кода наш компилятор не уступает коммерческому компилятору ACE, входящему в состав ОС Parix. Этим результатам более чем десятилетней давности до сих пор посвящен раздел в архиве “Internet Parallel Computing Archive” [8].

---

[6] В каждом транспьютере, как правило, поддерживалось 4 канала, однако, существовали транспьютероподобные системы с иным количеством каналов в узле.

[7] 1995 год, в России развиты только UUCP-сети, Интернет еще не вошел широко в нашу жизнь.

В это же время (1994–1995 годы) в институте велись исследования по возможности высокоскоростной связи между ПЭВМ при помощи RS-232. Было разработано и реализовано коммутационное оборудование для связи между собой ПЭВМ со скоростью передачи данных 1 Mbps. В дальнейшем такие предвестники кластерных систем — сети из ПЭВМ, связанные ускоренными линиями RS-232, — нами использовались как платформы для разработок Т-Системы — системы программирования, обеспечивающей автоматическое распараллеливание программ на этапе выполнения программ в мультипроцессорных вычислительных системах с общей и распределенной памятью [9, 10].

Базовые принципы Т-системы были впервые четко сформулированы в 1995 году. В качестве входного языка системы рассматривались диалекты известных языков программирования (С, Фортран), с небольшим количеством дополненных специальных конструкций и функционально-ориентированных ограничений [11]. Возможность автоматического распараллеливания основывается на представлении вычислений в виде автотрансформации вычислительной сети, состоящей из процессов и обрабатываемых данных. Выполнена первая экспериментальная реализация Т-системы.

В 1995 году было выполнено исследование применимости методов автоматического распараллеливания к основным алгоритмам вычислительной математики, показано, что представление характерных для вычислительной математики структур данных возможно на основе специальных списковых структур, используемых при реализациях Т-системы, что не приводит к существенной потере производительности [12].

Первые прототипные программные реализации для экспериментов с Т-системой создавались в среде MS DOS. И именно в 1995 г. начались работы по использованию ОС Linux и локальных сетей UNIX-станций в качестве платформы для Т-системы; была разработана сетевая компонента Т-системы, обеспечивающая возможность распределенной загрузки задачи и внутризадачного обмена управляющими и информационными сообщениями.

6. 1996–1997 годы: первые реализации Т-системы

В 1996–1997 годы в рамках создания первой рабочей версии Т-системы в институте были решены все основные вопросы по реализации ядра системы. Для управления в Т-системе памятью был разработан

алгоритм таблично-страничной компактирующей сборки мусора, являющийся модификацией алгоритма «катящихся таблиц». Новый алгоритм сохранял преимущества метода «катящихся таблиц» — отсутствие необходимости отводить дополнительный указатель для каждого элемента данных, хранимых в звеньевой памяти; но избавлялся от основного недостатка — потенциально невысокой эффективности.

В это же время для поддержки отладки Т-программ пришлось решать проблему отсутствия повторяемости (от запуска к запуску) трассы вычислений, обусловленной асинхронным характером взаимодействия компонент Т-системы. Был разработан метод отладки, основанный на принципе «повторения трассы выполнения».

Версия ядра Т-системы, разработанная с использованием найденных алгоритмов и методов, была опробована и отлажена в процессе реализации первой задачи для Т-системы: построения реалистических изображений виртуальных сцен методом трассировки лучей. Были проведены первые эксперименты по выполнению данной задачи в однопроцессорном режиме, а также на локальной сети (Ethernet 10Base-2) из четырех одинаковых однопроцессорных ПЭВМ, работающих под управлением ОС Linux. Результаты счета на одном, двух, трех и четырех процессорах показали, что Т-система обеспечивает для данной задачи:

- низкий (1.5–3%) уровень накладных расходов (по сравнению с традиционной непараллельной реализацией);
- практически линейный рост производительности в зависимости от числа процессоров [12–16].

Эта первая рабочая реализация Т-системы, первый простейший кластер (четыре ПЭВМ в сети Ethernet 10Base-2) и первая Т-программа стали началом современного этапа суперкомпьютерных исследований в нашем институте.

7. 1998 год: завершение разработки первой стабильной прототипной версии Т-системы

В начале 1998 года в институте был реализован новый крупный (для того времени) кластер — программно-аппаратный мультипроцессорный комплекс, в котором использовались различные по конфигурации вычислительные узлы: всего 24 процессора Intel PPro-200 и P-

II-266, пиковая производительность 5,6 GFlops[8], RAM 1.4 GB, HDD 70.4 GB. Заметим, что вычислительные узлы данной установки были первыми компьютерами в ИПС РАН с архитектурой SMP. Данная вычислительная техника позволила разработать и поддержать в Т-системе различные платформы класса «IP-сеть из Intel-совместимых компьютеров с ОС Linux, в том числе с SMP-архитектурой».

Весной того же года на языке Рефал Плюс был реализован первый компилятор для Т-языка — это было негладкое синтаксическое расширение языка С. Так была завершена разработка первой стабильной прототипной версии Т-системы.

За первые полгода опытной эксплуатации в ИПС РАН кластера с Т-системой были реализованы 10 задач из различных прикладных областей, имеющих различную алгоритмическую природу [13,14,16]. На данных задачах были исследованы свойства Т-системы и практически продемонстрированы важнейшие свойства данного подхода к организации параллельных вычислений:

- Т-система автоматически распараллеливает выполнение Т-программ, при этом для многих алгоритмов достигается почти линейный рост производительности при росте числа процессоров (от 1 до 24);
- разработанные на Т-языке задачи могут выполняться (без переписывания, перекомпиляции или иных других модификаций) на мультипроцессоре с произвольной аппаратной конфигурацией.

Тем самым Т-система позволяет снизить затраты на разработку параллельных программ (автоматизация распараллеливания), увеличить глубину параллелизма и более полно использовать возможности аппаратной части мультипроцессора (за счет распараллеливания в динамике).

[8]Единицы производительности суперЭВМ: 1 Gflops — миллиард операций с плавающей точкой в секунду, 1 TFlops = 1 000 GFlops — триллион операций с плавающей точкой в секунду, 1 Pflops = 1 000 TFlops — тысяча триллионов операций с плавающей точкой в секунду.

8. 1998–1999 годы: формирование суперкомпьютерной программы «СКИФ» Союзного государства

В мае 1998 года состоялись первый визит в Минск в НПО «Кибернетика» НАН Беларуси и первые контакты с белорусскими коллегами по тематике высокопроизводительных вычислений. В российской делегации были представители московской компании «Суперкомпьютерные системы» (СКС) и сотрудники ИПС РАН. Компания СКС демонстрировала макетный образец однородной вычислительной среды (ОВС), изготовленный на базе микросхем, выпущенных на предприятии «Интеграл» (Минск). Представители ИПС РАН демонстрировали Т-систему и прикладные Т-программы. Обе демонстрации получили положительную оценку у Президента НАН Беларуси Войтовича А. П.

Для более подробного изучения возможной кооперации был намечен ответный визит, который состоялся 5–12 июня 1998 года. В конце визита 12 июня 1998 года Президент НАН Беларуси Войтович А. П., директор ИПС РАН Айламазян А. К. и директор компании СКС Татур В. Ю. подписали трехстороннее соглашение о кооперации в области высокопроизводительных вычислений. Это был первый шаг к формированию суперкомпьютерной программы «СКИФ» Союзного государства.

После возвращения в Минск, Войтович А. П. доложил Президенту Республики Беларусь Лукашенко А. Г. о результатах первых контактов и о возможном сотрудничестве между Россией и Беларусью в суперкомпьютерной отрасли. Эти идеи нашли горячую поддержку на высшем государственном уровне Беларуси. Было выделено целевое финансирование, создан временный научный коллектив, целью которого было исследование всех аспектов возможного сотрудничества и формирование (подготовка) совместной программы «СКИФ» Союзного государства.

К концу 1998 года временный научный коллектив завершил свою работу, текст совместной программы «СКИФ» был сформирован. После этого весь 1999 год ушел на согласование программы в российских министерствах и ведомствах. И только с осени 2000 года началась реальная работа по выполнению Программы «СКИФ».

Конечно, хождение по министерствам мы в 1999 году совмещали и с продолжением исследований. В этом же году для поддержки эффективной работы Т-системы на SMP-узлах была разработана и

реализована система управления разделяемой памятью, обладающая следующими особенностями:

- процессы операционной системы, реализующие единое приложение, разделяют виртуальные адресные пространства не полностью, а только частично — в отличие от стандартных POSIX-совместимых реализаций систем поддержки легковесных процессов;
- когерентное размещение сегментов разделяемой памяти в виртуальное адресное пространство процессов может осуществляться динамически.

Данное свойство позволяет экономно использовать ресурс сегментов разделяемой памяти ОС, общее количество которых ограничено.

В этом же году продолжались развитие Т-языка, программирование в Т-системе демонстрационных и реальных прикладных задач, опытная эксплуатация кластера ИПС РАН. В таком инициативном порядке работы по Т-системе развивались до конца лета 2000 года — до начала финансирования суперкомпьютерной программы «СКИФ» Союзного государства.

9. 2000–2004 годы, период исполнения суперкомпьютерной программы «СКИФ» Союзного государства

Первые пять лет нового века работы ИЦМС ИПС РАН в области суперкомпьютерных технологий проходили в рамках исполнения суперкомпьютерной программы «СКИФ» Союзного государства. Полное название программы «СКИФ» — «Разработка и освоение в серийном производстве семейства моделей высокопроизводительных вычислительных систем с параллельной архитектурой (суперкомпьютеров) и создание прикладных программно-аппаратных комплексов на их основе», — точно определяло и содержание работ. Это была серьезная совместная научно-техническая программа двух стран — России и Беларуси.

Институт программных систем Российской академии наук был определен головным исполнителем по программе «СКИФ» от России. Работы по созданию аппаратных и программных средств для семейства суперкомпьютеров «СКИФ» ИПС РАН велись в тесном сотрудничестве с исполнителями от Республики Беларусь и с основными исполнителями Программы со стороны России, среди которых были:

- ОАО «Научно-исследовательский центр электронно-вычислительной техники» (НИЦЭВТ, Москва);
- Центр научных телекоммуникаций и информационных технологий (ЦНТК РАН, Москва);
- НИИ механики МГУ имени М. В. Ломоносова (Москва);
- Институт высокопроизводительных вычислений и информационных систем (ИВВиИС, СПб.);
- Российский НИИ региональных проблем (РосНИИ РП, Переславль-Залесский);
- Компания «Суперкомпьютерные системы» (СКС, Москва).

Мероприятия программы «СКИФ» охватывали все области суперкомпьютерной отрасли:

- разработка и производство микроэлементной базы, супер-ЭВМ, системного ПО для них, инструментального ПО и прикладных систем;
- вспомогательные мероприятия — подготовка и переподготовка кадров, создание единого информационного пространства проекта.

Суперкомпьютерной программе «СКИФ» посвящено достаточно много публикаций. Поэтому в данной работе будет дан весьма краткий обзор основных результатов и приведены ссылки на соответствующие публикации.

9.1. Основные результаты суперкомпьютерной программы «СКИФ» Союзного государства

В рамках суперкомпьютерной программы «СКИФ» Союзного государства в 2000–2003 годах были получены следующие результаты [17–20]:

- Разработана конструкторская документация (КД) и образцы высокопроизводительных систем «СКИФ» Ряда 1, которые прошли приемочные (государственные) испытания. По результатам государственных испытаний конструкторской документации присвоена литера $О_1$.
- Разработано базовое программное обеспечение кластерного уровня (ПО КУ) и ряд прикладных систем суперкомпьютеров «СКИФ» Ряда 1 и Ряда 2. Данное ПО прошло приемочные (государственные) испытания и по результатам испытаний данному ПО присвоена литера $О_1$.

Среди прочего, на испытания выносилось более двадцати программных систем, среди них:

- модифицированное ядро операционной системы Linux-SKIF (ИПС РАН и МГУ);
- модифицированные пакеты параллельной файловой системы PVFS-SKIF и системы пакетной обработки задач OpenPBS-SKIF (ИПС РАН и МГУ);
- мониторная система FLAME-SKIF кластерных установок семейства «СКИФ» (ИПС РАН и МГУ);
- стандартные средства (MPI, PVM) поддержки параллельных вычислений, 12 адаптированных пакетов, библиотек и приложений (ИПС РАН и МГУ);
- T-система и сопутствующие пакеты: T-ядро, компилятор tgcc, пакет tcmode для редактора Xemacs, демонстрационные и тестовые T-задачи (ИПС РАН и МГУ);
- отладчик TDB для MPI-программ (ИПС РАН);
- две прикладные системы, разрабатываемые по программе «СКИФ»: система автоматизации проектирования химических реакторов (ИВВиИС, СПб.), система классификации большого потока текстов при помощи технологий искусственного интеллекта (ИЦИИ ИПС РАН).

- В ОАО «НИЦЭВТ» подготовлена производственная база, проведена разработка КД и освоены в производстве адаптеры (N330, N335, N337,) системной сети SCI, которые являются полными функциональными аналогами адаптеров SCI компании Dolphin (D330, D335, D337).
- В 2000–2003 гг. построено шестнадцать опытных образцов и вычислительных установок Ряда 1 и Ряда 2 семейства «СКИФ»[9].
- Начаты работы по инженерным расчетам на системах семейства «СКИФ» и по созданию единого информационного пространства программы «СКИФ». В рамках приемочных (государственных) испытаний сверх программы и методики испытаний были показаны первые результаты в этом направлении:

[9] В том числе предприятием «Суперкомпьютерные системы» (Москва) совместно с НИИ ЭВМ (Минск) изготовлен экспериментальный гибридный макет, в котором кластерный уровень сочетается с вычислительными модулями ОВС.

– Была создана метакластерная распределенная вычислительная структура на базе сети Интернет и трех кластерных систем «СКИФ» ИПС РАН (Переславль-Залесский), НИИ механики МГУ (Москва) и ОИПИ НАН Беларуси (Минск). Подтверждена функциональность и перспективность использования T-системы в качестве базы для создания высокоуровневой среды поддержки подобных конфигураций.
– Проверен режим удаленного доступа из Минска к ресурсам пакета STAR-CD[10], установленного на суперкомпьютере «СКИФ» в ИПС РАН (Переславль-Залесский).
– Проверен режим удаленного доступа из Минска с помощью Web-интерфейса к ресурсам установленного на суперкомпьютере «СКИФ» в ИПС РАН (Переславль-Залесский) программного комплекса для расчета процессов в PECVD-реакторах.
– Показаны результаты использования ведущего в области механики деформируемого твердого тела инженерного пакета LS-DYNA, установленного на суперкомпьютере «СКИФ» в г. Минске (УП «НИИ ЭВМ»).

В Программе «СКИФ» было предусмотрено и мероприятие, связанное с подготовкой и переподготовкой кадров для работы с высокопроизводительными установками семейства «СКИФ». В рамках данного мероприятия в летний период, каждый год, начиная с 2002 года и до сих пор в Переславле-Залесском проводятся студенческие школы-семинары по Программе «СКИФ», с участием студентов и аспирантов из России, Белоруссии, Украины. На различных секциях школы студенты занимаются инженерными расчетами и программированием на T-системе, освоением технологий, использованных в семействе суперкомпьютеров «СКИФ». В последние дни работы школ проводятся конференции, где каждый участник докладывает о результатах, полученных в рамках школы.

В подразделах, перечисленных ниже, мы остановимся несколько подробнее на некоторых результатах программы «СКИФ»:

- GRACE и OpenTS: развитие реализаций T-системы;

[10] STAR-CD — один из ведущих инженерных пакетов в области механики жидкости и газа.

- собственное программное обеспечение для семейства суперкомпьютеров «СКИФ»;
- разработка сервисной сети ServNet (версии 1 и 2) для супер-ЭВМ семейства «СКИФ»;
- вхождение в мировой рейтинг Top500 суперкомпьютеров семейства «СКИФ» Ряда 2;
- общая оценка результатов суперкомпьютерной программы «СКИФ» Союзного государства.

9.2. GRACE и OpenTS: развитие реализаций T-системы

В рамках выполнения Программы «СКИФ» продолжались исследования, связанные с T-системой. Были разработаны две новые версии T-системы (в сотрудничестве с МГУ имени М. В. Ломоносова):

- GRACE (1999–2002 гг.) — версия [21–23], поддерживающая в качестве базового T-языка синтаксически-гладкое расширение языка C. Система GRACE в 2002 году прошла приемочные (государственные) испытания с присвоением литеры O_1.
- OpenTS (2003–2004 гг.) — T-система с открытой архитектурой, реализованная в виде надстройки (суперструктуры) над языком C++. Завершение разработки первой версии и прохождение приемочных испытаний OpenTS было выполнено в 2004 году, по результатам испытаний системе OpenTS была присвоена литера O_1.

В новой версии T-системы с открытой архитектурой (OpenTS) были поддержаны новые возможности: эффективная поддержка архитектуры SMP, асинхронный режим обменов внутри T-системы, механизм обеспечения отказоустойчивости, новая система сбора статистики, механизм трассировки T-программ. Была проведена модификация языковых средств T-системы (компилятора TG++, конвертора с языка T++ и транслятора с языка T-Fortran) в соответствии с новыми возможностями T-системы.

9.3. Собственное программное обеспечение для семейства суперкомпьютеров «СКИФ»

Огромные усилия, большая доля времени, сил и средств были потрачены в Программе «СКИФ» на разработку программного обеспечения (ПО) и литерной программной документации (ПД). Отметим,

что вся ПД была разработана в соответствии с требованиями ЕСПД, проведена через нормоконтроль, успешно прошла приемочные (государственные) испытания с присвоением литеры «$О_1$». Общий объем комплекта ПО для семейства суперкомпьютеров «СКИФ» составляет 10 дисков CD-ROM. Перечислим основные компоненты данного комплекта:

- стандартное ядро ОС Linux и модифицированное ядро ОС Linux-SKIF (с большим уровнем информационной безопасности);
- параллельная файловая система PVFS-SKIF, модифицирована под специфику семейства «СКИФ»;
- система очередей OpenPBS-SKIF (модифицирована);
- оригинальная система мониторинга и управления установками семейства «СКИФ» Flame-SKIF, среди прочего включающая и поддержку сервисной сети СКИФ-ServNet;
- TDB — распределенный интерактивный отладчик MPI-программ, с поддержкой отладки T-программ (отечественная замена дорогостоящей системы TotalView);
- 12 адаптированных к особенностям семейства «СКИФ» свободных пакетов, библиотек и параллельных приложений;
- 7 прикладных программных систем, разработанных в среде OpenTS:
  - MultiGen (ЧелГУ) — система расчета биологической активности молекул с учетом их конформационного многообразия, прогнозирование и проектирование в химии (лекарства и другие соединения);
  - пакет расчета аэромеханики подвижных плохообтекаемых тел (НИИ механики МГУ имени М. В. Ломоносова);
  - обработка и поиск XML-данных (НИИ механики МГУ имени М. В. Ломоносова);
  - программная система формирования фокусированных радиолокационных изображений (НИИ КС);
  - программная система моделирования широкополосных пространственно-временных радиолокационных сигналов (НИИ КС);
  - программная система поточечной обработки цветных и полутоновых видеоданных космических систем дистанционного зондирования (НИИ КС);

  - программная система классификации гиперспектральных изображений со спутника LANDSAT (ИПС РАН);
- 14 параллельных приложений собственной разработки, среди которых:
  - ИПС РАН, Росгидромет: модель проф. В. М. Лосева и другие метеорологические модели;
  - ОИПИ НАН Беларуси, Республиканский Гидрометеорологический центр: численные методы прогнозирования погоды, модели регионального прогноза погоды на 48 часов;
  - ИЦИИ ИПС РАН: три прикладные системы искусственного интеллекта: АКТИС — классификация текстов по заданным в процессе обучения классам (глубокий анализ текста, высокая релевантность); INEX — извлечение знаний из неструктурированных текстов на естественном языке, заполнение заданной реляционной БД; MIRACLE (система разработана на OpenTS) — инструментальная система для проектирования интеллектуальных систем;
  - ОИПИ НАН Беларуси, РНПЦ «Кардиология» и УП «НИИЭВМ»: кардиологический комплекс на базе кластера «СКИФ»;
  - ИВВиИС: кардиологическая экспертная система реального времени «ADEPT-C».

9.4. Сервисная сеть ServNet (версии 1 и 2) для суперЭВМ семейства «СКИФ»

Разработка сервисной сети [24] (ServNet версии 1 и 2) для суперЭВМ интересна тем, что удалось небольшим и простым аппаратным изделием поддержать достаточно богатый набор возможностей. С помощью управляющей сети ServNet можно было:

- селективно включать и отключать питание на любом узле суперЭВМ;
- селективно выполнить аппаратный сброс любого вычислительного узла;
- селективно осуществлять подключение к порту RS232 узла, а это, в свою очередь, обеспечивает взаимодействие с узлом кластера в консольном режиме Linux serial console, а также возможность удаленной работы в режиме BIOS Setup, если

BIOS узла поддерживает взаимодействие через последовательный интерфейс.

Возможность удаленного доступа с управляющей станции на сериальную консоль узла позволяет реализовывать различные функции управления узлом:

Упомянутые выше возможность работы с *Linux serial console* вычислительного узла и удаленной работы в режиме BIOS Setup.

- Возможность управления загрузчиком (LILO) операционных систем вычислительного узла. На сериальную консоль может быть сконфигурировано управление LILO и, если на узлах установлено несколько различных операционных систем, то можно с управляющей станции выбрать тип ОС, загружаемой на каждом узле и таким образом на всем кластере (или на части узлов) может быть загружена та или иная ОС (из предустановленных на узле).
- Возможность изменять параметры загрузки ядра Linux на каждом узле.
- Возможность «посмертного» просмотра некоторого количества последних строк (до 4 Кбайтов), выведенных на сериальную консоль. При «зависании» вычислительной системы, когда уже не работают ни системная, ни вспомогательная сети кластера, в независимой памяти адаптера ServNet сохраняется информация о состоянии системы перед «зависанием». Таким образом, с управляющей станции можно восстановить картину последних «мгновений жизни» вычислительной системы и понять причину сбоя. Данная возможность поддержана только в ServNet, ее нет в аналогичных западных разработках.

9.5. Вхождение в мировой рейтинг Top500 суперкомпьютеров семейства «СКИФ» Ряда 2

Из шестнадцати опытных образцов и вычислительных установок семейства «СКИФ», созданных в 2000–2003 гг., две суперкомпьютерные установки вошли в мировой рейтинг пятисот самых мощных ЭВМ Top500[11].

[11] http://www.top500.org/

Рис. 1. СуперЭВМ «СКИФ К-1000»

- В ноябре 2003 года суперЭВМ «СКИФ К-500» заняла 405 место в рейтинге Top500, показав пиковую производительность 716.8 Gflops, реальную производительность на задаче Linpack — 471.6 Gflops (65.79% от пиковой).
- В ноябре 2004 года суперЭВМ «СКИФ К-1000» (Рис. 1) заняла 98 место в рейтинге Top500, показав пиковую производительность 2534 Gflops, реальную производительность на задаче Linpack — 2032 Gflops (80.19% от пиковой).

Добавим, что суперкомпьютер «СКИФ К-1000» на момент создания (октябрь 2004 года) являлся самой мощной из всех вычислительных систем, установленных на территориях России, СНГ и Восточной Европы. В рейтинге Top Cranch (поддержан агентством перспективных оборонных исследований — DARPA) суперкомпьютер «СКИФ К-1000» занял тогда первое место в мире на задаче расчета столкновения трех автомобилей.

В ноябре 2004 г. суперкомпьютеры «СКИФ К-1000» и «СКИФ К-500», конструкторская и программная документация суперкомпьютеров Ряда 2 семейства «СКИФ» успешно прошли государственные испытания с присвоением литеры «$О_1$». Государственная комиссия отметила готовность промышленного выпуска суперкомпьютеров семейства «СКИФ» Ряда 2 с производительностью до 15 триллионов операций в секунду (15 Tflops), соответствующих международным стандартам по техническим показателям, составу, комплектности и программному обеспечению. Очень важным результатом программы «СКИФ» являлась подготовленная производственная база, возможности участников программы «СКИФ» позволяющие серийно выпускать:

- суперкомпьютеры с производительностью до 15 Тфлопс — по технологиям, ранее проверенным на суперЭВМ Ряда 2 семействе «СКИФ»;
- отечественные адаптеры высокоскоростных сетей SCI для кластеров — полных аналогов адаптеров фирмы Dolphin (SCI PCI64/66 Dolphin ICS, 1D- и 2D-тор, D330, D337, D335);
- адаптеры сервисной сети СКИФ-ServNet.

9.6. Общая оценка результатов суперкомпьютерной программы «СКИФ» Союзного государства

Программа «СКИФ» была признана одной из самых успешных программ Союзного государства. Так, в Постановлении Совета Министров Союзного государства от 21 апреля 2005 г. № 17 «Об итогах выполнения программы Союзного государства «Разработка и освоение в серийном производстве семейства высокопроизводительных вычислительных систем с параллельной архитектурой (суперкомпьютеров) и создание прикладных программно-аппаратных комплексов на их основе» вместо дежурного «принять представленный отчет» было сказано:

Совет Министров Союзного государства постановляет:

1. Считать завершенной программу Союзного государства «Разработка и освоение в серийном производстве семейства высокопроизводительных вычислительных систем с параллельной архитектурой (суперкомпьютеров) и создание прикладных программно-аппаратных комплексов на их основе» и одобрить представленный Национальной академией наук Беларуси и Федеральным агентством по

науке и инновациям (Министерство образования и науки Российской Федерации) отчет об итогах ее реализации в 2000–2004 годах (прилагается).

2. Федеральному агентству по науке и инновациям и Национальной академии наук Беларуси подготовить и внести установленным порядком в Совет Министров Союзного государства предложение о дальнейшем развитии работ в области создания и разработки высокопроизводительных вычислительных систем в рамках Союзного государства.

3. Настоящее постановление вступает в силу со дня его подписания.

Немного позже эти результаты получили и высокую правительственную оценку. За работу «Разработка конструкторской и программной документации, подготовка промышленного производства и выпуск образцов высокопроизводительных вычислительных систем (суперкомпьютеров) семейства «СКИФ» Ряда I и Ряда II» была присуждена премия Правительства Российской Федерации в области науки и техники за 2006 год группе исполнителей Программы «СКИФ». Лауреатами стали:

- От ИПС РАН, Переславль-Залесский: Айламазян Альфред Карлович (посмертно), Абрамов Сергей Михайлович, Адамович Алексей Игоревич, Коваленко Максим Русланович, Пономарев Александр Юрьевич, Шевчук Юрий Владимирович;
- От НИЦЭВТ, Москва: Слуцкин Анатолий Ильич;
- От компании «Т-Платформы», Москва: Опанасенко Всеволод Юрьевич;
- От ОИПИ НАН Беларуси, Минск: Анищенко Владимир Викторович, Парамонов Николай Николаевич.

Пожалуй, самым важным результатом программы «СКИФ» можно назвать восстановление и создание кооперационных связей, организацию такой команды исполнителей, которой по плечу самые сложные задачи в области суперкомпьютерных технологий.

Это особенно ярко проиллюстрировал опыт создания «СКИФ К-1000». Как отмечалось, в ноябре 2004 г. суперкомпьютер «СКИФ К-1000» занял 98-е место в рейтинге Top500. При этом суперкомпьютеры из «первой сотни», кроме участников программы «СКИФ», выпускали в это время только США, Япония и Китай. Многие другие

страны Западной Европы и Азии обладали развитой суперкомпьютерной отраслью и создавали суперкомпьютеры, входящие в Top500, но не в «первую сотню»!

Этот факт доказывал тезис, что на этот момент команда исполнителей программы «СКИФ» действительно достигла мирового уровня в освоении суперкомпьютерных технологий. И не использовать такой ресурс в интересах России и Беларуси было бы ошибкой.

10. 2005–2007 годы: бюрократические проволочки и потеря темпа

Еще в середине 2004 г. в сотрудничестве ИПС РАН с белорусскими и российскими организациями были подготовлены предложения по формированию новой суперкомпьютерной программы Союзного государства: «Разработка и использование программно-аппаратных средств ГРИД-технологий и перспективных высокопроизводительных (суперкомпьютерных) вычислительных систем семейства «СКИФ» (шифр «СКИФ-ГРИД»)».

К сожалению, предложения по формированию новой суперкомпьютерной программы Союзного государства «СКИФ-ГРИД» рассматривались, согласовывались, утверждались в российских ведомствах[12] долгих три с лишним года: с 2004 по март 2007 года. А без государственной поддержки продолжения научного направления программы «СКИФ» научный задел и потенциал команды исполнителей программы «СКИФ» безнадежно терялся.

Общеизвестно, что мировые суперкомпьютерные технологии развиваются по закону Мура — приблизительно «удвоение за год», — по всем основным показателям. Это обстоятельство означает, что 1 год простоя приводит к потере 1/2 части задела и потенциала; 2 года простоя — от задела и потенциала остается 1/4 часть и т. д. Это позволяет оценить тот ущерб, который был нанесен бюрократическими проволочками научному заделу и потенциалу, ранее огромным трудом созданному исполнителями программы «СКИФ».

В силу указанных обстоятельств, в 2005–2007 годах исследования в области суперкомпьютерных и grid-технологий институтом выполнялись в инициативном порядке и в рамках фундаментальных исследований Российской академии наук.

[12] В Беларуси согласования были завершены весной 2005 года.

В 2005 году была реализована распределенная вычислительная сеть T-Grid в г. Переславле-Залесском, объединяющая в единую сеть часть вычислительных ресурсов ИПС РАН и Университета города Переславля. Вычисления в сети T-Grid производились с использованием среды динамического автоматического распараллеливания программ — T-Системы. Проведены эксперименты по запуску T-приложений на гетерогенных кластерах и системе T-Grid, а также по использованию стенда T-Grid в метакластерной установке (120 процессоров).

В это же время была начата разработка основных принципов и аппаратных компонент сервисной сети (ServNet версии 3) для суперЭВМ семейства «СКИФ» Ряда 3 с большим числом узлов — десятки тысяч и более. Отличия данной версии ServNet от предыдущих были в следующем:

- ServNet версии 3 был ориентирован на суперЭВМ, созданные на базе blade-решений — вычислитель суперЭВМ собирался из шасси, внутри шасси устанавливались вычислительные модули;
- соответственно, в сети ServNet версии 3 использовались две печатные платы: ServNet CNB — плата управления шасси и подключенные к ней платы ServNet T-60 — по одной плате на каждый вычислительный узел;
- для связи в единую сеть платы ServNet T-60 снабжались интерфейсами RS 485, а плата ServNet CNB — интерфейсами RS 485 и Ethernet 10Base-T;
- в рамках монтажного шкафа все платы ServNet CNB объединяются сетью RS 485. И все платы ServNet T-60 каждого шасси подключаются при помощи RS 485 к соответствующей плате ServNet CNB;
- в каждом шкафу одна из плат ServNet CNB (или несколько для надежности) подключаются к Ethernet-сегменту, охватывающему все шкафы, и управляющие станции суперЭВМ.

Таким образом, была обеспечена практически безграничная масштабируемость сервисной сети для суперЭВМ любого диапазона производительности.

ServNet версии 3 была востребована при построении суперкомпьютеров «СКИФ МГУ» и «СКИФ Урал» в 2008 году. К этому времени разработка была полностью завершена, и был развернут выпуск аппаратуры ServNet версии 3 на базе опытного производства ИПС

РАН. Всего в короткий срок нами было выпущено необходимое (для суперЭВМ «СКИФ МГУ» и «СКИФ Урал») количество изделий: более 800 плат ServNet T-60 и 80 плат ServNet CNB.

Выполнялись в эти же годы и работы по развитию OpenTS — современной реализации T-системы. В 2004 году на Open TS обратила внимание корпорация Microsoft. В результате нескольких рабочих обсуждений был оформлен контракт на сравнительный анализ эффективности использования системы OpenTS и библиотеки MPI для написания прикладных программ. Задача исследования была поставлена следующим образом. Корпорация Microsoft выбрала два приложения[13], которые разрабатывались многие годы на базе библиотеки MPI усилиями сотен программистов — сторонников движения Open Source. Таким образом, выбранные приложения были максимально оптимизированы (отшлифованы) — насколько это позволяет MPI-подход. Сотрудники ИПС РАН в сжатые сроки, не более 3-х месяцев, должны были переписать две отобранные системы с MPI на OpenTS. Результат считался бы положительным, если в T-системе код получился бы более ясным и компактным. При этом допускалась потеря производительности до 30%.

Тем самым корпорация Microsoft явно выражала свои предпочтения: допуская даже небольшую потерю производительности программ важно повысить производительность труда программистов, обеспечить более ясный и компактный код (а, значит, и более надежный код: меньше строчек — меньше места для ошибок, большая ясность — меньше причин для ошибок).

Результаты данной работы были неожиданными даже для нас:

- огромные программные комплексы были в двухмесячный срок 2–3 программистами переписаны на OpenTS, отлажены и запущены;
- было достигнуто радикальное улучшение компактности кода — T-программы были значительно меньше, чем MPI-программы: для Powray — в 7–15 раз, а для ALCMD — в 7 раз;
- при этом T-программы и MPI-программы показывали примерно одинаковую производительность.

[13] Powray — пакет построения реалистичных трехмерных сцен методом трассировки лучей; ALCMD — программа для расчетов в области молекулярной динамики.

Данный сравнительный анализ выполнялся в операционной системе Linux — для которой и была изначально разработана система OpenTS. Результаты анализа показались корпорации Microsoft интересными, и в 2006 году наше сотрудничество было продолжено в рамках нового контракта, целью которого был перенос OpenTS из операционной системы Linux в операционную систему Windows Compute Cluster Server. Контракт был успешно выполнен в 2006 году: ядро T-системы было перенесено в WinCCS, транслятор с языка T++ был интегрирован в Visual Studio, был разработан интерактивный инсталлятор, справочная подсистема и все остальные компоненты, традиционные для систем программирования под Windows. Результаты представлялись на международной конференции и выставке «Supercomputing'2006'», Тампа, США, ноябрь, 2006. В последующие годы (2007–2008) новые версии OpenTS нами создавались в кросс-платформенном варианте: один и тот же код OpenTS можно было собрать в дистрибутив для операционной системы Linux и для Windows Compute Cluster Server.

В 2005–2007 годах в ИПС РАН была разработана система сбора данных о показаниях сенсоров узлов вычислительных кластеров, в которой осуществляется постоянный сбор данных и накопление информации об отказах оборудования. Система предназначена не только для сбора информации, но и для упреждающего предсказания на базе собранных данных вероятного отказа оборудования, а также нетипичного поведения аппаратных и программных средств супер-ЭВМ. Система установлена на разных суперкомпьютерах семейства «СКИФ»: Переславль, Москва, Минск, Челябинск, Томск и др.

В 2006–2007 году в дополнение к T-системе была создана и развивалась библиотека T-Sim параллельного программирования на основе шаблонных классов C++. В ее состав включен набор шаблонов — функций высшего порядка, применяющих операции, переданные пользователем (программистом), к множеству входных данных — модель map-reduce. Поведение шаблонов определяется как классами-стратегиями, переданными программистом, так и типами элементов, составляющих входное и результирующее множества. При создании параллельных программ на основе T-Sim разработчик может использовать как высокоуровневые функции-шаблоны (в наиболее простых случаях), так и реализуя нестандартные решения с использованием механизмов T-системы или механизмов активных сообщений. Это

позволяет пользователю выбирать инструменты, наиболее адекватные задаче.

В 2006 году нами был предложен новый подход к созданию вычислительных Web-сервисов на основе системы автоматического динамического распараллеливания программ, основанный на автоматической генерации вычислительных grid-сервисов по описанию T-программ. Были разработаны программные средства генерации grid-сервисов, поддерживающие метакластерные и гетерогенные конфигурации за счет механизма сериализации объектов в XML-форму. Была реализована возможность использования протокола MPI для обмена данными и другие дополнительные возможности. Разработанные программные средства предназначены для организации инструментальной среды создания grid-приложений с использованием системы автоматического динамического распараллеливания программ T-Sim.

Нами были выполнены исследования среды на основе интеграции разнородных источников данных Open Grid Software Architecture — Data Access Integration (OGSA-DAI) (данный пакет является программным обеспечением с открытыми исходными кодами, что позволяет дополнять его необходимыми модулями). Создан набор дополнений к пакету OGSA-DAI, позволяющий осуществлять подключение источников данных XML СУБД Sedna. В рамках проекта «Виртуальная обсерватория» — веб-средства предоставления доступа к данным grid-сред для конечных пользователей, — создана система самоописания структур разделов Виртуальной обсерватории и поисковый сервис, включающий в себя ряд улучшенных пользовательских функций.

10.1. Июнь 2007–2008 год: Первый этап исполнения суперкомпьютерной программы «СКИФ-ГРИД» Союзного государства

К весне 2007 года было закончено формирование и согласование, и в июне 2007 года началось исполнение суперкомпьютерной Программы[14] Союзного государства «СКИФ-ГРИД», в которой ИПС

[14]Исполнение: 2007–2010 годы, полное наименование программы: «Разработка и использование программно-аппаратных средств ГРИД-технологий и перспективных высокопроизводительных (суперкомпьютерных) вычислительных систем семейства «СКИФ»».

РАН является головным исполнителем от России. Программа включает четыре направления работ:

- *GRID-технологии:* исследование, создание, развитие и внедрение средств высокопроизводительных вычислений на основе GRID-технологий; поддержка гетерогенных, территориально-распределенных вычислительных комплексов.
- *Суперкомпьютеры семейства «СКИФ» (Ряд 3 и 4):* создание суперкомпьютеров «СКИФ» нового поколения на базе новых перспективных процессоров и вычислительных узлов, новых технических средств системной сети, управления системой, спецвычислителей и гибридных узлов, разработка соответствующего программного обеспечения.
- *Защита информации:* реализация (аппаратных и программных) средств защиты информации в создаваемых вычислительных комплексах.
- *Пилотные системы:* реализация прикладных систем в перспективных областях применения создаваемых высокопроизводительных вычислительных систем, решение актуальных задач на суперкомпьютерах и GRID-системах, усилия по подготовке и переподготовке кадров в области суперкомпьютерных и GRID-технологий.

К исполнению Программы привлечено большое количество организаций. В том числе, российских исполнителей программы — более 20, среди них: ГЦ РАН, ИКИ РАН, ИПМ им. М. В. Келдыша РАН, ИППИ РАН, ИПХФ РАН, ИХФ РАН, НИВЦ МГУ, НИИ КС, НИИФХБ МГУ, НИИЯФ МГУ, ННГУ, НПЦ «Элвис», ОИЯИ, ООО «Т-Платформы», ООО «ЮникАйСиз», ПензГУ, СПбАЭП, СПбГПУ, ТГУ, Химический факультет МГУ, ЧелГУ, ЮУрГУ. В последующих разделах обсуждаются только основные результаты первого этапа выполнения программы «СКИФ-ГРИД».

10.2. Разработка решений для Grid-технологий

Среди разработок для совершенствования системы доступа к научным данным скажем о «Виртуальной обсерватории». В этой области выполнены: развитие механизмов работы с источниками данных и метаданных, разработка дополнительных средств внешнего поиска, включая поддержку нечетких запросов, и обмена данными между логически взаимосвязанными виртуальными обсерваториями, а также улучшение пользовательского и административного интерфейсов.

Была выполнена еще одна интересная разработка — программное обеспечение SKIF@Home, которое позволяет развертывать вычислительные сети, использующие для высокопроизводительных, распределенных вычислений мощности неполностью загруженных персональных компьютеров (так называемых «компьютеров-доноров»). От аналогов (таких как, например, пакет BOINC — основы известных проектов SETI@Home и Folding@Home) ПО SKIF@Home отличает использование технологии виртуальных машин для обеспечения «контракта» между донором и организаторами сети: разграничение программного окружения вычислительных приложений и компьютера, на котором устанавливается ПО SKIF@Home, ограничения объемов памяти, используемой сетью. Разработана оригинальная методика ограничения доли процессорного времени: при помощи графического интерфейса администратор компьютера-донора способен задать ограничения с точностью до нескольких процентов. В состав разработанного ПО входит веб-сервис, проводящий точный учет ресурсов (процессорного времени, памяти), предоставленного для проведения расчетов отдельными донорами. Нами была показана работоспособность в прототипе среды SKIF@Home распределенных приложений, разработанных с использованием библиотеки X-Com, что открывает широкие возможности по использованию среды SKIF@Home.

В эти же годы сотрудниками института:

- была усовершенствована система предсказания отказов кластерных установок (СПОКУ);
- разработан ряд алгоритмов статистического анализа данных показаний сенсоров кластерных установок;
- разработана и реализована схема предсказания отказов оборудования на основе экстраполяции показаний сенсоров в ближайшее будущее (при этом вероятным отказом считается выход прогнозного значения за пределы расчетного интервала);
- разработана и реализована схема оценки качества предсказаний, а также сопоставления вновь получаемых данных со статистической моделью данной кластерной установки;
- разработана система оповещения администраторов установок о возможных отказах оборудования, усовершенствован веб-сайт, позволяющий администраторам просматривать и анализировать данные, накопленные в хранилище.

10.3. Дальнейшее развитие OpenTS

Система параллельного программирования OpenTS доработана до поддержки гетерогенных распределенных вычислительных архитектур (grid-систем, территориально-распределенных неоднородных вычислительных систем). В том числе, в OpenTS поддержана отказоустойчивость исполнения T-приложений в версии системы OpenTS для платформы Windows:

- доработанная в части отказоустойчивости реализация наиболее употребимого подмножества функций MPI — DMPI;
- доработанная в части отказоустойчивости реализация среды исполнения T++ программ.

Была разработана система генерации Web-сервисов для T-функций в системе OpenTS для платформы Windows Compute Cluster Server (WCCS). ИПС РАН также были разработаны средства визуализации процесса параллельного исполнения T++ программ для платформы WCCS, позволяющие наглядно отображать во время исполнения ключевые характеристики среды исполнения T++.

В системе OpenTS были поддержаны многоядерные вычислительные архитектуры под управлением ОС семейств UNIX и Windows. Осуществлен перенос под Windows модуля OpenTS, который позволяет T-программе эффективно задействовать все вычислительные ядра всех процессоров SMP-системы.

10.4. СКИФ-Полигон и пилотные прикладные системы

Помимо одиночных суперкомпьютерных установок упомянем еще об одном результате, который достигнут в рамках первого этапа исполнения программы «СКИФ-ГРИД»: это создание распределенной вычислительной системы «СКИФ-Полигон» (см. Рис. 2). «СКИФ-Полигон» объединяет суперкомпьютерные центры, в которых расположены суперЭВМ семейства «СКИФ» и научно-исследовательские центры, которым необходим доступ к вычислительным ресурсам. К осени 2008 года были объединены ресурсы суперкомпьютерных центров, расположенных в самых разных городах Беларуси и России: Минск, Москва, Переславль-Залесский, Челябинск, Томск, Ставрополь, Владимир и другие города. Суммарная пиковая производительность системы «СКИФ-Полигон» сегодня превышает 100 Tflops. В дальнейшем предусмотрено расширение состава участников «СКИФ-Полигона» и увеличение его производительности. Система «СКИФ-

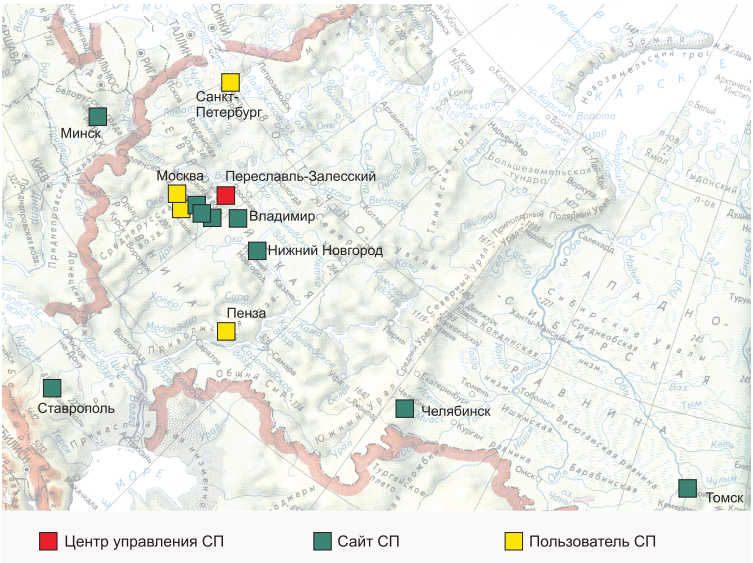


Рис. 2. Схема распределенной вычислительной системы СКИФ-Полигон

Полигон» является эффективной платформой, на которой проводятся разработки системного и прикладного программного обеспечения для суперЭВМ семейства «СКИФ». Разрабатываемые программные комплексы предназначены для выполнения научных и инженерных расчетов в таких важнейших областях как:

- исследования в области обработки данных в здравоохранении (например, система для обработки и хранения цифровых маммограмм);
- исследования в области проектирования лекарств с заданными свойствами;
- расчеты в интересах разработки новых материалов, в том числе — наноматериалов;
- исследования в области биоинформатики;
- расчеты, связанные с обеспечением безопасности атомных электростанций;
- расчеты в интересах различных разделов наук о Земле

Таблица 1. Основные характеристики СуперЭВМ «СКИФ МГУ»

Производительность пиковая/Linpack:	60 / 47.17 Tflops (КПД=78.6%)
Вычислительных узлов:	625
Число CPU Intel Xeon E5472, 3,0 GHz / ядер:	1250 / 5000
Суммарная память узлов RAM / HDD:	5.5 ТБ / 15ТБ
Конструктив узла:	blade
Монтажных шкафов вычислителя:	14
Системная сеть:	Infiniband DDR
Задержка при передаче пакетов данных:	$<$ 2.2 мкс
Вспомогательная сеть:	Gigabit Ethernet
Сервисная сеть:	СКИФ ServNet v.3 + IPMI
Занимаемая площадь:	96 кв.м.
Потребляемая мощность установки в целом:	520 кВт

- и другие исследования.

10.5. СуперЭВМ Ряда 3 семейства «СКИФ»

К марту 2008 года было завершено создание еще двух суперкомпьютеров Ряда 3 семейства «СКИФ»:

- «СКИФ МГУ», Linpack-производительность 47.17 Tflops; пиковая производительность 60 Tflops, Таблица 1;
- «СКИФ Урал», Linpack-производительность 12.2 Tflops, пиковая производительность 16 Tflops.

Обе эти суперЭВМ созданы на базе одинаковых технических решений, разработанных в рамках первого этапа программы «СКИФ-ГРИД». В суперЭВМ использованы самые передовые технические решения:

- четырехъядерные процессоры, выполненные по технологии 42 нм — Intel Xeon E5472, 3.0 GHz — использованы впервые в России;
- системная сеть DDR InfiniBand; вспомогательная сеть Gigabit Ethernet, сервисная сеть СКИФ ServNet v.3;
- высокая плотность упаковки в «СКИФ МГУ» и «СКИФ Урал» достигнута за счёт использования отечественных серверов-лезвий, имеющих плотность вычислительной мощности на 20% лучшую, чем у любых других серверов-лезвий.

Суперкомпьютеры Ряда 3 «СКИФ МГУ» и «СКИФ Урал» предназначены для решения задач со сложной логикой и большим количеством вычислений.

10.6. Вхождение суперЭВМ семейства «СКИФ» в рейтинг Top500

В рамках исполнения программы «СКИФ» и первого этапа программы «СКИФ-ГРИД» было выпущено девятнадцать опытных образцов суперкомпьютеров Рядов 1, 2 и 3 семейства «СКИФ», вне рамок Программ в различные организации России и Беларуси поставлено более 60 суперкомпьютеров семейства «СКИФ».

Пять суперкомпьютеров семейства «СКИФ» вошли в мировой рейтинг Top500 — пятисот самых мощных суперЭВМ мира. Это очень серьезный результат и для России, и для Беларуси, поскольку за всю историю развития вычислительной техники в СССР и в странах СНГ было всего шесть суперкомпьютеров, которые вошли в мировой рейтинг Top500 с признанием мировым сообществом их отечественного происхождения. Вот эти шесть суперЭВМ (см. Рис. 3):

- суперЭВМ «МВС 1000М» вошла в рейтинг Top500 в июне 2002 года заняв 64 место с производительностью 0.734 Tflops на тесте Linpack, пиковая производительность — 1.024 Tflops. Разработчики: НИИ «Квант», ИПМ имени М. В. Келдыша РАН, МСЦ РАН;
- суперЭВМ «СКИФ К-500» вошла в рейтинг Top500 в ноябре 2003 года заняв 405 место с производительностью 0.423 Tflops на тесте Linpack, пиковая производительность — 0.717 Tflops;
- суперЭВМ «СКИФ К-1000» вошла в рейтинг Top500 в ноябре 2004 года заняв 98 место с производительностью на тесте Linpack 2 Tflops, пиковая производительность — 2.534 Tflops;
- суперЭВМ «СКИФ Cyberia» вошла в рейтинг Top500 в июне 2007 года заняв 105 место с производительностью на тесте Linpack 9.013 Tflops, пиковая производительность — 12.002 Tflops;
- суперЭВМ «СКИФ Урал» вошла в рейтинг Top500 в июне 2008 года заняв 282 место с производительностью на тесте Linpack 12.2 Tflops, пиковая производительность 15.9 Tflops;

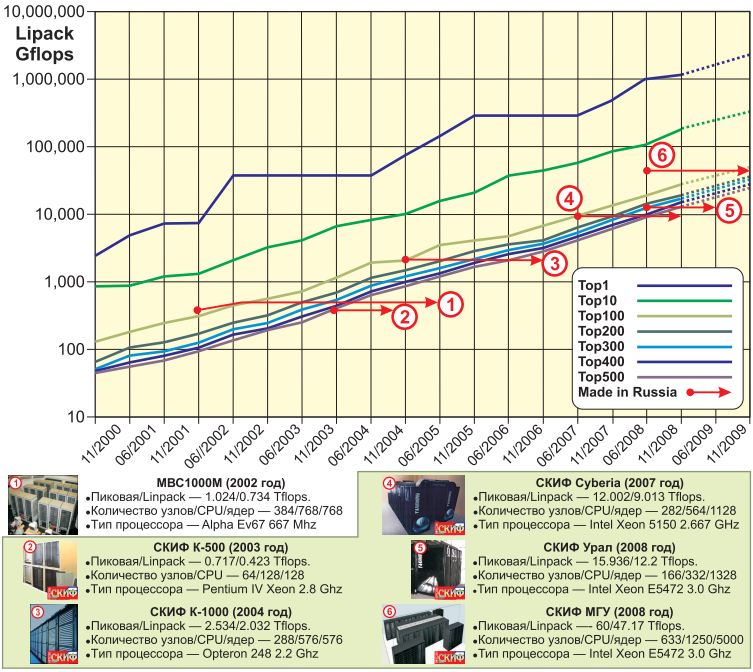


Рис. 3. Все шесть признанные отечественные суперЭВМ в Top500. Пять из шести — суперкомпьютеры семейства «СКИФ»

- суперЭВМ «СКИФ МГУ» вошла с рейтинг Top500 в июне 2008 года, заняв 36 место с производительностью на тесте Linpack 47.1 Tflops, пиковая производительность 60 Tflops.

Кроме этих шести, были и другие суперЭВМ на территории России, которые входили в список Top500, но они все являлись импортными вычислительными системами.

Еще раз подчеркнем: из шести вошедших в рейтинг Top500 отечественных суперЭВМ пять являются суперкомпьютерами семейства «СКИФ». То есть, развитие суперкомпьютерных технологий на территории СНГ сегодня в подавляющей доле обеспечено союзными суперкомпьютерными программами «СКИФ» и «СКИФ-ГРИД». Это иллюстрирует большое значение, которое играют научно-технические

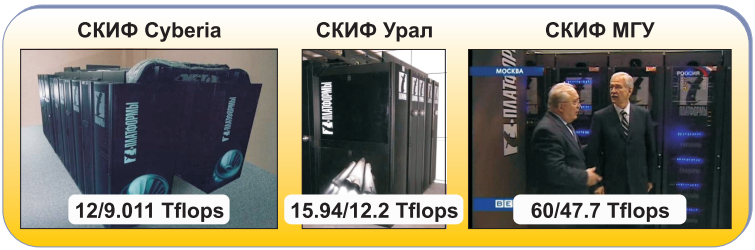


Рис. 4. СуперЭВМ семейства «СКИФ» Ряда 3, одновременно вошедшие в последнюю редакцию (июнь 2008 года) рейтинга Top500

программы Союзного государства в деле развития инновационных и наукоемких технологий в России и Беларуси.

Завершая обсуждение вхождения суперкомпьютеров семейства «СКИФ» в мировой рейтинг Top500 отметим следующее обстоятельство: в последней редакции (июнь 2008 года) рейтинга Top500 Россия впервые получила следующие результаты:

(1) В рейтинге Top500 одновременно присутствовало три отечественные суперЭВМ семейства «СКИФ» Ряда 3 (см. Рис. 4): «СКИФ МГУ», «СКИФ Урал» и «СКИФ Cyberia». До этого, как правило, в выпусках рейтинга либо находилась одна отечественная суперЭВМ, либо ни одной.
(2) СуперЭВМ «СКИФ МГУ» заняла в рейтинге 36 место — это самое высокое достижение российских суперЭВМ.
(3) В аналитическом отчете, которым сопровождается выпуск рейтинга Top500, Россия впервые вышла из категории «и другие страны» и была отмечена отдельно.

В Таблице 2 представлены основные параметры суперкомпьютеров семейства «СКИФ» Ряда 1, 2, 3 и 4. Кроме того в таблице указаны элементы суперЭВМ, являющиеся отечественной интеллектуальной собственностью — в дополнение к общим схемотехническим решениям, конструкторской и программной документации на суперЭВМ семейства «СКИФ», которые всегда разрабатываются нами и принадлежат нам.

Таблица 2. Суперкомпьютеры семейства «СКИФ» Ряд 1, 2, 3 и 4

Ряд	Годы, пиковая производительность (расчетный диапазон)	Ядер в CPU/ разрядность	Сетевые решения вспомогательной/ системной сети	Форм-фактор **(CPUs/U)**	Примечание
1	2000–2003, 0.020–0.5 TFlops	1/32	FastEthernet/ SCI (2D-тор), Myrinet	4U–1U **(0.5–2)**	Отечественный SCI (2D-тор). Охлаждение: воздух
2	2003–2007, 0.1–5 Tflops	1/32–64	GbEthernet/ SCI (3D-тор), Infiniband	1U, HyperBlade **(2)**	ServNet v.1, v.2. Ускорители: FPGA, ОВС. Охлаждение: воздух
3	2007–2008, 5–150 Tflops	2–4/64	GbEthernet/ Infiniband DDR	1U, blades 20 CPU в 5U **(2–4)**	ServNet v.3. Охлаждение: воздух-вода-фреон
4	2009–2011, 500–5 000 Tflops	4–8/64	Infiniband QDR/ отечественная системная сеть (3D-тор)	blades 64 CPU в 6U **(10.667)**	Новые подходы к охлаждению. Ускорители: FPGA, GPU, МЦОС...

11. Перспективы развития супекомпьютерных работ в ИПС РАН. Суперкомпьютеры Ряда 4 семейства «СКИФ»

Суперкомпьютеры Ряда 1, 2 и 3 — это разработки, которые уже выполнены в рамках программы «СКИФ» и первого этапа программы «СКИФ-ГРИД». На втором этапе запланирована разработка технических решений для суперЭВМ Ряда 4. Эта работа уже во многом сделана: выполнены большая часть исследований и часть конструкторских разработок. При наличии достаточного финансирования команда исполнителей программы «СКИФ-ГРИД» готова завершить

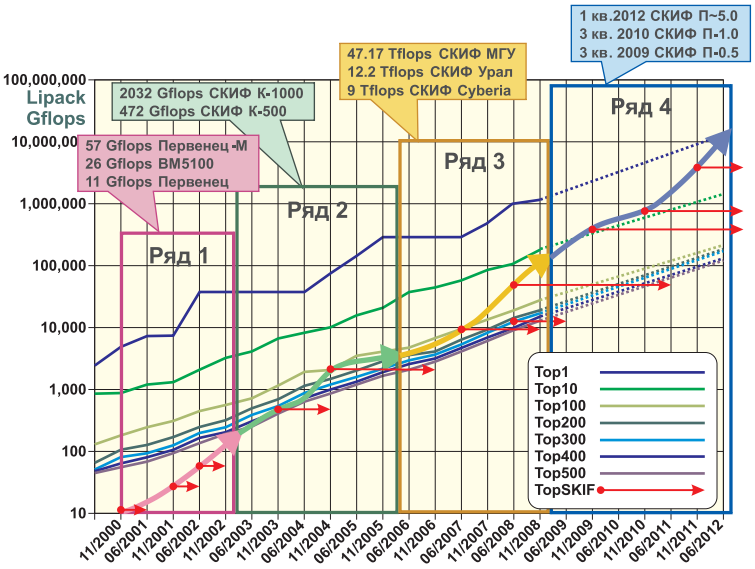


Рис. 5. Семейство суперЭВМ «СКИФ»: Ряд 1, 2, 3 и 4

все исследования, подготовить конструкторскую и программную документацию и выпустить опытные образцы с пиковой производительностью (см. Рис. 5):

- к ноябрю 2009 года — 0.5 Pflops;
- к ноябрю 2010 года — 1.0 Pflops;
- к марту 2012 года — 5.0 Pflops.

В любом случае, в рамках программы «СКИФ-ГРИД» мы подготовим технические решения, всю необходимую конструкторскую документацию, изготовим модули, из которых возможно собрать такие суперЭВМ. Будут ли суперЭВМ семейства «СКИФ» Ряда 4 с производительностью 0.5–5 Pflops изготовлены в указанные сроки — это вопрос наличия политической воли и вопрос расширения финансирования данных работ. Потому что данные суперЭВМ — 0.5, 1 и 5 Pflops, — требуют соответствующего дополнительного финансирования работ. Мы надеемся, что и в дальнейшем Союзное государство

и Россия будет оказывать необходимую поддержку отечественных суперкомпьютерных разработок.

Благодарности

За то, что посчастливилось работать в Институте программных систем Российской академии наук, автор благодарен судьбе и тем людям, которые способствовали такому повороту в судьбе, в первую очередь: Айламазяну А. К. и Кондратьеву Н. В.

Автор благодарен всем сотрудникам ИПС РАН, с которыми выпала честь работать вместе в 1986–2008 гг., и всем коллегам из организаций-партнеров по суперкомпьютерным исследованиям нашего института: БГУ, БГУИР, ГЦ РАН, ИВВиИС, ИКИ РАН, ИПМ им. М. В. Келдыша РАН, ИППИ РАН, ИПХФ РАН, ИХФ РАН, Компания СКС, НИВЦ МГУ, НИИ КС, НИИ механики МГУ, НИИФХБ МГУ, НИИЭВМ, НИИЯФ МГУ, ННГУ, НПЦ «Элвис», ОИПИ НАН Беларуси, ОАО «НИЦЭВТ», ОИЯИ, ООО «ЮникАйСиз», ПензГУ, РосНИИ РП, СПбАЭП, СПбГПУ, ТГУ, Химический факультет МГУ, ЦНТК РАН, ЧелГУ, ЮУрГУ и др.

Список литературы

[1] Абрамов С. М. Программа ДАРОС. Единая система электронных вычислительных машин. СПО ЕС2704. Руководство программиста. Ц5.00079-01 33 02. — М.: НИЦЭВТ, 1987. ↑5

[2] Абрамов С. М., Барбан А П, Сибиркова Л А. *Средства взаимодействия с машиной динамической архитектуры в ОС 7 ЕС* // 1-я Всесоюзная конференция «Проблемы создания суперЭВМ, суперсистем и эффект их применения», 15–17 сентября 1987 г. Т. **1**. — Минск: ИМ АН БССР, 1987, с. 97–98. ↑

[3] Абрамов С. М., Барбан А П, Михнушев Д П, Сибиркова Л А. *Программное обеспечение комплекса ЕС ЭВМ: машина динамической архитектуры* // Тематический сборник, серия ВТ. Вопросы РЭ, № 7, 1988, с. 1–25. ↑

[4] Абрамов С. М., Абакумов А А, Адамович А. И., Нестеров И А, Пименов С. П., Рядченко А В, Хаткевич М. И., Шевчук Ю. В. *Концепция разработки ПО МДА* // Тезисы докладов. Всесоюзная конференция молодых ученых и специалистов по проблемам кибернетики и вычислительной техники. — Москва–Переславль-Залесский, 1989. ↑

[5] Abramov S. M., Adamowitch A. I., Nesterov I. A, Pimenov S. P., Shevchuck Yu. V. *Principles of software tools implementation for multiprocessor with automatic dynamic parallelizing* // The 16th International School Programming'91. — Sofia, Bulgaria, 1991. ↑5

[6] Пономарев А. Ю. Интерфейсная плата DAD005. Техническое описание. — М.: НИЦЭВТ, 1994. ↑5

[7] Шевчук Ю. В. TTOLS. Электронный ресурс. — http://www.botik.ru/~sizif/ttools/, 1996. ↑5

[8] IPCA: Transputer: Software: Compilers: Gcc:.Электронный ресурс. Раздел «Internet Parallel Computing Archive». — http://wotug.kent.ac.uk/parallel/transputer/software/compilers/gcc/pereslavl/, 1996. ↑5

[9] Abramov S. M., Adamowitch A. I., Nesterov I. A, Pimenov S. P., Shevchuck Yu. V. *Autotransformation of evaluation network as a basis for automatic dynamic parallelizing* // The 6th NATUG Meeting, NATUG'1993. Spring Meeting «Transputer: Research and Application». — Vancouver, Canada: IOS Press, May 10. ↑5

[10] Абрамов С. М., Адамович А. И., Нестеров И. А, Пименов С. П., Шевчук Ю. В. *Автотрансформация вычислительной сети — основа для автоматического и динамического распараллеливания* // Теоретические и прикладные основы программных систем. — Переславль-Залесский: Институт программных систем РАН, 1994, с. 103–124. ↑5

[11] Adamowitch A. I., Nesterov I. A, Pimenov S. P., Shevchuck Yu. V. *cT: an Imperative Language with Parallelizing Features Supporting the Computation Model "Autotrasformation of the Evaluation Network"* // Parallel Computing Technologies: Third International Conference, 1995, с. 127–141. ↑5

[12] Nesterov I. A, Suslov I V *Towards programming of numerical problems within the system providing automatic parallelizing* // The 7th SIAM Conference on Parallel Proc-Essing for Scientific Computing. — San-Francisco, CA, 1995, с. 716. ↑5, 6

[13] Абрамов С. М., Адамович А. И. *Т-система — среда программирования с поддержкой автоматического динамического распараллеливания программ* // Программные системы. Теоретические основы и приложения ред. А. К. Айламазян. — М.: Наука-Физматлит, 1999, с. 201–214. ↑7

[14] Адамович А. И., Коваленко М. Р., Конышев А. П. *Реализация в Т-системе программы построения качественных изображений трехмерных сцен методом трассировки лучей* // Программные системы. Теоретические основы и приложения ред. А. К. Айламазян. — М.: Наука-Физматлит, 1999, с. 224–233. ↑7

[15] Абрамов С. М., Адамович А. И., Конышев А. П. *Т-система — среда программирования с поддержкой автоматического динамического распараллеливания программ. Тезисы докладов* // Десятая юбилейная международная конференция по вычислительной механике и современным прикладным программным средствам, Переславль-Залесский, 7–12 июня 1999 г. — М.: МГИУ, 1999, с. 14–15. ↑

[16] Абрамов С. М., Адамович А. И., Коваленко М. Р. *Т-система — среда програм-мирования с поддержкой автоматического динамического распараллеливания программ. Пример реализации алгоритма построения изображений методом трассировки лучей* // Программирование. — **25 (2)**, 1999, с. 100–107. ↑6, 7

[17] Абрамов С. М., Анищенко В. В., Парамонов Н. Н., Чиж О. П. *Разработка и опыт эксплуатации суперкомпьютеров семейства «СКИФ»* // Материалы конференции. I международная конференция «Информационные системы и технологии» (IST'2002), 5–8 ноября 2002 г. — Минск, 2002, с. 115–117. ↑9.1

[18] Абламейко С. В., Абрамов С. М. *Основные результаты суперкомпьютерной программы «СКИФ» Союзного государства* // АКИИ'03: Третий расширенный семинар «Использование методов искусственного интеллекта и высокопроизводительных вычислениях и в аэрокосмических исследованиях». Труды семинара. — М.: Физматлит, 2003. — ISBN 5-940-52-065-9, с. 135–140. ↑

[19] Абрамов С. М., Адамович А. И., Коваленко М. Р., Слепухин А. Ф., Парамонов Н. Н. *Кластерные системы семейства суперкомпьютеров «СКИФ»* // Научный сервис в сети Интернет: Труды Всероссийской научной конференции, 22–27 сентября 2003 г., Новороссийск. — М.: Изд-во МГУ, 2003, с. 147–151. ↑

[20] Абрамов С. М., Адамович А. И., Коваленко М. Р., Роганов В. А. *Биатлон для СКИФов: быстро и точно. Математика, информатика: теория и практика* // Сборник трудов, посвященный 10-летию Университета города Переславля ред. А. К. Айламазян. — Переславль-Залесский: Изд-во «Университет города Переславля», 2003. — ISBN 5-901795-02-4, с. 91–96. ↑9.1

[21] Абрамов С. М., Васенин В. А., Мамчиц Е Е, Роганов В. А., Слепухин А. Ф. *Динамическое распараллеливание программ на базе параллельной редукции графов. Архитектура программного обеспечения новой версии Т-системы* // Труды конференции. Всероссийская научная конференция «Высокопроизводительные вычисления и их приложения», 30 октября–2 ноября 2000 г., г. Черноголовка. — М.: Изд-во МГУ, 2000, с. 261–264. ↑9.2

[22] Абрамов С. М., Васенин В. А., Мамчиц Е Е, Роганов В. А., Слепухин А. Ф. *Архитектура программного обеспечения новой версии Т-системы* // Сборник научных трудов. Научная сессия МИФИ–2001, 22–26 января 2001 г. Т. **2**, 2001, с. 234. ↑

[23] Абрамов С. М., Кондратьева А В, Роганов В А, Чеповский А М. *Проектирование высокопроизводительного процессора обработки графов* // Информационные технологии. — **3**, 2001. ↑9.2

[24] Суперкомпьютерная программа «СКИФ». Электронный ресурс. Раздел «Результаты :: 2003 :: Управляющая сеть кластеров "СКИФ"». — http://skif.pereslavl.ru/, 2003. ↑9.4

[25] Плюснин В. У., Кушнеров Ф Р. Вычислительные системы с динамической архитектурой. Статья на Веб-сайте «Виртуальный компьютерный музей». — http://compmus9.valuehost.ru/histussr/dynaarc.htm, 04.03.2003. ↑5

Исследовательский центр мультипроцессорных систем ИПС имени А. К. Айламазяна РАН

S. M. Abramov. *HPC Researches in the Program Systems Institute of Russian Academy of Sciences: Retrospectives and Perspectives* // Proceedings of Program Systems institute scientific conference "Program systems: Theory and applications". — Pereslavl-Zalesskij, v. **1**, 2009. — p. 153–192. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. This article describes supercomputers technologies researches and developments conducted or planed in the Ailamazyan Program Systems Institute of Russian Academy of Sciences. The corresponding projects and its results since 1984 are discussed.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 004.382.2

С. М. Абрамов, В. Ф. Заднепровский, А. А. Московский, А. Б. Шмелев

Суперкомпьютеры Ряда 4 семейства «СКИФ»

Аннотация. В статье описывается проект создания отечественных суперЭВМ Ряда 4 семейства «СКИФ». В рамках создания данных суперЭВМ разрабатываются самые передовые суперкомпьютерные технологии: жидкостное охлаждение печатных плат, сверхплотная упаковка вычислительной мощности, новые решения в части системной, вспомогательной и сервисных сетей. Впервые в суперЭВМ Ряда 4 семейства «СКИФ» отечественная интеллектуальная собственность будет охватывать все конструкции, все печатные платы — то есть, все, кроме микросхем. Планируемая достижимая максимальная пиковая производительность данных суперЭВМ: 0.5 Pflops к осени 2009 года, 1 Pflops к осени 2010 года, более 5 Pflops к весне 2012 года.

1. Суперкомпьютерные программы Союзного государства

1.1. Суперкомпьютерная программа «СКИФ»

Суперкомпьютерная программа «СКИФ» [1] Союзного государства выполнялась в 2000–2004 годах. Полное название суперкомпьютерной программы — «Разработка и освоение в серийном производстве семейства моделей высокопроизводительных вычислительных систем с параллельной архитектурой (суперкомпьютеров) и создание прикладных программно-аппаратных комплексов на их основе», — точно определяло содержание работ.

В программе «СКИФ» головными исполнителями являлись: со стороны Беларуси — Объединенный институт проблем информатики Национальной академии наук Беларуси, со стороны России — Институт программных систем Российской академии наук. Заказчики-координаторы программы «СКИФ»: со стороны Беларуси — Национальная академия наук Беларуси, со стороны России — Федеральное агентство на науке и инновациям.

Суперкомпьютерная программа «СКИФ» — это серьезная, комплексная программа, в рамках которой по единой концепции создавался широкий спектр моделей семейства «СКИФ» и обеспечивалась возможность подбора конфигураций, оптимальных для различных

применений. Девятнадцать мероприятий программы покрывали все слои суперкомпьютерной отрасли:

- разработка и реализация аппаратных средств;
- разработка и реализация системного программного обеспечения;
- разработка и реализация инструментальных средств и законченных (пилотных) прикладных систем;
- вспомогательные мероприятия: подготовка и переподготовка кадров, создание и эксплуатация единого информационного пространства программы.

Институт программных систем РАН работы по созданию аппаратных и программных средств для семейства суперЭВМ «СКИФ» вел в тесном сотрудничестве с исполнителями от Республики Беларусь и с основными исполнителями Программы со стороны России, среди которых были:

- ОАО «Научно-исследовательский центр электронно-вычислительной техники» (НИЦЭВТ, Москва);
- Центр научных телекоммуникаций и информационных технологий (ЦНТК РАН, Москва);
- НИИ механики МГУ имени М. В. Ломоносова (Москва);
- Институт высокопроизводительных вычислений и информационных систем (ИВВиИС, СПб.);
- Российский НИИ региональных проблем (Переславль-Залесский, РосНИИ РП);
- Компания «Суперкомпьютерные системы» (СКС, Москва);
- НИИ Космических систем (Королев).

Программа «СКИФ» была признана одной из самых успешных программ Союзного государства. Результаты, достигнутые в ходе выполнения программы, получили и высокую правительственную оценку. За работу «Разработка конструкторской и программной документации, подготовка промышленного производства и выпуск образцов высокопроизводительных вычислительных систем (суперкомпьютеров) семейства «СКИФ» Ряда I и Ряда II» была присуждена премия Правительства Российской Федерации в области науки и техники за 2006 год группе исполнителей Программы «СКИФ».

1.2. Суперкомпьютерная программа «СКИФ-ГРИД»

Суперкомпьютерная программа «СКИФ-ГРИД» Союзного государства рассчитана на выполнение в 2007–2010 годах. Полное наименование программы: «Разработка и использование программно-аппаратных средств ГРИД-технологий и перспективных высокопроизводительных (суперкомпьютерных) вычислительных систем семейства «СКИФ»».

Как и в программе «СКИФ», головными исполнителями программы «СКИФ-ГРИД» являются: со стороны Беларуси — Объединенный институт проблем информатики Национальной академии наук Беларуси, со стороны России — Институт программных систем Российской академии наук.

Заказчики-координаторы программы «СКИФ-ГРИД»: со стороны Беларуси — Национальная академия наук Беларуси, со стороны России — Федеральное агентство на науке и инновациям. Программа «СКИФ-ГРИД» включает четыре направления работ:

- GRID-технологии: развитие, исследование, внедрение высокопроизводительных вычислений на основе GRID-технологий; поддержка гетерогенных, территориально-распределенных вычислительных комплексов.
- Суперкомпьютеры семейства «СКИФ» (Ряд 3 и 4): создание суперкомпьютеров «СКИФ» нового поколения на базе новых перспективных процессоров и вычислительных узлов, новых технических средств системной сети, управления системой, спецвычислителей и гибридных узлов, разработка соответствующего программного обеспечения.
- Защита информации: реализация (аппаратных и программных) средств защиты информации в создаваемых вычислительных комплексах.
- Пилотные системы: реализация прикладных систем в перспективных областях применения создаваемых высокопроизводительных установок, решение актуальных задач на суперкомпьютерах и GRID-системах, усилия по подготовке и переподготовке кадров в области GRID- и суперкомпьютерных технологий.

Программа «СКИФ-ГРИД» примерно в два–три раза крупнее программы «СКИФ» по масштабам: по количеству привлеченных

предприятий, по объемам запланированных работ и объемам ресурсов, привлекаемым для выполнения данных работ. Так, в исполнение программы «СКИФ» было вовлечено примерно по десять предприятий со стороны Беларуси и России. В программе «СКИФ-ГРИД» только со стороны Российской Федерации сегодня участвуют уже более 20 организаций. В том числе, российских исполнителей программы — более 20, среди них: ГЦ РАН, ИКИ РАН, ИПМ им. М. В. Келдыша РАН, ИППИ РАН, ИПХФ РАН, ИХФ РАН, НИВЦ МГУ, НИИ КС, НИИФХБ МГУ, НИИЯФ МГУ, ННГУ, НПЦ «Элвис», ОИЯИ, ОАО «Т-Платформы», ООО «ЮникАйСиз», ПензГУ, СПбАЭП, СПбГПУ, ТГУ, Химический факультет МГУ, ЧелГУ, ЮУрГУ, а от белорусской стороны в программе участвуют институты Национальной академии наук Беларуси: Институт математики, ОИЭЯИ, ИБОХ; ведущие государственные университеты: БГУ, БНТУ, БГУИР, ГрГУ; отраслевые институты, конструкторские бюро и ведущие предприятия Республики Беларусь, среди которых НИИ ЭВМ, «Белмикросистемы», «КБ системного программирования», «Минский моторный завод» и др.

1.3. Создание программного обеспечения суперЭВМ семейства «СКИФ»

Как правило, когда говорят о результатах программ «СКИФ» и «СКИФ-ГРИД», внимание уделяется аппаратным средствам, мощностям разработанных суперЭВМ, а также фактам вхождения в список пятисот самых мощных суперЭВМ мира (Top500). Это, действительно, очень важно, но хотелось бы отметить, что большая часть усилий, большее время, большие трудозатраты и значительная часть финансов в обеих программах были потрачены на создание программного обеспечения (ПО). Так, чтобы оценить масштабность комплекта программного обеспечения суперкомпьютеров семейства «СКИФ», перечислим, что в него уже на момент завершения программы «СКИФ» (2004 год) входили:

- системное ПО: операционная система; базовые библиотеки поддержки параллельного счета; файловые системы; системы очередей, мониторинга и управления; стандартные системы программирования — C, C++, Fortran; и т. п.;
- средства разработки параллельных программ — программные системы, инструментальные средства и библиотеки: MIRACLE, Open TS, Grace, и др.;

- два десятка параллельных прикладных систем для различных областей.

2. Роль суперкомпьютерных технологий в государствах с экономикой, основанной на знаниях

Прежде, чем обсуждать суперкомпьютерные технологии и супер-ЭВМ, разрабатываемые в программах «СКИФ» и «СКИФ-ГРИД», обсудим роль суперкомпьютерных технологий. Сегодня критические (прорывные) технологии в государствах, строящих экономику, основанную на знаниях, исследуются и разрабатываются на базе широкого использования высокопроизводительных вычислений [2, 3]. И другого пути — нет. Без серьезной суперкомпьютерной инфраструктуры:

- невозможно создать современные изделия высокой (аэрокосмическая техника, суда, энергетические блоки электростанций различных типов) и даже средней сложности (автомобили, конкурентоспособная бытовая техника и т. п.);
- невозможно быстрее конкурентов разрабатывать новые лекарства и материалы с заданными свойствами;
- невозможно развивать перспективные технологии (биотехнологии, нанотехнологии, решения для энергетики будущего и т. п.).

Сегодня суперкомпьютерные технологии по праву считаются, пожалуй, важнейшим фактором обеспечения конкурентоспособности экономики любой страны, а единственным способом победить конкурентов объявляют возможность обогнать их в расчетах. Здесь характерны слова Президента Совета по конкурентоспособности США: *«Технологии, таланты и деньги доступны многим странам. Поэтому США стоит перед лицом непредсказуемых экономических конкурентов из-за рубежа. Страна, которая желает победить в конкуренции, должна победить в вычислениях»*.

Неважно, о конкуренции в каком секторе экономики идет речь: сказанное верно для добывающих и перерабатывающих секторов экономики, и особенно это верно при разработке новых технологий. И поэтому в развитых странах мира для перехода к экономике знаний создается новая инфраструктура государства — государственная система из мощных суперкомпьютерных центров, которые объединены сверхбыстрыми каналами связи в грид-систему. То есть, по сути, речь

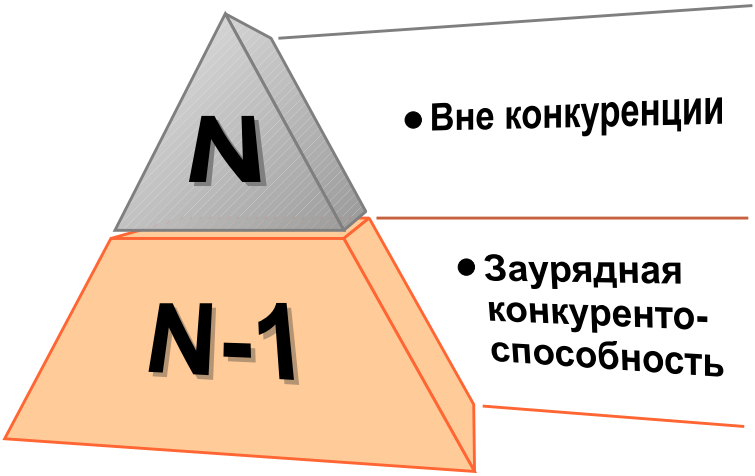


Рис. 1. Соответствие между уровнем (N или N-1) суперкомпьютерных технологий суперЭВМ и уровнем конкурентоспособности разработок, создаваемых при помощи данной суперЭВМ

идет о национальной научно-исследовательской информационно-вычислительной сети. Для такой системы часто используют термин *киберинфраструктура*. В этих странах на создание национальной киберинфраструктуры выделяются большие финансы из государственных бюджетов: в 2005–2008 гг. США тратили на эти цели 2–4 млрд. долларов в год.

Тем самым, краткое определение сегодняшней роли суперкомпьютерных технологий может быть таким: это ключевая критическая технология, единственный инструмент, дающий возможность победить в конкурентной борьбе.

В каждый момент времени, если посмотреть уровень развития суперкомпьютерной отрасли (например, список Top500), то можно выделить два слоя (см. Рис. 1):

- *Технологии уровня «N»*. Инновационные, совершенно новые суперкомпьютеры, которые сильно вырываются вперед. Они сделаны по технологиям будущего, которые еще не вполне освоены, а только-только разрабатываются в мире. Такие

машины соответствуют первым 10–20 местам списка Top500. Эти суперЭВМ обладают мощностью, которая радикально отличает их от всех других машин. И на платформе таких суперЭВМ можно выполнить вычисления — разработать новые материалы, новые технологические решения, — которые позволят обладающей ими стране быть вне конкуренции и существенно оторваться от других производителей материалов, лекарств, механизмов и тому подобного.

- *Технологии уровня «N-1»*. Технологии более низкого уровня, отработанные решения, воспроизводить их способны многие страны. Соответственно, расчеты, выполняемые на таких машинах, позволяют достичь нормальной (обычной, заурядной) конкурентоспособности. То есть, позволяют создавать материалы, механизмы, решения такие же, как и у многих других стран. В данном случае мы получаем рядовую конкурентоспособность: с разработками можно выходить на мировой рынок, на котором нам придется вести изнурительную конкурентную борьбу с десятком подобных разработок.

Надо честно отметить, что разработанные в предыдущие годы суперЭВМ Рядов 1–3 относились к технологическому уровню N-1. А вот суперЭВМ Ряда 4 планируется разрабатывать на технологическом уровне N.

2.1. СуперЭВМ семейства «СКИФ» Рядов 1, 2 и 3

Суперкомпьютеры семейства «СКИФ» [2,4] выпускались отдельными группами, называемыми «рядами» (Таблица 1, Рис. 2). На сегодняшний день:

- разработаны и выпущены опытные и серийные образцы суперЭВМ Рядов 1, 2 и 3;
- проработаны технические решения, подготовлена эскизная конструкторская документация суперЭВМ Ряда 4 семейства «СКИФ».

2.2. Ряд 1 суперЭВМ семейства «СКИФ»

Конструкторская документация для суперЭВМ Ряда 1 (семейства «СКИФ»), а также их опытные образцы разрабатывались и выпускались в 2000–2003 годах. Решения, которые были при этом выработаны, были способны обеспечивать мощность суперЭВМ 20–500 GFlops[1].

Для этих суперЭВМ были характерны следующие технические решения:

- использовались 32-х-разрядные одноядерные CPU;
- для системной сети использовался SCI (2D-тор) и Myrinet;
- как вспомогательная сеть использовался FastEthernet;
- форм-фактор для вычислительных узлов в данных суперЭВМ — монтируемые в стойки корпуса — от 4U до 1U.

В эти же годы была разработана и освоена в производство отечественная системная сеть — плата SCI (2D-тор). Работа была выполнена исполнителем программы «СКИФ» ОАО НИЦЭВТ.

2.3. Ряд 2 суперЭВМ семейства «СКИФ»

Конструкторская документация и опытные образцы Ряда 2 суперЭВМ семейства «СКИФ» разрабатывались и выпускались в 2003–2007 годах. Полученные здесь решения позволяли выпускать суперЭВМ мощностью 0,1–5 Tflops.

Для этих суперЭВМ были характерны следующие технические решения:

- использовались одноядерные CPU как 32-х-разрядные, так и 64-х-разрядные (для старших моделей Ряда 2);
- в качестве системной сети использовались сети SCI (3D-тор) и Infiniband;
- в качестве вспомогательной сети использовался GbEthernet;
- существенно повысилась плотность упаковки вычислительной мощности — использовались серверы с форм-фактором 1U и даже так называемые решения Hyper-Blade.

[1]Единицы производительности суперЭВМ: 1 Gflops — миллиард операций с плавающей точкой в секунду, 1 TFlops = 1 000 GFlops — триллион операций с плавающей точкой в секунду, 1 Pflops = 1 000 TFlops — тысяча триллионов операций с плавающей точкой в секунду.

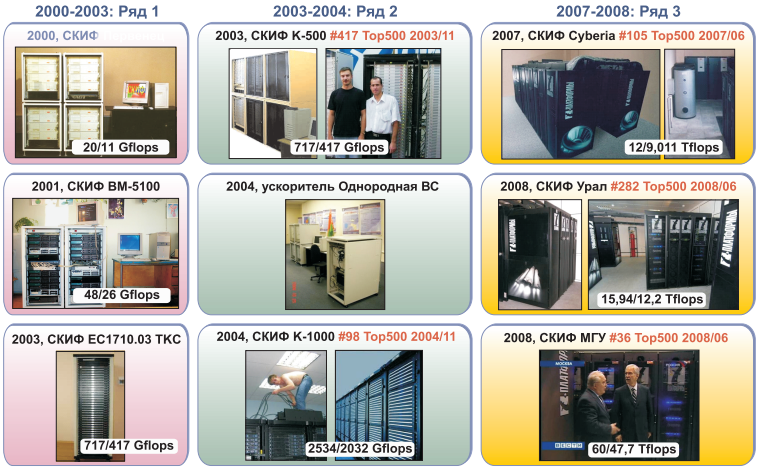


Рис. 2. Ряды 1, 2, и 3 семейства суперЭВМ «СКИФ»

Отметим, что в эти же годы были разработаны системы управления и мониторинга суперкомпьютеров ServNet v.1 и ServNet v.2 (разработка ИПС РАН). Также начались работы по изучению и применению ускорителей, как построенных на FPGA, так и ускорителей, выполненных полностью на отечественной элементной базе (так называемые однородные вычислительные системы, ОВС).

2.4. Ряд 3 суперЭВМ семейства «СКИФ»

Конструкторская документация и опытные образцы Ряда 3 [4] суперЭВМ семейства «СКИФ» разрабатывались в 2007–2008 гг. Полученные здесь технические решения позволяют строить суперкомпьютеры с производительностью 5–150 Tflops. Для этих суперЭВМ были характерны следующие технические решения:

- использовались 2–4-х-ядерные 64-х-разрядные CPU;
- в качестве системной сети использовалась сеть Infiniband DDR;
- в качестве вспомогательной сети использовался GbEthernet;

- в младших моделях использовались монтируемые в 19"монтажный шкаф серверы с форм-фактором 1U, в старших моделях использовались отечественные blade-решения, позволяющие в 5U упаковывать 10 вычислительных узлов.

Для данных суперЭВМ использовалась новая версия управляющей сети — ServNet v.3 (разработка ИПС РАН). Повысилась плотность упаковки процессоров до уровня 4 CPU на 1 U. Соответственно повысилась и плотность выделения тепловой энергии на единицу объема. И если до этого в суперЭВМ семейства «СКИФ» использовалось воздушное охлаждение, то в машинах Ряда 3 уже использовалось трехконтурное охлаждение «воздух–вода–фреон».

3. Что есть отечественного в суперЭВМ семейства «СКИФ»?

Когда обсуждаются суперЭВМ семейства «СКИФ», то всегда задается вопрос: «А что отечественного есть в этих суперЭВМ, ведь в них же используются импортные комплектующие?» Это правда, пока еще в странах-участниках Союзного договора не развито производство необходимых для суперЭВМ отечественных микропроцессоров и сопутствующих комплектующих. В результате приходится использовать импортную элементную базу. Впрочем, такая ситуация не является исключением. Суперкомпьютеры (впрочем, как и компьютеры) — технически сложные устройства. Как правило, такого рода изделия создаются с широким использованием мирового распределения труда. Общая практика, когда в суперЭВМ, разрабатываемой одной компанией некоторой страны, широко используются компоненты, разработанные и производящиеся в самых различных компаниях в разных странах мира. В настоящее время ни одна страна мира (за исключением разве что США), не производит все без исключения компоненты компьютерной техники и суперкомпьютеров в частности.

В полном соответствии с данной тенденцией суперкомпьютеры семейства «СКИФ» основываются на использовании передовой зарубежной элементной базы, что позволяет обеспечить конкурентоспособность по такому важнейшему параметру как производительность. Суперкомпьютеры семейства «СКИФ» разрабатываются, собираются, налаживаются и тестируются нашими специалистами. При этом

Таблица 1. Суперкомпьютеры семейства «СКИФ» Ряд 1, 2, 3 и 4

Ряд	Годы, пиковая производительность (расчетный диапазон)	Ядер в CPU/ разрядность	Сетевые решения вспомогательной/ системной сети	Форм-фактор **(CPUs/U)**	Примечание
1	2000–2003, 0.020–0.5 TFlops	1/32	FastEthernet/ SCI (2D-тор), Myrinet	4U–1U **(0.5–2)**	Отечественный SCI (2D-тор). Охлаждение: воздух
2	2003–2007, 0.1–5 Tflops	1/32–64	GbEthernet/ SCI (3D-тор), Infiniband	1U, HyperBlade **(2)**	ServNet v.1, v.2. Ускорители: FPGA, ОВС. Охлаждение: воздух
3	2007–2008, 5–150 Tflops	2–4/64	GbEthernet/ Infiniband DDR	1U, blades 20 CPU в 5U **(2–4)**	ServNet v.3. Охлаждение: воздух-вода-фреон
4	2009–2011, 500–5 000 Tflops	4–8/64	Infiniband QDR/ отечественная системная сеть (3D-тор)	blades 64 CPU в 6U **(10.667)**	Новые подходы к охлаждению. Ускорители: FPGA, GPU, МЦОС...

Союзное государство является собственником конструкторской документации на узлы суперЭВМ семейства «СКИФ» и на изделия целиком. На часть разработок имеются патенты. Это еще одно документальное подтверждение оригинальности отечественных разработок.

Независимая экспертиза страны происхождения суперЭВМ выполняется и при включении суперЭВМ в рейтинг Top500. Поданные заявителем сведения о стране происхождения и о производителе проверяются составителями списка и, если нужно, исправляются — такие случаи известны. Во всех случаях вхождения всех суперЭВМ

семейства «СКИФ» данная проверка страны происхождения проходила успешно — составители списка оставляли без изменения сведения об отечественном происхождении суперЭВМ семейства «СКИФ»: «СКИФ К-500», «СКИФ К-1000», «СКИФ Cyberia», «СКИФ МГУ» и «СКИФ Урал»[2].

В целом, за всю историю Top500 отечественное происхождение [2] признавалось только у этих пяти суперЭВМ семейства «СКИФ» и еще у «МВС-1000М» (НИИ «Квант», редакции рейтинга 06/2002–06/2004). Все остальные установленные в России системы, попавшие в Top500, являются импортными — производства: IBM, Sun Microsystems и Hewlett-Packard. Еще одно объективное доказательство отечественного происхождения суперЭВМ семейства «СКИФ» — превышение зарубежных аналогов по показателям. Если некоторая суперЭВМ обладает характеристиками, которые превышают достижения отрасли, то это является неоспоримым доказательством уникальности, оригинальности установки. СуперЭВМ семейства «СКИФ» часто показывали лучшие в отрасли результаты. Например:

- «СКИФ К-500», «СКИФ Cyberia», «СКИФ МГУ», «СКИФ Урал» продемонстрировали лучший показатель КПД для суперЭВМ на процессорах Intel. Да, в суперЭВМ семейства «СКИФ» используются импортные процессоры, но отечественным разработчикам удается их использовать лучше, чем кому бы то ни было!
- В ноябре 2004 «СКИФ К-1000» занял первое место в мире на тесте «столкновение 3 автомобилей» в рейтинге TopCrunch (`www.topcrunch.org`, поддержан DARPA).
- В феврале 2007 «СКИФ Cyberia» выдает показатели лучшие, чем у современных суперЭВМ (Cray, HP, IBM, SUN): лучший (на 8..13%) КПД, лучшую (в 2–1.5 раза) масштабируемость на прикладном инженерном пакете STAR-CD.

Часто разработанные нами решения превышают зарубежные аналоги и по техническим возможностям:

- blade-решение по суперЭВМ Ряда 3 «СКИФ МГУ», «СКИФ Урал» имело (на момент выпуска): плотность упаковки вычислительной мощности процессоров Intel — на 20% лучше всех аналогичных изделий в мире; стандартный разъем PCI

[2]Редакции рейтинга 11/2003, 11/2004–06/2006, 06/2007, 11/2007, 06/2008, 11/2008.

Express; «N+1» резервирование и «горячую замену» как блоков питания, так и вентиляторов. Такое сочетание этих важных эксплуатационных свойств встречалось только в данной blade-системе;
- система управления СКИФ ServNet версии 1, 2, 3 (разработана в ИПС РАН) поддерживает ряд уникальных возможностей. Например, функцию «черного ящика» — сохранение последних записей о событиях в отказавшем блоке.

СуперЭВМ семейства «СКИФ» являются отечественными системами, разработанными на базе импортных комплектующих, с постепенно нарастающей долей импортозамещения. В суперкомпьютерах «СКИФ» Ряда 1 отечественными были:

- схемотехнические решения;
- конструкторская документация (КД) корпусов и стоек, при этом стойки и корпуса выпускались в Минске;
- программное обеспечение (ПО) кластерного уровня семейства «СКИФ» — ПО КУ СКИФ.

При этом набор отечественного базового программного обеспечения (ПО КУ СКИФ) создавался и на основе оригинальных разработок, и на основе доработок и адаптации программного обеспечения с открытыми исходными текстами. СуперЭВМ «СКИФ» Ряда 2 также разработаны по оригинальному проекту. И здесь отечественными являлись:

- схемотехнические решения;
- конструкторская документация (КД) корпусов и стоек;
- разработка и программное обеспечение — ПО КУ СКИФ.

Кроме того:

- отдельные компоненты узлов были доработаны по документации наших разработчиков — например, материнские платы для «СКИФ К-500» и «СКИФ К-1000»;
- суперЭВМ «СКИФ» Ряда 2 оснащались сетью управления и мониторинга отечественной разработки — ServNet версии 1 и 2, разработка ИПС РАН;
- в суперкомпьютере «СКИФ ЕС1710.03» использовался интерконнект отечественного производства (НИЦЭВТ, интерконнект SCI 2D-тор).

Суперкомпьютеры «СКИФ» Ряда 3 «СКИФ МГУ» и «СКИФ Урал» созданы на основе blade-серверов отечественной разработки, имеющих уникальные показатели. Таким образом, здесь отечественными были:

- схемотехнические решения;
- конструкторская документация (КД) на сами blade-серверы и шасси;
- программное обеспечение — ПО КУ СКИФ;
- конструкторская и программная документация на сервисную сеть ServNet версии 3 (платы ServNet T-60 и ServNet CMB).

Тем самым, суперЭВМ Рядов 1–3 по праву называют отечественными. Правда, надо заметить, что в них использовались целые блоки, на которые отсутствовала и отечественная конструкторская документация, и интеллектуальная собственность (включая право на производство и право на модификацию). И к таким блокам относились не только элементная база (не только микросхемы), но и, например, практически все печатные платы. За исключением ServNet (разработанного ИПС РАН) все печатные платы (материнские, соединительные и т. п.) суперЭВМ Рядов 1–3 были импортными. В рамках реализации суперЭВМ Ряда 4 семейства «СКИФ» планируется серьезно изменить данное положение вещей — подробнее ниже, в разделе 4.8.

4. СуперЭВМ семейства «СКИФ» Ряда 4

Конструкторская документация и опытные образцы Ряда 4 суперЭВМ семейства «СКИФ» запланированы к разработке в 2008–2012 гг. Данные суперЭВМ будут иметь пиковую производительность 200–5 000 Tflops (0.2–5 Pflops) и выше (см. Рис. 3).

В суперкомпьютерах Ряда 4 семейства «СКИФ» предусмотрены самые современные решения[3]:

- в вычислительных узлах использованы многоядерные (4–8 ядер и выше) 64-х-битные процессоры стандартной архитектуры x86. В дополнение к ним в узле предусмотрена ПЛИС, ресурсы которой могут быть использованы для ускорения специализированных алгоритмов;

[3]По сути, речь идет о разработке технологий уровня N.

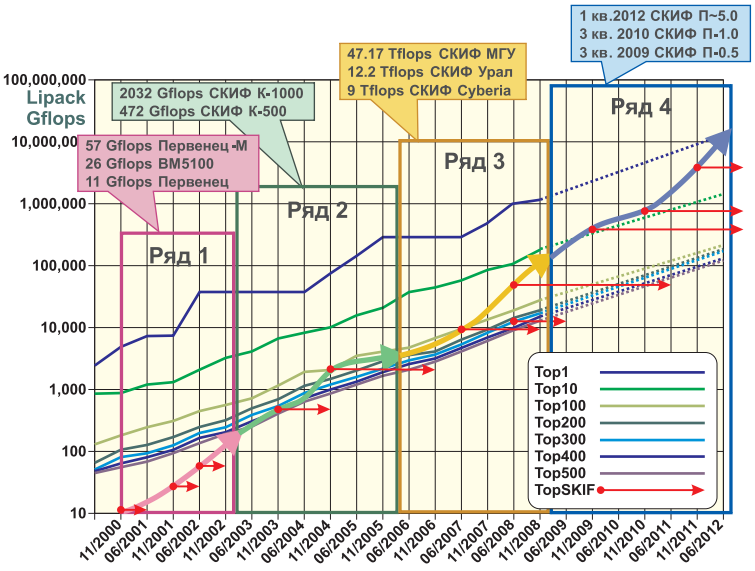


Рис. 3. Семейство суперЭВМ «СКИФ»: Ряд 1, 2, 3 и 4

- достигнута более высокая плотность упаковки вычислительной мощности. Разработаны оригинальные blade-системы, позволяющие упаковать более 10 процессоров в 1U стоечного пространства;
- такая высокая плотность упаковки требует новых подходов к охлаждению вычислительной установки. В СуперЭВМ семейства «СКИФ» Ряда 4 применяется система охлаждения вычислительных узлов с использованием воды либо этиленгликоля;
- в качестве системной сети в суперЭВМ используется отечественная системная сеть (3D-тор на базе FPGA), а в качестве вспомогательной сети — Infiniband QDR или 10GbEthernet.

В дальнейших разделах подробно обсуждаются различные характеристики суперЭВМ Ряда 4 семейства «СКИФ».

4.1. Производительность, компактность, надежность

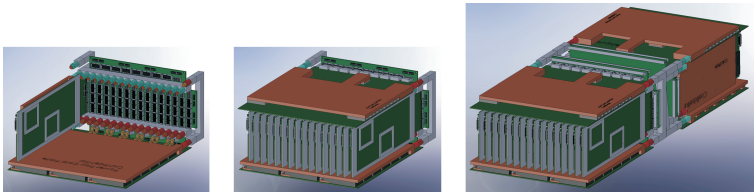
СуперЭВМ высокой производительности по необходимости содержит большое количество узлов. При росте числа вычислительных узлов критическими становятся такие параметры как надежность и размер установки (с ростом физических размеров растет задержка при передаче данных в системной сети, что снижает характеристики суперЭВМ). К счастью, и повысить надежность, и уменьшить размер установки удается одним и тем же приемом: повышение плотности упаковки вычислительных узлов. По мере того как все большее количество вычислительных узлов упаковывается в рамки монтажного шасси, мы достигаем следующих эффектов:

- уменьшаются физические размеры установки, уменьшаются длины соединительных линий между вычислительными узлами, уменьшаются задержки;
- большое количество соединений выполняется в рамках монтажного шасси. Такие соединения выполнены либо в виде контактных дорожек на печатных платах, либо в виде соединений через разъемы соединительной печатной платы (backplane). Таким образом, происходит существенное снижение количества соединительных кабелей и кабельных разъемов в системе, за счет чего серьезно улучшается надежность.

В суперкомпьютерах Ряда 4 семейства «СКИФ» в шасси с размером 6U входят (см. Рис. 4) две соединительные панели (backplane), к которым подключены две группы печатных плат, каждая из которых включает:

- плату поддержки электропитания (Рис. 5);
- корневую плату, содержащую средства управления и мониторинга аппаратурой шасси и коммутатор Infiniband QDR (Рис. 5);
- 16 плат-лезвий вычислительных узлов (Рис. 7, раздел 4.5).

Существенная часть соединений вычислительных узлов в системной сети и во вспомогательной сети (Infiniband QDR) выполнены в рамках шасси за счет соединительной панели (не при помощи кабельных соединений). В моделях СКИФ 4/Н и СКИФ 4/В суперкомпьютеров семейства «СКИФ» Ряда 4 шасси содержит 32 двухпроцессорных вычислительных узла.



(1) *Начало сборки:* снизу плата блоков питания с охлаждающей пластиной, слева — вычислительный узел, сзади — соединительная плата (backplane);
(2) *Собрана половина шасси:* снизу плата блоков питания с охлаждающей пластиной, в середине — 16 вычислительных узлов, сверху — корневая плата с охлаждающей пластиной, сзади — соединительная плата (backplane);
(3) *Две половины объединены в одно шасси 6U:* для модели СКИФ 4/Н — 32 вычислительных узла, 64 процессора Intel Nehalem, 256 ядер.

Рис. 4. Схемы компоновки шасси суперЭВМ Ряда 4 семейства «СКИФ»

При работе передняя и задняя стороны шасси закрываются сенсорными ЖКД-мониторами, при помощи которых поддерживаются отображение состояния и управление аппаратурой шасси. Шасси не содержит подвижных частей (вентиляторов, механических дисков), является законченной (стационарной или возимой) суперЭВМ (пиковая производительность 3 Tflops для модели СКИФ 4/Н) и блоком для более крупных систем.

4.2. Охлаждение: передовые решения

Такая высокая плотность упаковки требует новых подходов к охлаждению вычислительной установки. В суперЭВМ Ряда 4 семейства «СКИФ» применена система непосредственного водяного охлаждения вычислительных узлов.

Решения подобного класса сегодня, несомненно, относятся к технологиям уровня N. Лидеры в области суперкомпьютерных технологий переходят от уже освоенных схем охлаждения «вода на уровне

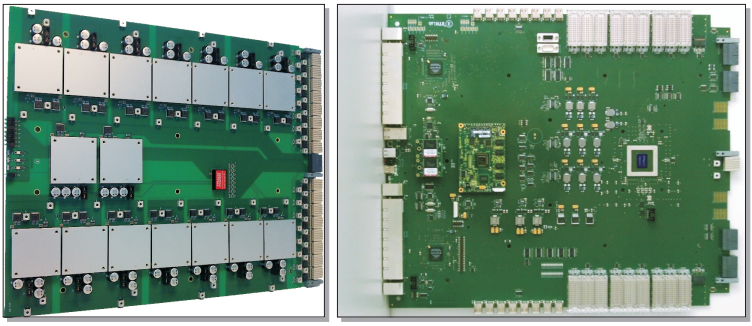


Рис. 5. Плата электропитания (слева) и корневая плата (справа) шасси для суперЭВМ Ряда 4 моделей СКИФ 4/Н и СКИФ 4/В

шкафа», «горячий коридор», «воздух–вода–фреон» к новым подходам к охлаждению вычислительной установки. Примером могут послужить разработки SGI (система охлаждения Kelvin), водяное охлаждение процессоров у фирм IBM и Fujitsu и разработки компаний Cray и IBM по использованию фазового перехода (испарения) как способа охлаждения микросхем.

Заметим, что, по сравнению со схемами охлаждения, где в качестве теплоносителя используется воздух, у водяного охлаждения имеется ряд серьезных преимуществ:

- данная схема охлаждения требует как минимум в 2 раза меньше энергозатрат;
- при остановке циркуляции теплоносителя за счет большей теплоемкости вода в течение некоторого времени сохраняет способность охлаждать микросхемы;
- система охлаждения в вычислителе не содержит ни одной механической подвижной части. Это повышает надежность установки и ее эргономические качества (бесшумность).

4.3. Модели Ряда 4 и повторное использование разработок

СуперЭВМ Ряда 4 запланированы к разработке и производству в течение 2008–2012 гг. За это время произойдет выпуск как минимум

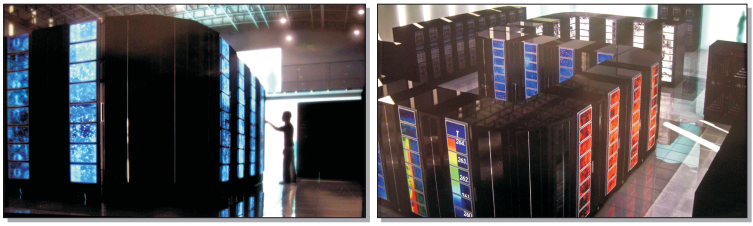


Рис. 6. Компьютерная модель суперЭВМ СКИФ П-0.5 Ряда 4 семейства «СКИФ» с производительностью 500 Tflops

трех различных семейств микропроцессоров. Основываясь на прогнозах и планах ведущих компаний, мы предусматриваем выпуск четырех последовательностей моделей в рамках Ряда 4: «СКИФ 4/Н», «СКИФ 4/В», «СКИФ 4/С», «СКИФ 4/П». Ожидается, что к концу жизненного цикла Ряда 4 плотность упаковки вырастет более чем в 8 раз, а энерго-эффективность (производительность на Ватт) более чем в 5 раз по сравнению с сегодняшним днем.

При этом предусмотрено широкое повторное использование конструкторской документации различных блоков и модулей. Так, для всех моделей одинаковыми будут являться все конструкции и соединительная инфраструктура шкафа и шасси, а также большинство печатных плат: соединительные, корневые и подсистемы электропитания. Изменяться будут (и то лишь частично) только печатные платы вычислительных узлов.

4.4. Не просто рекордные установки, а широкий ряд изделий

Каждая последовательность моделей охватывает широкий спектр производительности от нескольких Tflops до нескольких тысяч Tflops и предусматривает доступность для потребителя трех видов изделий:

- *Персональная или мобильная суперЭВМ* представляет собой одно шасси, которое можно расположить на рабочем месте сотрудника, тем более что это изделие бесшумное и имеет вполне приемлемое (для рабочего места) электропотребление. Пиковая производительность такого вычислителя может быть до трех Tflops. Заметим, что вся коммутация

вычислительных узлов системной и вспомогательной сети уже выполнена в рамках шасси. Шасси является первым уровнем законченного изделия и строительным блоком для более крупных систем (шкаф, система из нескольких шкафов).

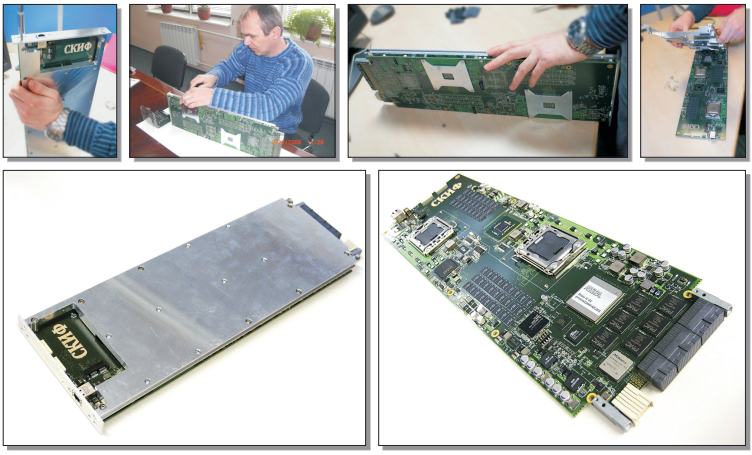
- *СуперЭВМ для лабораторий (конструкторских отделов и т. п.)* и региональных ВУЗов представляет собой один или несколько шкафов, каждый из которых содержит от двух до восьми шасси и всю необходимую соединительную инфраструктуру для них — соединения системной сети, вспомогательной сети, сервисной сети, подсистем электропитания и охлаждения. Пиковая производительность такого вычислителя может быть от шести до 350 Tflops.
- *Суперкомпьютерная система* для крупных суперкомпьютерных национальных центров представляет собой несколько (15 и более) шкафов, объединенных общей инфраструктурой — соединения системной сети, вспомогательной сети, сервисной сети, подсистем электропитания и охлаждения. Пиковая производительность такого вычислителя может быть от 350 Tflops до 10 Pflops (см. Рис. 6).

Таким образом, суперкомпьютеры Ряда 4 семейства «СКИФ» охватывают большое разнообразие областей применения и широкий диапазон производительности.

4.5. Вычислительный узел моделей

В состав вычислительного узла суперкомпьютер Ряда 4 семейства «СКИФ» входит:

- два современных стандартных (x86) многоядерных (четыре ядра и больше) 64-х-разрядных микропроцессора;
- память (RAM) объемом 6, 12 или 24 Гбайт;
- микросхема адаптера (NIC) Infiniband QDR;
- твердотельный жесткий диск (SSD) для хранения образа операционной системы, вспомогательных файлов, записей контрольных точек и раздела для организации виртуальной памяти;
- микросхема FPGA, которая используется, с одной стороны, для организации системной сети, а, с другой стороны, оставшиеся свободными ресурсы FPGA могут быть использованы для ускорения некоторых вычислений.



- *Верхний ряд:* первичный осмотр и разборка вычислительного узла, февраль 2009;
- *Нижний ряд слева:* вычислительный узел с охлаждающей пластиной;
- *Нижний ряд справа:* вычислительный узел без охлаждающей пластины.

Рис. 7. Вычислительный узел суперЭВМ Ряда 4 семейства «СКИФ» моделей СКИФ 4/Н и СКИФ 4/В

Все компоненты вычислительного модуля размещаются на одной печатной плате. К этой печатной плате прижимается (вплотную ко всем микросхемам) так называемая охлаждаемая пластина, через которую организован поток охлаждающей жидкости (см. Рис. 7).

4.6. Больше, чем просто системная сеть

Системная сеть в суперЭВМ Ряда 4 организована с использованием FPGA. В качестве топологии для системной сети используется трехмерный тор. За счет прошивки FPGA и его подключения к различным компонентам системы реализуется:

- быстрый обмен между FPGA и системной шиной вычислительного модуля, например, PCI Express;
- шесть двусторонних каналов, позволяющих объединять вычислительные узлы по топологии трехмерный тор;

- аппаратный маршрутизатор сообщений в системной сети топологии 3D тор;
- аппаратная поддержка коллективных операций библиотеки MPI, например, all_reduce.

Тем самым, разрабатывается масштабируемая в широких пределах системная сеть с явными чертами технологического уровня N.

Упомянем также, что в суперЭВМ Ряда 4 семейства «СКИФ» будут использованы еще две независимые сети, аналоги которых встречаются только в топовых моделях суперкомпьютеров (уровня N):

- отдельная сеть для реализации операций барьерной синхронизации;
- отдельная подсистема синхронизации системных часов всех микропроцессоров в вычислителе.

Среди прочего это позволяет на уровне операционной системы реализовать поддержку контрольных точек.

4.7. Мониторинг и управление

Для обеспечения высокой надежности в суперЭВМ Ряда 4 запланировано использовать расширенный состав сенсоров, располагаемых на различных печатных платах вычислителя, и три независимые сенсорные сети — сети мониторинга и управления. Опуская подробности, упомянем, что третья из этих сетей является новой версией сети ServNet. Она использует собственную подсистему электропитания и сетевую инфраструктуру для передачи данных.

4.8. Интеллектуальная собственность Союзного государства и перспективы массового производства

В реализации суперЭВМ Ряда 4 семейства «СКИФ» впервые интеллектуальная собственность на изделия принадлежит Союзному государству. В частности, в нашем распоряжении находится полный комплект конструкторской и производственной документации. Это дает право и возможность разместить изготовление всех блоков и узлов на любых предприятиях, в том числе и отечественных. А также возможность вносить изменения в конструкторскую документацию, создавать новые модификации суперЭВМ Ряда 4 семейства «СКИФ», в том числе на различной микропроцессорной базе (включая отечественную, если такая будет доступна).

Тем самым мы будем в максимальной готовности к восприятию отечественной элементной базы по мере ее появления.

Заключение

В рамках программы «СКИФ-ГРИД» обеспечивается разработка конструкторской документации и выпуск опытных образцов вычислительных узлов, шасси, шкафов суперЭВМ Ряда 4. В настоящий момент завершена разработка эскизной конструкторской документации суперЭВМ Ряда 4 семейства «СКИФ». Ведется выпуск опытных образцов вычислительных узлов (февраль 2009), шасси (март–апрель 2009) и шкафов (апрель–май 2009) этих суперкомпьютеров. В мае 2009 года организуется серийный выпуск и поставка потребителям изделий последовательности «СКИФ 4/Н». Суперкомпьютеры семейства «СКИФ» Ряда 4 при организации их массового производства становятся основой оснащения отечественной суперкомпьютерной техникой учреждений образования и науки, исследовательских и конструкторских бюро, предприятий промышленности и государственных структур Союзного государства.

Список литературы

[1] Абрамов С. М. *Итоги суперкомпьютерной программы «СКИФ» Союзного государства и перспективы ее развития* // «Пути ученого. Е. П. Велихов» ред. Академик РАН В. П. Смирнов. — М.: РНЦ «Курчатовский институт», 2007. — ISBN 978–5–9900996–1–6, с. 325–333. ↑1.1

[2] Абрамов С. М., Заднепровский В. Ф., Московский А. А. *Отечественные суперЭВМ и грид-системы. Проблемы развития национальной киберинфраструктуры в России* // «Российские суперкомпьютеры: Наука. Технологии. Производство»: Сборник статей. — Т. **2**. — М.: Библиотека ЦСПП, 2008. — ISBN 5–8027–0061–0, с. 36–54. ↑2, 2.1, 3

[3] Абрамов С. М., Заднепровский В. Ф., Московский А. А. *Опыт использования СуперЭВМ для эффективного развития «прорывных технологий» (на примере нанотехнологий)* // XII научно-практическая конференция Университета города Переславля «Программные системы: теория и приложения». — Т. **1**. — Переславль-Залесский: Изд-во «Университет города Переславля», 2008. — ISBN 978–5–901795–11–8, с. 37–50. ↑2

[4] Абрамов С. М., Анищенко В. В., Заднепровский В. Ф., Московский А. А., Криштофик А. М., Опанасенко В. Ю., Парамонов Н. Н. *Развитие семейства отечественных суперкомпьютеров «СКИФ» в рамках программы Союзного государства «СКИФ-ГРИД»* // Научный сервис в сети Интернет: решение больших задач. Труды Всероссийской научной конференции. 22–27 сентября 2008 г. г. Новороссийск. — М.: Изд-во МГУ имени М. В. Ломоносова, 2008. — ISBN (CD)978–5–211–05616–9, с. 286–291. ↑2.1, 2.4

Исследовательский центр мультипроцессорных систем ИПС имени А. К. Айламазяна РАН

ИПС имени А. К. Айламазяна РАН

S. M. Abramov, V. F. Zadneprovskiy, A. A. Moskovskiy, A. B. Shmelev. *Supercomputers SKIF series 4* // Proceedings of Program Systems institute scientific conference "Program systems: Theory and applications". — Pereslavl-Zalesskij, v. **1**, 2009. — p. 193–216. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. The paper outlines "SKIF" series 4 supercomputers, which feature many technology advances: ultra-dense packaging of computational power, liquid cooling on node level, new solutions for system, supplementary and service networks. For the first time in "SKIF" projects, Russian organizations will have intellectual property rights for all components down to, but excluding semiconductor chips. Planned achievable maximum peak performance for "SKIF" series 4 is 500 TFlops by fall 2009, 1 PFLops by fall 2010 and more than 5 PFlops by spring 2012.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 519.68

Е. С. Афонькина, М. А. Гришина, Н. Н. Ившина,
Г. А. Матвеев, В. А. Потемкин

Реализация параллельной версии программы расчёта лекарственных средств с использованием Т-Системы с открытой архитектурой (OpenTS)

Аннотация. В статье рассматривается реализация параллельной версии программы расчёта лекарственных средств (ПС BiS) с использованием Т-Системы с открытой архитектурой (OpenTS). Приведены результаты тестирования исходной (однопроцессорной) и параллельной (многопроцессорной) версий программы.

Ключевые слова и фразы: Т-Система, OpenTS, вычислительный кластер, язык параллельного программирования, лекарственное средство, биологическая активность соединений.

Введение

В статье используется следующий перечень сокращений и специальных терминов:

- OpenTS — Т-Система с открытой архитектурой;
- Т++ — язык параллельного программирования для Т-Системы, являющийся синтаксически и семантически гладким расширением C/C++ при помощи нескольких ключевых слов;
- Т-функция — функция, реализованная на языке Т++;
- ПС BiS — программная система BiS анализа биологической активности соединений и прогноза биологической активности новых потенциальных лекарственных средств;

Работа выполнена при поддержке научно-технической Программы Союзного государства «Разработка и использование программно-аппаратных средств Грид-технологий перспективных высокопроизводительных (суперкомпьютерных) вычислительных систем семейства «СКИФ» (шифр «СКИФ-ГРИД»), РФФИ (гранты РФФИ: № 07–03–96041, № 07–04–96053), Human Capital Foundation (London, Great Britain).

- рецептор — чувствительное нервное окончание или специализированная клетка, преобразующая воспринимаемое раздражение в нервные импульсы;
- лиганды — в комплексных соединениях — молекулы или ионы, непосредственно связанные с центральным атомом;
- конформация молекул — это геометрические формы, которые могут принимать молекулы органических соединений при вращении атомов или групп атомов;
- супрамолекулярная химия — это раздел химии, изучающий структуру и функции ассоциаций двух или более химических частиц, удерживаемых вместе межмолекулярными силами.

Системы параллельного программирования имеют важное прикладное значение. Это связано с быстрым развитием компьютерной техники, с переходом на многоядерные архитектуры, с развитием кластерных и распределённых систем. Для реализации параллельной версии программы расчёта лекарственных средств была выбрана система программирования OpenTS [1, 2], обладающая следующими важными характеристиками:

- совместимость со стандартными платформами и программными средствами;
- универсальность — система OpenTS применима в широком классе прикладных областей;
- масштабируемость в широком диапазоне числа процессоров;
- позволяет создавать эффективные приложения для компьютерных систем с высокой степенью параллелизма.

1. Описание алгоритма BiS

Программная система BiS позволяет определить строение комплексов «модельный рецептор-лиганд», установить взаимосвязь биологической активности с параметрами строения таких комплексов, выявить фармакофорные фрагменты молекулы. Последовательная версия программы реализована на языке Fortran.

Блок-схему алгоритма можно представить в виде следующих шагов:

(1) ввод исходных данных: выборка соединений, координат атомов и величин биологических активностей соединений;
(2) создание псевдоатомного окружения;

(3) оптимизация ориентации молекул в псевдоатомном поле;
(4) уточнение потенциалов поля (зарядов и радиусов псевдоатомов);
(5) вывод результатов: молекулы, ориентированные в псевдоатомном рецепторе, энергии и силы взаимодействия в модельном комплексе «рецептор-лиганд», зависимость биологической активности от характеристик взаимодействия в системе «рецептор-лиганд».

На первом шаге строится псевдо-атомное окружение, комплементарное к наиболее активной молекуле выборки. В каждой точке поверхности молекулы можно определить потенциал кулоновских и ван-дер-ваальсовых взаимодействий и смоделировать комплементарное поле. Потенциал кулоновских взаимодействий в m-ой точке поверхности определяется по известной формуле:

$$\varphi_m^q = \sum_{i=1}^{N} \frac{q_i}{R_{im}} k,$$

где N — число атомов рассматриваемой структуры, q_i — заряд атома i, R_{im} — расстояние от точки m до атома i, k — коэффициент пересчёта в единицы СИ.

Для расчета потенциала ван-дер-ваальсовых взаимодействий, наводимых на точку m, по аналогии с кулоновским предложено использование следующей формулы:

$$\varphi_m^{VDW} = -2 \sum_{i=1}^{N} V_{im} \frac{2^3 r_i^3}{R_{im}^6},$$

где V_{im} — глубина минимума потенциальной энергии в соответствии с потенциалом Леннард-Джонса, r_i — ван-дер-ваальсовый радиус атома i. Расчет величин V_{im} и r_i может быть произведен в рамках модели MERA [3]. В данном уравнении для упрощения расчетов использован только член притяжения в ван-дер-ваальсовых взаимодействиях.

Комплементарное поле строится в виде псевдоатомного окружения молекулы. В точках поверхности строятся пробные сферы (псевдоатомы) с зарядами и радиусами, комплементарными полю в данной точке. Заряд и радиус пробной сферы вычисляются по формулам

$$q_m = -\frac{\varphi_m^q}{\sum_{i=1}^{N} \frac{k}{R_{im}}}, \quad r_m = \sqrt[3]{\frac{\varphi_m^{VDW}}{-2^3 \sum_{i=1}^{N} \frac{2 V_{im} r_i^3}{R_{im}^6}}}.$$

На следующем шаге производится оптимизация ориентации второй молекулы в поле, комплементарном первой молекуле, с использованием комбинации симплексного и квази-ньютоновского методов до достижения минимума совокупной вероятности контакта (P) её атомов со всеми точками модельного рецептора.

$$P = 1 - \Pi_{m=1}^{M}(1 - p_m),$$

где

$$p_m = \exp\left(-\frac{E_m}{RT}\right),$$

$$E_m = \sum_{i=1}^{N}\left(\frac{kq_iq_m}{R_{im}} - 2V_{im}\frac{(r_m + r_i)^6}{R_{im}^6} + V_{im}\frac{(r_m + r_i)^{12}}{R_{im}^{12}}\right),$$

где N — число атомов во второй молекуле.

При этом вторая молекула может иметь размеры, значительно отличающиеся от размеров первой молекулы. Расположение пробных сфер на поверхности второй молекулы позволяет учесть конформационную подстройку рецептора под размер встраиваемой молекулы. В найденной ориентации второй молекулы уточняется комплементарное поле рецептора путём добавки потенциалов поля второй молекулы к имеющимся:

$$\varphi_m^q = \varphi_m^q + \varphi_m^q\prime; \quad \varphi_m^{VDW} = \varphi_m^{VDW} + \varphi_m^{VDW}\prime.$$

Потенциалы поля второй молекулы $\varphi_m^q\prime$ и $\varphi_m^{VDW}\prime$ вычисляются аналогично потенциалам первой молекулы. Аналогичная процедура осуществляется для третьей, четвертой и всех последующих молекул выборки. По окончании процедуры производится рассмотрение расположения первой, второй и т.д. молекул выборки в уточненном комплементарном поле. Процедура прекращается, если различие потенциала поля на новом и предыдущем шаге не превышает заданного предела. После этого строится линейная зависимость между величиной биологической активности соединения и параметрами взаимодействий псевдо-атомов модельного рецептора с молекулой. В качестве параметров взаимодействий могут быть использованы энергии взаимодействия и силы F_m:

$$F_m = \sum_{i=1}^{N}\frac{dE}{dR_m}.$$

Полученное уравнение позволяет выделить те псевдо-атомы модельного рецептора, взаимодействие с которыми определяет величину

биологической активности соединений. Данные псевдо-атомы моделируют активные центры рецептора. Фрагменты лекарственного средства, расположенные вблизи активных центров, определяют фармакофорную часть молекулы.

На основе предложенного алгоритма не сложно предложить новый подход к дизайну перспективных лекарственных средств. В рамках данного подхода производится комбинаторная замена атомов или фрагментов на другие атомы или фрагменты у наиболее активной молекулы с целью получения более активного соединения. Вновь полученное соединение помещается в модельный рецептор. На основе характеристик взаимодействия соединения с модельным рецептором вычисляется величина биологической активности. Если полученное значение находится на достаточно высоком уровне — перспективное соединение может быть синтезировано. Данный подход к дизайну перспективных лекарственных средств реализован в рамках алгоритма BiS.

Таким образом, алгоритм BiS позволяет определить строение комплексов «модельный рецептор-лиганд», установить взаимосвязь биологической активности с параметрами строения таких комплексов, выявить фармакофорные фрагменты молекулы.

2. Параллельная версия ПС BiS. Результаты тестирования

Для моделирования супрамолекулярных комплексов «рецептор-лекарственное средство» создан параллельный прототип ПС BiS с использованием OpenTS.

Алгоритм работы версии программы на T++ выглядит так: основная подпрограмма осуществляет поворот молекулы лекарственного средства в полярных координатах. При каждом изменении угла производится вызов T-функции, которая строит для молекулы модельный рецептор и вычисляет энергию взаимодействия в системе «модельный рецептор-лекарственное средство». Эта энергия записывается в выходные данные T-функции. Таким образом, осуществляется поиск наиболее выгодной (низкой по энергии) ориентации молекулы в модельном рецепторе.

Программа тестировалась на кластере «СКИФ МГУ» (T60), имеющем следующую конфигурацию:

- операционная система: AltLinux HPC, x86_64;
- версия ядра ОС: 2.6.18;

Таблица 1. Результаты тестирования ПС BiS на кластере «СКИФ МГУ»

Количество процессоров	Время выполнения последовательной версии, сек	Время выполнения параллельной версии, сек	Ускорение
1	982,061	982,061	1,000
2		982,046	1,000
4		487,481	2,015
8		240,244	4,088
12		169,255	5,802
16		116,524	8,428
20		98,675	9,952
24		81,108	12,108
28		64,042	15,335
32		63,582	15,446
37		46,909	20,935
111		21,211	46,300

- число вычислительных узлов: 625;
- число процессоров: 1250;
- число ядер: 5000;
- модель процессора: Intel Xeon E5472 3.0 ГГц;
- число процессоров на узле: 2;
- число ядер на процессоре: 4;
- системная сеть: InfiniBand DDR.

Тестирование производилось на кластере с использованием от 1 до 111 процессоров. Результаты тестирования представлены в таблице 1.

Результаты расчётов параллельного прототипа ПС BiS совпадают с результатами последовательного аналога. Результаты моделирования супрамолекулярных комплексов «рецептор-лекарственное средство» в рамках алгоритма BiS сопоставлены с данными рентгеноструктурного анализа (РСА).

Расчёты проведены для агонистов $5HT_{1A}$-рецептора (транквилизаторные средства) и ингибиторов дигидрофолатредуктазы (противоопухолевые и туберкулостатические средства).

Выводы

Исследования показали следующее:

- параллельная реализация программы с помощью T-системы с открытой архитектурой (OpenTS) компактна и проста;
- потребовались лишь незначительные изменения исходного кода, чтобы сделать версию программы на языке T++;
- получено хорошее ускорение работы параллельной версии программы относительно её последовательного аналога.

Список литературы

[1] Абрамов С. М., Адамович А. И., Инюхин А. В., Роганов В. А., Шевчук Ю. В., Шевчук Е. В. *T-система с открытой архитектурой* // Суперкомпьютерные системы и их применение SSA'2004: Труды Международной научной конференции, 26–28 октября 2004 г. Минск, ОИПИ НАН Беларуси. — Минск, 2004, с. 18–22. ↑(document)

[2] Abramov S., Adamovich A., Inyukhin A., Roganov V., Shevchuk E., Shevchuk Yu., Vodomerov A. *OpenTS: An Outline of Dynamic Parallelization Approach* // PACT'2005, LNCS, 2005. ↑(document)

[3] Потемкин В. А., Барташевич Е. В., Белик А. В. *Модель расчета атомных объемных характеристик в молекулярных системах* // Журнал физической химии. — **т.72**, № 4, 1998, с. 650–656. ↑1

Челябинский Государственный Университет

Институт Органического Синтеза им. И.Я. Постовского УрО РАН

Исследовательский центр мультипроцессорных систем ИПС РАН

E. S. Afonkina, M. A. Grishina, N. N. Ivshina, G. A. Matveev, V. A. Potemkin. *Development and implementation of a parallel version of the algorithm «Biological Substrate Search» (BiS) using the T-System with the open architecture (OpenTS)* // Proceedings of Program Systems institute scientific conference "Program systems: Theory and applications". — Pereslavl-Zalesskij, v. **1**, 2009. — p. 217–223. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. The article considers development and implementation of a parallel version of the algorithm «Biological Substrate Search» (BiS) using the T-System with the open architecture (OpenTS). Tests results of a original (uniprocessor) and parallel (multiprocessor) versions of the algorithm showed.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 519.687.4

Г. И. Есин, А. А. Кузнецов, В. А. Роганов

Экспериментальная реализация отказоустойчивой системы распределенных вычислений "SkyTS" для параллельного счета ресурсоемких T++ приложений в гетерогенной распределенной вычислительной среде

Аннотация. В данной работе рассмотрен подход к реализации распределенной вычислительной системы, предназначенной для счета ресурсоемких T++ приложений. Дано описание экспериментальной реализации этого подхода в виде системы „SkyTS“, которая обладает свойствами отказоустойчивости, масштабируемости и поддержкой неоднородных вычислительных сетей.

1. Введение

OpenTS — система параллельного программирования, разработанная в ИПС РАН в 2000-2004 годах в рамках суперкомпьютерного проекта „СКИФ“ Союзного государства России и Беларуси. После успешного завершения программы „СКИФ“ разработка OpenTS была продолжена в рамках программы „СКИФ-ГРИД“ и различных академических проектов.

Система OpenTS (Open T-System) [1, 2] представляет собой современную и наиболее удачную реализацию концепции T-системы — программной среды параллельного программирования с поддержкой автоматического динамического распараллеливания приложений, которая сочетает в себе функциональную и императивную парадигмы программирования. OpenTS — это среда поддержки исполнения приложений, написанных на языке T++. Данный язык программирования является параллельным диалектом Си++ и расширяет исходный язык семью новыми ключевыми словами. В остальных известных нам системах программирования, в которых взяты за основу языки Си/Си++, набор новых ключевых слов и команд значительно шире, что осложняет процесс изучения языка и создания программ. Среда поддержки исполнения T++ приложений (T-приложений), берет на себя основную часть работы по организации параллельного

счета (синхронизация, распределение нагрузки, транспортировка сообщений). Тем самым, система OpenTS позволяет снизить затраты на разработку параллельных программ, увеличить глубину параллелизма и более полно использовать возможности аппаратной части мультипроцессора за счет распараллеливания в динамике.

Изначально система OpenTS разрабатывалась в среде Linux для функционирования на Linux-кластерах и рабочих станциях. В ходе развития системы ее исходный код и подсистема сборки были портированы на ОС Windows [3], что позволило расширить ее область применения на кластерные установки под управлением инфраструктуры „Windows HPCS 2008“ фирмы Microsoft.

Концепция T-системы хорошо подходит для Grid-компьютинга. Вычислительные задания (T-задачи), которые порождает пользовательское T-приложение, могут быть вычислены не только на любом узле сильносвязанного кластера, но и в принципе на любом компьютере в сети Интернет. Поскольку T-задачи имеют свой „вес“, а для узлов распределенной системы вводится понятие надежности, то можно организовать работу T-приложения так, что наиболее весомые (значимые) T-задачи верхнего уровня распределяются планировщиком по наиболее надежным счетным узлам вычислительной системы. Соответственно, менее значимые задачи могут быть назначены на менее надежные листовые узлы в общей иерархии узлов системы. Поэтому в перспективе систему OpenTS можно использовать не только для кластеров, но и для территориально-распределенных неоднородных установок.

Предпосылками для развития данного подхода явились проведенные в рамках программы „СКИФ“ эксперименты по использованию совместно с системой OpenTS следующих реализаций технологии MPI в качестве транспортного уровня: MPICH-G2, IMPI, PACX-MPI. Эти пробные испытания показали, что технология OpenTS может быть доведена до полноценного Grid-решения. Под Grid-технологиями мы будем понимать набор средств и технических решений, которые позволяют объединять разрозненные разнородные компьютеры и суперкомпьютеры в территориально-распределенную гетерогенную информационную и вычислительную систему [4]. При этом основной задачей будет интеграция всех вычислительных ресурсов всех компьютеров, входящих в систему для решения тяжелых и сверхтяжелых научно-прикладных задач.

Все проведенные в последние годы доработки системы OpenTS (улучшение свойств кроссплатформенности, разработка подсистем отказоустойчивости и масштабируемости) позволили системе перейти на новый уровень развития, который состоит в поддержке выполнения параллельных T-приложений в неоднородной распределенной вычислительной среде. В рамках программ Союзного государства „Триада“ и „СКИФ-ГРИД“ разработана архитектура и проведена экспериментальная реализация распределенной отказоустойчивой системы „SkyTS“ (прежнее название „SkylighTS“), которая позволяет объединить разнородные ресурсы компьютеров в сети Интернет для счета T-приложений, решающих ресурсоемкие научно-прикладные задачи.

2. Отказоустойчивость работы T-приложений

Отказы оборудования — это одна из наиболее частых причин остановки счета в супервычислительных установках. В случае распределенных Grid-систем отказы оборудования будут происходить неизбежно и довольно часто, по сравнению с кластерными системами, поскольку узлы такой системы скорее всего будут находиться вне контроля головной организации Grid-системы. Для реализации такой распределенной системы на базе OpenTS необходимо наделить среду поддержки исполнения T-приложений возможностью восстанавливать процесс счета после аварийного сбоя (или даже штатного выключения) какого-либо из счетных узлов.

Были разработаны различные инструменты для имплементации подсистемы отказоустойчивости работы T-приложений [5, 6]:

- Улучшенная для работы по TCP/IP реализация наиболее употребимого подмножества функций MPI — DMPI, дополненная функциями автоматического мониторинга исправности вычислительной конфигурации сильно- либо слабосвязанного множества вычислительных узлов. Такая реализация позволяет в каждом конкретном случае наиболее эффективно реализовать восстановление нормального счета T-приложения путем переповтора пострадавшего фрагмента вычислений в случае аппаратного сбоя на узле или потери связи с ним.

- Доработанная реализация среды поддержки исполнения T-приложений (микроядро OpenTS). Эта среда опирается на отказоустойчивую реализацию DMPI и автоматически обеспечивает отказоустойчивость T-приложений, реализованных в функциональном стиле на языке T++, которые могут работать как в сильно-, так и в слабосвязанной вычислительной среде.

В системе OpenTS использованы следующие оригинальные методы обеспечения отказоустойчивости:

- Используемая системой OpenTS коммуникационная библиотека DMPI способна сигнализировать о сбоях, продолжая нормально функционировать и отслеживать, какие сообщения следует перепослать, а какие нет. Доработка DMPI состоит в том, что попытка запуска приложения на каждом узле производится не однократно, а многократно, то есть в случае сбоя и перезапуска узла возникает новая „реинкарнация“ вычислительного процесса, который готов продолжить работу. Также происходит корректная обработка возможных ошибок уровня сокетов TCP/IP и передаче статуса этих ошибок в функцию-обработчик сбоев.
- Реализовано сохранение пренатальных процессов (вызванных T-функций), так как они еще не получили стэка для своей работы и находятся в наиболее компактной (и даже адресно-независимой) форме. Реализовано запоминание T-функций и их аргументов, а также назначенных для их исполнения вычислительных узлов с целью обеспечения возможности их повторного вычисления на исправных узлах в случае аварии. Повторный запуск T-функций производится после реконфигурации коммуникационной подсистемы и Суперпамяти.

Безусловно, использование TCP/IP в качестве базового уровня для реализации MPI-взаимодействия может внести некоторые накладные расходы при использовании внутри тесносвязанных кластеров. Однако важнее уверенность в безукоризненной отказоустойчивости базового уровня коммуникаций. Кроме того, уже начиная с небольших метакластерных установок, накладные расходы, вносимые уровнем TCP/IP, не будут такими уж существенными хотя бы

потому, что сами кластерные установки обычно связаны именно по этому протоколу.

Кроме запуска командой mpirun/mpiexec, возможно динамическое вхождение вычислительного узла в расчетное поле по собственной инициативе. При этом, вычислительный узел может не обладать собственным IP-адресом, так как соединение с коммуникационным сервером происходит по инициативе самого узла. В этой схеме головной процесс работает дополнительно в режиме сервера, принимая запросы на соединение с узлов. После успешного установления такого соединения каждый подключенный по собственной инициативе узел взаимодействует со всеми остальными узлами по протоколу TCP/IP, обмениваясь активными сообщениями уровня OpenTS. Новая схема интегрирована с традиционной формой запуска, которая хорошо подходит для находящихся в распоряжении относительно высоконадежных узлов кластера.

3. Масштабируемость T-приложений на большое количество узлов

При разработке распределенной вычислительной системы не должно ставиться каких-либо ограничений на количество задействованных в счете узлов. В перспективе это число может достигать десятков и сотен тысяч. Поэтому в основе Grid-системы должна лежать технология, обеспечивающая эффективное масштабирование приложений, выполняющихся распределенно на большом число узлов.

Для системы OpenTS реализован набор усовершенствований, позволяющий обеспечить эффективное распараллеливание ресурсоемких T-приложений на сверхбольших кластерных и метакластерных установках (насчитывающих до миллиона вычислительных ядер), и в крупных Grid-системах. Изменения затронули подсистему Суперпамяти, встроенный планировщик и обмен информацией о ресурсах.

3.1. Подсистема Суперпамяти OpenTS

Подсистема Суперпамяти является ключевым элементом реализации системы OpenTS, поэтому масштабирование структур данных и алгоритмов работы Суперпамяти явилось ключевым этапом доработки.

Для реализации Суперпамяти в OpenTS используется модель общей памяти. В модели программирования с общей памятью все процессы совместно используют общее адресное пространство, к которому они асинхронно обращаются с запросами на чтение и запись. В таких моделях для управления доступом к общей памяти используются всевозможные механизмы синхронизации типа семафоров и блокировок процессов. Преимущество этой модели с точки зрения программирования состоит в том, что понятие собственности данных (монопольного владения данными) отсутствует, следовательно, не нужно явно задавать обмен данными между производителями и потребителями. При использовании системы OpenTS программист освобожден от необходимости явно специфицировать общие данные и упорядочивать доступ к ним с помощью средств синхронизации, поскольку все эти функции автоматически выполняет система.

Как известно, существуют различные схемы организации общей памяти в распределенных системах. Некоторые из них напрямую отображают виртуальное адресное пространство на области памяти локальных узлов. Наряду с простотой, такие схемы обладают определенными недостатками. Прежде всего, единицей работы с памятью в такой схеме является аппаратная страница, что замедляет работу с совокупностями небольших по размеру объектов. На 32-разрядной архитектуре пределом совокупного объема данных оказывается 4 Гб, что по современным меркам выглядит достаточно серьезным ограничением, особенно в суперкомпьютерных применениях. Этих недостатков лишены схемы так называемой объектно-ориентированной общей памяти, где единицей адресации является не аппаратная страница, а объект (ячейка). Перекладывая часть работы на программное обеспечение, удается достичь снятия указанных ограничений. Дополнительно, такая схема организации памяти может быть легко интегрирована с различными аспектами объектно-ориентированных технологий, такими как автоматическая сборка мусора.

В OpenTS Суперпамять организована в виде матрицы; при этом общий размер матрицы равен *N*M*sizeof(TCell)*, где N и M — максимальное количество узлов установки и максимальное количество супер-ячеек на каждом узле соответственно.

В случае использования ОС Linux Суперпамять размещается в виртуальном адресном пространстве. Это позволяет зарезервировать большой объем виртуального адресного пространства, расходуя физическую память по мере необходимости. Однако в случае других ОС,

а также и под ОС Linux в некоторых случаях разумнее не выходить в захвате виртуального адресного пространства за некие разумные рамки. А поскольку в случае больших установок произведение N*M может быть очень большим, то данный подход неприемлем без доработок. В случае использования 32-разрядных ОС ограничений, накладываемых системой, становится еще больше.

Путь решения этого вопроса классический; он используется во многих аппаратных MMU (Memory Management Unit) и состоит в том, что делается несколько уровней/директорий. Нижний уровень содержит непосредственно супер-ячейки, предыдущие содержат массивы ссылок на супер-ячейки и так далее. Фактически это означает, что вместо сплошной суперматрицы (матрицы ячеек суперпамяти в OpenTS) хранится разреженная суперматрица.

3.2. Требования к ресурсам

Хорошая масштабируемость предполагает не только эффективность работы на больших установках, но и экономное (по потребности) расходование системных ресурсов. Этот момент очень важен для разработчиков T-приложений, поскольку при отладке и тюнинге T-приложения часто запускают на небольших установках или просто на персональном компьютере в так называемом „режиме эмуляции большого кластера“. При этом запускается значительно больше системных процессов, чем имеется вычислительных ядер.

Новая схема реализации суперсегментов в совокупности с оптимизацией некоторых системных констант позволили снизить минимальные требования к объему необходимой T-приложению памяти приблизительно в восемь раз.

Измерения проводились под ОС Linux Fedora 9 x86_64 на двухъядерном ПК с 2-мя гигабайтами оперативной памяти. До доработки удавалось имитировать кластер с 8-ю узлами, после доработки — с 64-мя.

3.3. Оптимизация обменов информацией о ресурсах

Семантика Суперпамяти в OpenTS различна для разных сегментов, но в первой ее реализации каждый узел обменивается с каждым информацией о наличии у него подзадач и свободных мощностей. Разумеется, это эффективно либо на узком классе задач, либо для кластеров небольшой размерности. В случае общей задачи на большой установке количество обменов может вырасти квадратично.

По этой причине обмены информацией о ресурсах целесообразно также сделать многоуровневыми: то есть поступать так же, как поступают при реализации функции *broadcast()*: вместо рассылки информации всем узлам сразу посылать нескольким (скажем, четырем) с просьбой распространить эту информацию далее.

Основная идея оптимизации этих обменов состоит в том, что OpenTS не совершает обращений к ячейкам суперпамяти, которые лежат вне пределов некоторого графа, как раз соответствующего иерархии узлов, используемой доработанным планировщиком (см. ниже). При этом „лишним“ обменам в большинстве случаев попросту неоткуда взяться. Тем не менее, они не запрещены, поэтому приложение не ограничено в своей свободе устраивать и свои собственные схемы обменов данными.

Отдельно изменяется схема обмена информацией о вычислительной мощности каждого вычислительного узла в самом начале счета.

3.4. Доработка планировщика

Новый планировщик, который распределяет подзадачи в больших системах, имеет иерархическую структуру. Каждый вычислительный узел принадлежит тому или иному уровню иерархии. На практике иерархия может возникать естественным путем (например, метакластер, состоящий из нескольких кластеров), но в некоторых случаях дополнительные уровни целесообразно создавать искусственно (например, когда в кластере нескольких сотен узлов).

В текущей экспериментальной версии используется синтетическая иерархия, построенная на основе одинаковых по размеру гиперкубов, соединенных в некоторых местах в древоподобную структуру.

При поиске задач каждый свободный от работы узел обращается к вышестоящим, публикуя свои свободные ресурсы. При поиске помощников для вычисления подзадач каждый перегруженный (имеющий более одной задачи) узел обращается к свободным нижестоящим. Кроме этого, узлы-члены каждого уровня иерархии связаны в гиперкуб и также обмениваются по горизонтали. На основе полученной информации происходит обмен задачами.

Роль „досок объявлений“ как раз играют суперсегменты с публикуемыми задачами и свободными ресурсами узлов. Собственно, изначально OpenTS и была ориентирована на использование иерархии „досок объявлений“, через которые бы шел обмен задачами. Однако реально использовалась лишь одна доска объявлений на весь кластер.

4. SkyTS — Grid-система для счета T++ приложений

С учетом всех выполненных доработок среды поддержки исполнения T-приложений стало возможным разработать систему распределенных вычислений для счета T-приложений. На основе разработанных ранее механизмов запуска T-приложений в отказоустойчивом режиме, а также подсистемы масштабируемости, была разработана экспериментальная версия Grid-системы „SkyTS“, которая позволяет осуществить интеграцию вычислительных ресурсов разрозненных и разнородных компьютеров в территориально-распределенную гетерогенную информационно-вычислительную систему. Она обладает следующими свойствами:

- Высокая масштабируемость: число параллельных процессов T-приложения может достигать сотен тысяч; также возможно подключение к системе любого количества серверных (управляющих) и рабочих (вычислительных) узлов.
- Отказоустойчивость: в системе задействованы механизмы отказоустойчивой работы T-приложений, что позволяет системе функционировать в территориально-распределенной среде.
- Кроссплатформенность: серверные и счетные модули реализованы на интерпретируемом языке программирования, что делает их код переносимым.
- Поддержка эмуляции: T-приложение, созданное для какой-либо ОС, способно работать на других ОС за счет использования инструмента эмуляции Wine.
- Поддержка виртуализации: серверные и счетные модули способны работать в среде виртуальной машины в гостевой ОС Linux.

Система состоит из следующих компонентов:

- Web-интерфейс пользователя, который служит для получения вычислительных задач от пользователей и обратной связи с ними;
- база данных для хранения информации о заданиях, а также репозитории для хранения исполняемых файлов заданий;
- серверный (управляющий) модуль, который распределяет T-задания между счетными модулями;
- счетный модуль, который осуществляет запуск заданий на счет на одиночных компьютерах в сети Интернет.

4.1. Web-интерфейс пользователя системы SkyTS

Для управления подачей и запуском заданий служит Web-интерфейс пользователя системы SkyTS. Интерфейс системы является важной компонентой, так как обеспечивает обратную связь между конечным пользователем и самой системой. После авторизации через Web-интерфейс доступны следующие операции:

- загрузка исполняемого кода T-приложений;
- параметризация T-приложений входными данными;
- постановка заданий на счет;
- доставка результатов счета;
- управление пользователями и их правами.

Эта компонента SkyTS представляет собой Web-приложение, и работа с ним производится посредством использования штатного браузера. Такой подход был выбран потому, что Web-приложения не имеют ограничений на использование какой-либо операционной системы, соответственно они являются кросс-платформенными, что делает их универсальным и удобным инструментом для работы в любых условиях.

4.2. База данных системы

Центральным элементом системы SkyTS является база данных. Она создана на основе СУБД MySQL версии 5.0. В ней хранятся все сведения о сущностях системы: приложения, зарегистрированные пользователи, серверы, и т.д. В качестве хранилищ загруженных приложений используются SVN-репозитории. Данный подход является стандартным при проектировании эффективных информационных систем, поскольку в этом случае происходит разделение информационной части системы (база данных) от файловой, управление которой поручается файловому серверу. Это ведет к уменьшению нагрузки как на СУБД, так и на вычислительные узлы и, соответственно, повышению быстродействия системы в целом.

4.3. Репозитории приложений

Данный компонент необходим для хранения загруженных приложений. В данный момент используется репозиторий, основанный на системе Subversion (SVN). Репозиториев может быть несколько. Это необходимо для распределения нагрузки на сетевые каналы и ускорения доставки приложений к вычислителям. Соответственно,

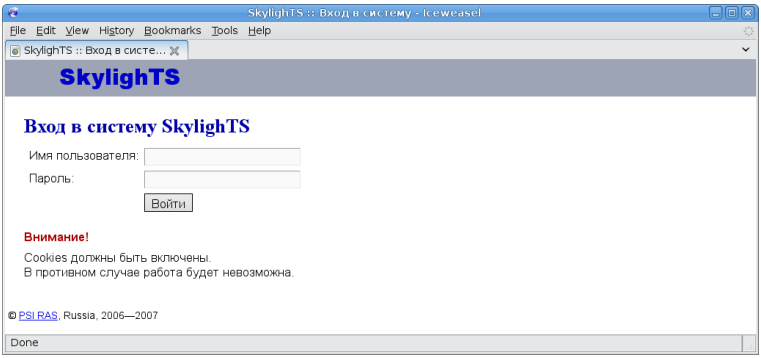


Рис. 1. Форма авторизации

производится синхронизация хранилищ с некоторой периодичностью, которая задается администратором системы SkyTS.

4.4. Роли пользователей системы

В целях обеспечения безопасности, в системе SkyTS существуют 3 роли пользователей:

(1) Администраторы; пользователи с такой ролью имеют доступ ко всем функциям интерфейса и могут просматривать любую информацию, содержащуюся в базе данных.
(2) Модераторы; пользователи с такой ролью имеют ограничение на использование функций интерфейса и доступ к информации.
(3) Непривилегированные (обычные) пользователи.

4.5. Авторизация

Авторизация производится с помощью Web-формы, содержащей 2 атрибута: имя пользователя и его пароль (см. рис. 1).

4.6. Управление приложениями

4.6.1. *Просмотр доступных приложений*

После авторизации, отображается форма, содержащая список доступных приложений (см. рис. 2). Среди них могут быть такие, о которых можно только просмотреть сведения, а есть такие, которые

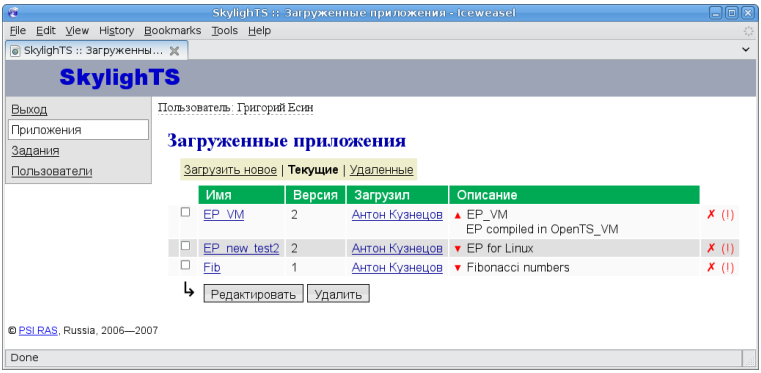


Рис. 2. Список доступных приложений

можно редактировать и удалять. Доступность этих функций определяется ролью пользователя и его участием в разработке приложения.

Также справа от таблицы указаны один или более флагов, сообщающих является ли программа проверенной, находится ли она в данный момент в очереди и если да, то какой у нее статус счета.

4.6.2. *Загрузка нового приложения*

Загрузка нового приложения производится из Web-формы (см. рис. 3). В ней содержатся поля для составления описания приложения. Загружаемое приложение оформляется в виде файлового архива. После загрузки приложения его архив распаковывается и сохраняется в SVN-репозитории. Сведения о приложении сохраняются в базу данных, в числе прочего отмечается кто загрузил и указывается список разработчиков приложения. Этот список необходим как для общего информирования, так и для того, чтобы обозначить, кто может изменять сведения о данном приложении или обновлять его файлы в репозитории.

4.6.3. *Просмотр и редактирование информации о приложении*

На странице просмотра информации о приложении доступны все сведения о нем, включая имя, версию, дату загрузки, имя разработчика и описания. Также на ней расположены инструменты управления приложением, которые позволяют отредактировать приложение,

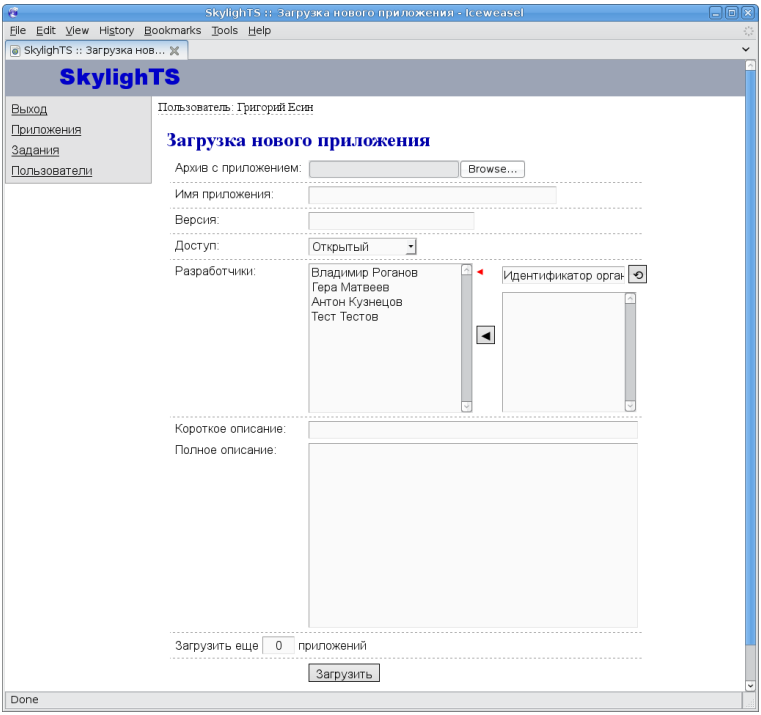


Рис. 3. Форма загрузки нового приложения

удалить его из системы или запустить на счет. Web-форма для редактирования приложения аналогична Web-форме для его загрузки (см. рис. 3) .

При удалении приложения в системе SkyTS данное приложение просто помечается как неактивное. Администратор и модератор могут изменить статус приложения с неактивного на активный и таким образом восстановить удаленное приложение.

Редактировать и удалять приложения могут только следующие категории пользователей системы:

- администраторы системы SkyTS;
- модераторы, принадлежащие к той же организации, что и владельцы приложения;
- пользователь, загрузивший приложение;

- пользователи, отмеченные как разработчики данного приложения.

4.7. Управление очередью задач

Управление очередью задач - одна из основных функций Web-интерфейса. Очередь задач была введена с целью распределения вычислительной нагрузки по сегментам вычислительной сети. Web-интерфейс системы позволяет производить следующие функции по управлению заданиями:

- создание нового задания;
- изменение его параметров (аргументов командной строки);
- запуск задания;
- приостановка задания;
- удаление из очереди;
- просмотр статуса;
- получение результатов.

4.7.1. *Постановка задачи в очередь*

При постановке задачи в очередь пользователь выбирает те приложения, которые он собирается запустить, указывает параметры командной строки, ссылки на файлы-данные и/или загружает файлы-данные, а также вносит краткое описание задания.

Кроме того, могут быть дополнительно указаны такие параметры, как количество необходимых процессов, необходимость оповещения об изменении статуса задания (задание принято/запущено/завершено) и следует ли послать по электронной почте результаты счета.

После того как указаны все необходимые параметры, заданию назначается приоритет и оно ставится в очередь. Положение задания в очереди определяется по его приоритету, который зависит от приоритета пользователя. Администратор и/или модератор могут повысить или понизить приоритет задания.

4.7.2. *Получение результатов счета*

В любой момент времени пользователь может просмотреть список заданий, находящихся в очереди. Но есть ограничения:

- задания, которые он создал сам;
- задания пользователей из той же организации;

- задания, созданные на основе приложений с открытой лицензией.

Получить результат можно через Web-интерфейс. В этом случае создается список завершенных задач. После просмотра какого-либо элемента этого списка, он (элемент) перемещается в архив заданий. Он представляет собой Web-форму, с помощью которой можно производить следующие действия:

- просмотреть какое-либо задание;
- удалить какое-либо задание из архива.

5. Серверный компонент системы SkyTS

Серверный компонент системы SkyTS действует как связующее звено между T-приложениями и ресурсами, необходимыми для счета пользовательских задач. Он способен выполнять T-приложения в отказоустойчивом режиме и взаимодействовать с вычислительными модулями и базой данных приложений. Серверный модуль написан на интерпретируемом языке TCL, что обеспечивает переносимость кода на большинство современных программно-аппаратных платформ. Язык TCL прост в освоении, активно развивается, и недавно был признан одним из 11-ти наиболее безопасных и защищенных технологий с открытым исходным кодом.

Для работы сервера необходимо наличие полноценной базы данных приложений, в которую поступают данные от Web-интерфейса. Если связь с БД отсутствует, то работа сервера невозможна. Отсутствие связи с БД еще не означает полную неработоспособность всей системы. Предусмотрено наличие нескольких территориально-распределенных серверов, которые взаимодействуют друг с другом, со счетными модулями и с общей базой данных приложений. В случае сбоя или перегруженности какого-либо сервера, другие сервера способны взять на себя дополнительную нагрузку по обслуживанию рабочих подключений.

В данном контексте, рабочее подключение — это компьютер, предоставляющий свободные ресурсы для распределенного счета T-приложений. На этом компьютере установлен, сконфигурирован и запущен счетный модуль системы SkyTS.

Через определенные промежутки времени (а также на начальном этапе работы) сервер делает запрос в БД на предмет заданий, готовых к счету. Постановка заданий в очередь на счет ведется с учетом

времени их запуска через Web-интерфейс и категорий пользователей-владельцев этих заданий. Эти данные определяют приоритет заданий и их порядок в очереди на выполнение.

В соответствии со схемой запуска Т-приложений в отказоустойчивом режиме, серверным модулем вызывается мастер-процесс. Он служит сокет-сервером, который ожидает подключения рабочих процессов по каналам TCP/IP. Рабочие процессы вызываются на тех компьютерах, на которых установлен и настроен счетный модуль системы SkyTS. Статус задания на Web-интерфейсе меняется на „работает“ в тот момент, когда происходит запуск мастер-процесса Т-приложения.

Соединение между серверным и счетными модулями защищено средствами протокола SSL. Серверный и счетный модули используют сертификаты, позволяющие шифровать все пересылки данных надежным ключом длиной в 2048 бит.

Взаимодействие по сети со счетными компонентами системы осуществляется посредством обмена сообщениями в определенном формате в соответствии с сетевым протоколом. В процессе взаимодействия со счетным модулем сервер получает от него информацию о программно-аппаратной архитектуре вычислительного узла, определяет степень его надежности, и принимает решение о том, какое из заданий очереди выделить данному рабочему подключению на счет. Если сервер принял решение об отправке задания данному счетному узлу, то он сообщает информацию о задании (название, версия, объем приложения в байтах, аргументы командной строки), а затем передает исполняемый код приложения, либо архив с приложением и всеми нужными файлами. Если это приложение уже ранее было загружено на рабочий узел, то оно не загружается снова.

6. Счетный компонент системы SkyTS

Счетный компонент системы SkyTS предназначен для инсталляции на обычные компьютеры, ресурсы которых большую часть времени простаивают. Владельцы этих компьютеров могут быть заинтересованы в утилизации аппаратных мощностей в свободное от работы время. С этой точки зрения для проекта подходят как корпоративные локальные компьютерные сети, так и одиночные компьютеры в сети Интернет. Данный компонент также написан на интерпретируемом языке TCL, что особенно важно для его переносимости.

Во время запуска счетный модуль соединяется с сервером и запрашивает задание. Если заданий нет, то программа ждет определенное время, а затем снова повторяет попытку запроса. Число таких итераций не ограничено.

После получения задания счетный модуль генерирует и исполняет командный сценарий на языке „Shell“ или „Windows command shell“ (в зависимости от ОС). Этот сценарий содержит объявления переменных окружения, нужных для подключения рабочих процессов к мастер-процессу, а также команду запуска исполняемого модуля в режиме „smp“ с указанием числа процессов, равного числу ядер процессора на рабочем узле. При успешном завершении работы приложения ведется передача серверу сообщения об этом. В случае потери связи с сервером возможны попытки повтора соединения; если они оказались неудачными, то при следующей попытке запроса задания этой рабочей машиной сервер заносит в БД запись о снижении степени надежности счетного модуля.

7. Заключение

Разработана система распределенных вычислений (Grid-система) SkyTS. Система предназначена для работы T++ приложений в распределенном отказоустойчивом режиме на множестве разрозненных неоднородных компьютеров в сети Интернет. Система может состоять из нескольких серверных и неограниченного количества вычислительных модулей, устанавливаемых на компьютеры в сети Интернет. Подача и мониторинг вычислительных заданий в системе осуществляется авторизованными пользователями через специальный Web-интерфейс. По сравнению с существующими аналогами, Grid-система SkyTS имеет целый ряд преимуществ, среди которых простота реализации, безопасность за счет использования виртуальных машин, масштабируемость на неограниченное множество серверных и счетных узлов, нацеленность на решение прикладных задач в области инженерного анализа и суперкомпьютерного моделирования.

8. Благодарности

Работы, положенные в основу данной статьи, были выполнены в рамках:

- проекта „Разработка и реализация языков T++ и соответствующих им средств для эффективной поддержки высокопроизводительного параллельного счета“ по Программе фундаментальных научных исследований ОНИТ РАН „Архитектура, системные решения, программное обеспечение, стандартизация и информационная безопасность информационно-вычислительных комплексов новых поколений“;
- программы „СКИФ-ГРИД“ „Разработка и использование программно-аппаратных средств ГРИД-технологий и перспективных высокопроизводительных (суперкомпьютерных) вычислительных систем семейства „СКИФ“ (2007–2009 гг.);
- научно-технической программы Союзного государства „Развитие и внедрение в государствах-участниках Союзного государства наукоемких компьютерных технологий на базе мультипроцессорных вычислительных систем (шифр „ТРИАДА“)“ (2005–2008 гг.).

Список литературы

[1] Абрамов С. М., Адамович А. И., Инюхин А. В., Московский А. А., Роганов В. А., Шевчук Ю. В., Шевчук Е. В. *Т-система с открытой архитектурой* // Суперкомпьютерные системы и их применение SSA'2004: Труды Международной научной конференции, 26–28 октября 2004 г., Минск, ОИПИ НАН Беларуси. — Минск, 2004, с. 18–22. ↑1

[2] Официальный сайт системы программирования OpenTS: Электронный сетевой ресурс, http://www.opents.net. ↑1

[3] Абрамов С. М., Кузнецов А. А., Роганов В. А. *Кроссплатформенная версия Т-системы с открытой архитектурой* // Параллельные вычислительные технологии (ПаВТ'2007): Труды Международной научной конференции, 29 января –2 февраля 2007 г., Челябинск. — Челябинск: изд. ЮУрГУ, 2007, с. Т.1, 115–121. ↑1

[4] Абрамов С. М., Московский А. А., Роганов В. А., Велихов П. Е. Пути ученого. Е.П. Велихов: Суперкомпьютерные и GRID-технологии. — М.: РНЦ „Курчатовский институт“, 2007. — ISBN 978-5-9900996-1-6. — 314–324 с., Под общей редакцией академика РАН В.П. Смирнова. ↑1

[5] Абрамов С. М., Есин Г. И., Загоровский И. М., Матвеев Г. А., Роганов В. А. *Принципы организации отказоустойчивых параллельных вычислений для решения вычислительных задач и задач управления в Т-Системе с открытой архитектурой (OpenTS)* // Программные системы: теория и приложения (PSTA-2006): Труды Международной научной конференции, 23–28 октября 2006 г., Переславль-Залесский, ИПС РАН. — Переславль-Залесский, 2006, с. 257–264. ↑2

[6] Кузнецов А. А., Роганов В. А. *Экспериментальная реализация отказоустойчивой версии системы OpenTS для платформы Windows CCS* // Суперкомпьютерные системы и их применение (SSA'2008): Труды Второй Международной научной конференции, 27–29 октября 2008 г., Минск. — Минск: ОИПИ НАН Беларуси, 2008. — ISBN 978-985-6744-46-7, с. 65–70. ↑2

Исследовательский центр мультипроцессорных систем ИПС РАН

G. I. Esin, A. A. Kuznetsov, V. A. Roganov. *Fault-tolerant software prototype "SkyTS" for distributed computing of heavy-load T++ applications in heterogeneous distributed environment* // Proceedings of Program Systems institute scientific conference "Program systems: Theory and applications". — Pereslavl-Zalesskij, v. **1**, 2009. — p. 225–243. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. This paper describes the design and development of the distributed computing system "SkyTS" for running heavy-load T++ parallel applications. A description of the "SkyTS" fault-tolerant scalable prototype is given that supports heterogeneous networks.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 681.32

А. П. Лисица, А. П. Немытых

Об одном приложении вычислений с оракулом

Аннотация. Пусть дана программа $P(d)$, реализующая частично рекурсивную функцию φ. Рассмотрим на области определения последней функцию $\mathcal{O}_P$, значением которой является путь вычисления программы P на фиксированном данном d_0. Пусть программа $Q(p,d)$ определена тогда и только тогда, когда $p = \mathcal{O}_P(d)$; причем $Q(\mathcal{O}_P(d),d) = P(d)$. Программа $Q(p,d)$, абсурдная с точки зрения ее практического вычисления на конкретных входных данных, может быть практически полезной при ее автоматическом мета-анализе. В статье показывается, как программа $Q(p,d)$ может быть использована для верификации программы $P(d)$ по постусловию. Предлагаемый метод опробован на задачах верификации протоколов когерентности кэша и других распределенных вычислительных систем.

1. Моделирование недетерминированных вычислений посредством детерминированных

Уточним понятия, уже введенные нами в аннотации. Прежде всего, будем считать, что программа P может быть недетерминированной. То есть, в процессе вычисления $P(d_0)$ на конкретных входных данных выбор очередной ветви вычисления может быть неоднозначен. Исполнитель программы P управляется оракулом, который и принимает решение о выборе конкретной ветви вычисления на данном шаге этого вычисления. Повторная попытка вычисления $P(d_0)$ (на тех же самых входных данных) может привести к результату вычисления, отличному от первого запуска этой программы. Логика оракула предсказуема лишь в одном: он всегда выбирает конечный путь вычислений, приводящий к конкретному значению $P(d_0)$ (к одному из возможных), если d_0 принадлежит области определения программы P. Таким образом, путь, выбранный оракулом, не может привести Исполнителя ни к аварийной остановке, ни к незавершенности процесса вычислений в конце этого пути. Договоримся

Работа выполнена при частичной поддержке Программы фундаментальных исследований Президиума РАН №3 «Фундаментальные проблемы системного программирования» и Российского фонда фундаментальных исследований (номера проектов: 07-07-92100-GFEN-a, 08-07-00280-a).

также, что в том случае, когда на каком-то шаге вычисления реального выбора нет (выбор ветви однозначен), оракул делает выбор из единственно возможной ветви. Далее под $\vec{d}$ будем понимать конечную последовательность $d_1, \ldots, d_n$. Пусть даны некоторые множества D и $\mathtt{Im}$, мы будем считать, что недетерминированная программа реализует недетерминированную вычислимую[1] функцию из D в $\mathtt{Im}$.[2]

Понятия абстрактной программы и пути эволюции абстрактной программы зависят от конкретной модели вычислений. Понятие конкретной программы P и конкретного пути эволюции программы P (пути вычисления на конкретных входных данных) зависят от выбора языка программирования и операционной семантики этого языка. Ниже мы даем определение Исполнителя недетерминированной программы P и оракула данной программы P в «абстрактных» терминах. В следующем разделе мы уточняем эти понятия для конкретного языка.

Пусть дана абстрактная программа $P(\vec{d})$, реализующая *недетерминированную* частично рекурсивную функцию φ. Пусть D — область определения φ, $\mathtt{Im}$ — множество значений функции φ, Π — множество возможных конечных путей эволюции программы P.

Определение: *Исполнителем абстрактной программы P назовем программу Q, определяющую* детерминированную *частично рекурсивную функцию из $\Pi \times D$ в $\mathtt{Im}$ такую, что: для всех $\vec{d_0} \in D$, если p_0 есть путь вычисления $P(\vec{d_0})$, приводящий к одному из возможных значений $P_{p_0}(\vec{d_0})$, тогда $Q(p_0, \vec{d_0}) = P_{p_0}(\vec{d_0})$. Иначе программа Q не определена и завершает свою работу в состоянии аварийной остановки.*

Определение: *Пусть дана абстрактная программа $P(\vec{d})$ и ее Исполнитель Q. Оракулом программы $P(\vec{d})$ назовем* недетерминированную *общерекурсивную функцию $\mathcal{O}_P : D \mapsto \Pi$ такую, что для всех $d \in D$ выполняется равенство: $Q(\mathcal{O}_P(\vec{d}), \vec{d}) = P_{\mathcal{O}_P(\vec{d})}(\vec{d})$.*

Замечание: *Часто необходимо различать два вида неопределенности программы P: аварийную остановку и бесконечное время эволюции. В этом случае неопределенность аварийной остановки удобно обозначить специальным символом $\perp$, не входящим в оригинальный*

[1] Частично–рекурсивную или общерекурсивную.

[2] То есть, формально, функцию из D в множество всех непустых подмножеств множества $\mathtt{Im}$. См. определение понятия недетерминированной функции в [1].

образ $\mathtt{Im}(P)$ *этой программы, и расширить этот образ до множества* $\mathtt{Im}(P) \cup \{\underline{\perp}\}$.

2. Язык Рефал-Н

Для уточнения понятий программы и пути вычисления, в качестве объектного языка мы определим язык Рефал–Н. Ниже дана грамматика, описывающая множество Рефал-Н программ. Неопределенные терминалы совпадают с соответствующими терминами функционального языка программирования Рефал [2]:

```
prog ::= entry func^*
entry ::= $ENTRY Go { sent^+ }
func ::= fn { sent^+ }

sent ::= sent_name left = right;
left ::= patt call^*
call ::= , <fn arg>: patt
patt ::= expr
right ::= expr
arg ::= expr
expr ::= texpr expr | empty
texpr ::= (expr) | symbol | var | par

var ::= s.name | t.name | e.name
par ::= #s.name | #t.name | #e.name
evar ::= e.name

symbol :: = SYMBOL    /* в смысле Рефала */
empty ::=

sent_name ::= S_i   /* где i  - порядковый номер предложения
                              в конкретной функции func  */
name ::= имя переменной в смысле Рефала
```

Обозначим $\mathtt{Vars}(\mathtt{expr})$ — множество переменных, входящих в выражение `expr`; $\mathtt{Pars}(\mathtt{expr})$ — множество параметров, входящих в выражение `expr`.

Определим дополнительные ограничения на синтаксис предложений `sent`. Для произвольного предложения

$\mathtt{S}_j\ \mathtt{patt}_0\mathtt{,}\ \mathtt{<fn}_1\ \mathtt{arg}_1\mathtt{>:}\ \mathtt{patt}_1\mathtt{,}\ \ldots\mathtt{,}\ \mathtt{<fn}_k\ \mathtt{arg}_k\mathtt{>:}\ \mathtt{patt}_k\ \mathtt{= right;}$

должны выполняться условия: для всех $0 < j \leq k.\mathtt{Vars}(\mathtt{arg}_j) \subset \bigcup_0^{(j-1)}\mathtt{Vars}(\mathtt{patt}_m)$, $\mathtt{Vars}(\mathtt{right}) \subset \bigcup_0^k \mathtt{Vars}(\mathrm{patt}_m)$; для всех $0 \leq j \leq k$ множество $\mathtt{Pars}(\mathtt{patt}_j)$ пусто.

Семантика языка Рефал–Н будет модификацией семантики Рефала (см. [2]) — в сторону недетерминизма. А именно:

- под «переменными» `var` понимаются связанные переменные в данном предложении sent; переменным можно присваивать значения, соответствующие их типу;
- под «параметрами» `par` понимаются свободные переменные в данном предложении `sent`; параметры в каждый момент работы программы имеют конкретные значения, соответствующие их типу, — это значения, которые выбираются Рефал–Н машиной недетерминированным образом;
- последовательность вызовов функций в левой части предложения представляет собой композицию этих вызовов с указанным порядком вычисления. При необходимости вычисления конкретного вызова (из стека вызовов функций):
  - если это стартовый вызов `<Go arg>` (входная точка) функции `Go`, тогда Рефал–Н машина недетерминированным образом фиксирует все параметры, входящие в аргумент `arg` и все параметры, входящие в определение функции `Go`;
  - если вызов `<F arg>` не стартовый, тогда (по предыдущему пункту) его аргумент является объектным выражением (не содержит параметров и переменных). Рефал–Н машина недетерминированным образом фиксирует все параметры, входящие в определение функции `F`. Каждому вызову в стеке соответствует своя недетерминированная выборка указанных в данном пункте параметров;
- выбор предложения функции `F` для отождествления конкретных входных данных `d` (то есть, без параметров) вызова `<F d>` с конкретным образцом $\texttt{patt}_0$ (из левой части предложения) будет происходить недетерминированным образом;
- если выбрано предложение $\texttt{S}_i$ $\texttt{patt}_0$ `call`* `= right;` для попытки отождествления и уравнение $\texttt{patt}_0$ `= d` имеет непустое множество значений, тогда результатом отождествления будет считаться одно из этих решений, которое выбирается недетерминированным образом;
- вычисление конструкции `<fn d>:`$\texttt{patt}_i$ состоит в недетерминированном выборе одного из результатов $\texttt{d}_1$ вызова `<fn d>` и решения уравнения $\texttt{patt}_i$ `=` $\texttt{d}_1$. Если это уравнение имеет

непустое множество значений, тогда результатом отождествления будет считаться одно из решений, выбранное недетерминированным образом;

- имена предложений при отождествлении игнорируются, то есть, являются комментариями и введены нами по техническим причинам.

Все другие понятия семантики языка Рефал–Н совпадают с понятиями семантики языка Рефал [2]. В частности, входной точкой программы, по определению, является вызов функции `<Go expr>`, где `expr` — входное параметризованное выражение (входные данные недетерминированной программы). Заметим, что, в отличие от Рефала, во время отождествления конкретного предложения функции `F` Рефал-Н машина обрабатывает каждый конкретный конструктор–запятую, по определению, только один раз. То есть, она никогда[3] не возвращается к этому конструктору; откаты за конструктор–запятую допускаются семантикой Рефала (см. [2]).

Пример 1. *Пусть дана входная точка* `<Go` n_0`>`, *где* n_0 — *натуральное число, представленное в унарной системе счисления*[4], *тогда нижеследующая Рефал–Н программа вычисляет* n_0*–ое число Фибоначчи.*

```
$ENTRY Go {  S1 e.n, <FibN e.n>: e.fib = e.fib; }
FibN {
S1    = I;
S2  I = I;
S3  I I e.m, <FibN  e.m>: e.x, <FibN I e.m>: e.y = e.x e.y;
}
```

Если не обращать внимание на порядковые номера предложений S_i, *то синтаксис этой программы не выходит за рамки синтаксиса языка Рефал [2], поэтому ее можно рассматривать (с точностью до* S_i*) одновременно и как Рефал программу, и как Рефал-Н программу. В данном случае Рефал–Н семантика программы полностью совпадает с Рефал семантикой. Причины этого в том, что (1) с точки зрения Рефал семантики, предложения в определении* `FibN` *можно переставлять местами, не меняя семантики* `FibN`*; (2) все отождествления однозначны — образцы программы не содержат открытых переменных (см. [2]).*

---

[3] При данном конкретном вызове функции `F`.

[4] Например, `2 = I I`. Ноль представляется пустой строкой.

Пример 2. *Если в предыдущем примере входную точку Рефал–Н программы заменить на* `<Go #e.n>`*, тогда результатом вычисления программы будет либо число Фибоначчи, номер которого выбирается недетерминированным образом; либо аварийная остановка* $\underline{\perp}$*, если недетерминированный выбор конкретного значения параметра* `#e.n` *не представляет натурального числа в унарном счислении.*

Пример 3. *Пусть дана Рефал-Н программа:*

```
$ENTRY Go {  S1 , <Fab A>: e.res = e.res; }
Fab {
S1    = ;
S2  A  e.x, <Fab e.x>: e.z = B e.z;
S3  s.y e.x, <Fab e.x>: e.z = s.y e.z;
}
```

Тогда возможны следующие результаты вычисления ее входной точки `<Go >: A` *или* `B`*.*

Пример 4. *Пусть дана Рефал–Н программа:*

```
$ENTRY Go {  S1 e.x e.y = e.x; }
```

Тогда результатом вычисления ее входной точки `<Go A B C D>` *может быть:*

- *пустое Рефал выражение;*
- `A`*;*
- `A B`*;*
- `A B C`*;*
- `A B C D`*.*

Каждый из указанных вариантов выбирается недетерминированным образом.

3. Понятие пути вычисления в языке Рефал-Н

Будем пользоваться терминологией языка Рефал [2]. Пусть даны Рефал–Н программа `P` и ее входная точка `<Go d>`. Пара (`P,<Go d>`) определяет корневое ориентированное дерево вычислений $\mathcal{T}$ — согласно правилам семантики. $\mathcal{T}$, вообще говоря, может быть бесконечным. Каждая вершина помечена состоянием Рефал–Н машины в данный момент вычисления. Корень `<Go d>` имеет только выходные ребра; все другие вершины имеют по одному входному ребру. Листья дерева

$\mathcal{T}$ помечены либо данным — результатом вычисления программы, либо символом $\perp\!\!\!\perp$ аварийной остановки (тупика). Вершины, не являющиеся листьями, также помечены либо меткой `call`, либо меткой `return`. Вершины типа `call` соответствуют точкам входа в конкретный вызов функции и являются точками ветвления. Вершины типа `return` соответствуют точкам выхода из конкретного вызова функции и также являются точками ветвления. Скажем, что вершина, не являющаяся листом, имеет имя `f`, если она соответствует вызову функции с именем `f`.

Опишем множество ребер, выходящих из вершины b типа `call`, соответствующей вызову `<f d>`, где `d` — Рефал выражение, возможно, содержащее параметры (только тогда, когда рассматриваемый вызов является корнем дерева). Рассмотрим произвольное предложение функции `f`:

$$\texttt{S}_i\ \texttt{patt}_0\ \ldots$$

и соответствующее ему уравнение с параметрами $\texttt{patt}_0$ `= d`. Обозначим σ — отображение–подстановку, сопоставляющее каждому параметру из `Pars`(d) конкретное данное, соответствующее типу этого параметра. Множество решений уравнения $\sigma(\texttt{patt}_0) = \sigma(\texttt{d})$ обозначим через $\mathcal{X}_i^\sigma$. Множество $\mathcal{X}_i^\sigma$ может быть либо пустым, либо конечным.

- Если объединение (по всем возможным подстановкам σ и предложениям функции `f`) множеств $\mathcal{X}_i^\sigma$ пусто, тогда из вершины b выходит единственное ребро, которое является входящим в лист типа $\perp\!\!\!\perp$.
- Иначе, для каждой σ, каждого i и каждого $x_0 \in \mathcal{X}_i^\sigma$ существует единственное ребро, выходящее из b. Пометим это ребро термом `((in` σ`) (`$\texttt{S}_i$ x_0`))`.

Теперь опишем множество ребер, выходящих из вершины b типа `return`, соответствующей конструкции `<f d>:` $\texttt{patt}_j$, где `d` — объектное выражение (оно, по определению семантики (см. выше), не может содержать параметров). Множество всех возможных значений вызова `<f d>` обозначим через $\mathcal{R}$. Оно может быть пустым, конечным или бесконечным. Пусть $\texttt{r}_0 \in \mathcal{R}$. Множество решений уравнения $\texttt{patt}_j = \texttt{r}_0$ обозначим через $\mathcal{X}_{\texttt{r}_0}$. Оно может быть либо пустым, либо конечным.

- Если объединение (по всем возможным $\texttt{r}_0$ из $\mathcal{R}$) множеств $\mathcal{X}_{\texttt{r}_0}$ пусто, тогда из вершины b выходит единственное ребро, которое является входящим в лист типа $\perp\!\!\!\perp$.

- Иначе, для каждого $\mathtt{r}_0 \in \mathcal{R}$ и каждого $x_0 \in \mathcal{X}_{\mathtt{r}_0}$ существует единственное ребро, выходящее из b. Пометим это ребро термом $\mathtt{(out\ (r}_0\ x_0\mathtt{))}$.

Определение: *Пусть даны Рефал–H программа* `P` *и ее входная точка* `<Go d>`*, где* `d` *— параметризованное данное Рефала. Рассмотрим* $p = (v_0, v_1), (v_1, v_2), \ldots (v_{(n-1)}, v_n)$ *— путь в дереве вычислений пары* (`P, <Go d>`)*, выходящий из корня* v_0 *и заканчивающийся в некотором листе* v_n*. Каждому ребру* $(v_i, v_{(i+1)})$ *сопоставим терм* a_i *по следующему правилу:*

- *если* v_i *— вершина типа* `call` *— есть точка ветвления вызова* `<f d>`*, тогда* $a_i =$ `(f ((in` σ`) (`S_i x_0`)))`*, где* `((in` σ`) (`S_i x_0`))` *есть метка ребра* $(v_i, v_{(i+1)})$*;*
- *если* v_i *— вершина типа* `return` *— есть точка выхода из вызова* `<f d>: patt`*, тогда* $a_i =$ `(f (out` x_0`))`*, где* `(out (r`$_0$ x_0`))` *есть метка ребра* $(v_i, v_{(i+1)})$*.*

Пусть $v_n \neq \perp$*. Путем вычисления пары* (`P, <Go d>`) *назовем последовательность термов* $a_0, \ldots, a_{(n-1)}$*.*

Замечание: *Множество путей вычисления пары* (`P, <Go d>`) *может быть конечным, бесконечным или пустым. Каждый путь вычисления является конечной последовательностью и представляет собой историю конкретного вычисления.*

Замечание: *Конкретный путь вычисления однозначно определяет результат каждого вызова функции. Следовательно, данное нами определение пути вычисления корректно.*[5]

Пример 5. *Пусть входная точка программы* `FibN` *(см. параграф 1) есть* `<Go >`*, где*

```
$ENTRY Go {  S1 , <FibN I I>: e.res = e.res; }
```

тогда путь вычисления, соответствующий указанным программе и входной её точке, есть:

$a_0 =$ `(Go ((in ) (`S_1`)))`,
$a_1 =$ `(FibN ((in ) (`S_3 `(e.m := ))))`,
$a_3 =$ `(FibN ((in ) (`S_1 `())))`,
$a_4 =$ `(FibN (out) (e.x := I)))`,

[5]Мы не включили в определение метки $\mathtt{r}_0$, помечающие исходящие ребра вершин типа `return`.

$a_5 =$ `(FibN ((in ) (`S_2 `())))`,
$a_6 =$ `(FibN (out) (e.y := I)))`
Здесь для всех a_i *подстановки* σ *тривиальны и обозначены пустыми выражениями, в* a_3 *и* a_5 *пустыми скобками* `()` *обозначены тривиальные решения.*

4. Верификация с предусловием по постусловию

Пусть даны Рефал–Н программа $P(d)$, ее Исполнитель Q (см. 1) и оракул $\mathcal{O}_P$. Пусть D — область определения программы P, `Data` — множество данных Рефала. Пусть G есть множество `Data` $\cup \{\underline{\perp}\}$.

Определение: *Пусть в множестве* D *выделена некоторая константа* a. *a–тождественной характеристической функцией множества* $B \subset$ `Data` $\setminus \{a\}$ *назовем* общерекурсивную *функцию* $\mathcal{I}_a$: `Data` $\setminus \{a\} \mapsto G$ *такую, что если* $x \in B$, *тогда* $\mathcal{I}_a(x) = x$, *иначе* $\mathcal{I}_a(x) = a$.

Ниже в данном параграфе, функции, имена которых помечены нижним индексом a, будут обозначать a–тождественные характеристические функции. Рассмотрим $\psi_{\underline{\perp}}$, ξ_a — *имярек*-тождественно характеристические функции множеств $A \subset D, Data \setminus B$ соответственно.

Пусть константа `false` не принадлежит образу программы `P`. Под задачей верификации программы `P` с предусловием $\psi_{\underline{\perp}}$ по постусловию $\xi_{\texttt{false}}$ мы будем понимать необходимость доказательства следующего свойства:

$$\forall d \in D.\ \xi_{\texttt{false}}(\texttt{P}(\psi_{\underline{\perp}}(d))) = d' \neq \texttt{false}.$$

Если сформулированное выше утверждение верно, тогда скажем, что программа `P` корректна по предусловию $\psi_{\underline{\perp}}$ и постусловию $\xi_{\texttt{false}}$.

Если $\psi_{\underline{\perp}}$ и $\xi_{\texttt{false}}$ заданы в виде *детерминированных* программ, то можно попытаться поручить решить эту задачу *детерминированной* мета-программе M, анализирующей композицию $\xi_{\texttt{false}}(\texttt{P}(\psi_{\underline{\perp}}(\#d)))$ в контексте определений `P` и $\psi_{\underline{\perp}}, \xi_{\texttt{false}}$, где $\#d$ — параметр, пробегающий множество Рефал данных.

Мы хотим поставить эту задачу для мета–программы M, которая способна анализировать только *детерминированные* программы.

Для всех Рефал данных $d \in D$ имеем равенство

$$Q(\mathcal{O}_P(d), d) = P_{\mathcal{O}_P(d)}(d).$$

Следовательно, для всех $d \in D$

$$\xi_{\texttt{false}}(Q(\mathcal{O}_P(d), \psi_{\underline{\perp}}(d))) = \xi_{\texttt{false}}(P_{\mathcal{O}_P(d)}(\psi_{\underline{\perp}}(d))).$$

Рассмотрим композицию $\xi_{\texttt{false}}(Q(p, \psi_{\underline{\perp}}(d)))$. По определению программ Q и $\xi_{\texttt{false}}$, $\psi_{\underline{\perp}}$ вычисление этой композиции всегда завершается, но, возможно, в аварийном состоянии Рефал машины (напомним, что все эти программы *детерминированные*). Таким образом, если M сможет доказать равенство $\xi_{\texttt{false}}(Q(p, \psi_{\underline{\perp}}(d))) = d' \neq \texttt{false}$ для любого $p \in \Pi$ и любого $d \in D$, тогда для любого $d \in D$ имеет место равенство $\xi_{\texttt{false}}(P(\psi_{\underline{\perp}}(d))) \neq \texttt{false}$.

И поставленная в начале данного параграфа задача будет решена.

Ниже, в параграфе 7, мы покажем, что описанная нами схема успешно может быть применена для верификации реальных распределенных недетерминированных вычислительных систем, *работающих бесконечное время*.

5. Двойственные вычислительные системы

В этом параграфе мы определим понятия двойственной недетерминированной вычислительной системы S' по отношению к данной недетерминированной вычислительной системе S и двойственной задачи верификации системы S' по отношению к данной задаче верификации системы S.

Пусть дана некоторая абстрактная недетерминированная вычислительная система S. Под двойственной к S системой S' мы будем понимать вычислительную систему, все шаги эволюции которой противоположны шагам эволюции системы S. То есть, если система S из состояния q переходит в состояние q', тогда система S' из состояния q' переходит в состояние q.

Определение: *Пусть* $D \subset \texttt{Data}$. *Генератором множества* D *назовем недетерминированную программу* $\gamma_D : \epsilon \mapsto D$, *образ которой совпадает с* D.[6] *Параметризованное выражение* ***expr*** *определяет некоторое подмножество в множестве данных Рефала* Data. *Обозначим генератор этого подмножества через* $\gamma_{\texttt{expr}}$.

Уточним это понятие в конкретных терминах языка Рефал–Н. Во–первых, мы построим граф вычислений данной программы P,

[6] Здесь ϵ — пустое Рефал выражение.

который представляет дерево вычислений программы P в виде, симметричном относительно входа и выхода P.

Пусть даны Рефал–Н программа P и ее входная точка `<Go expr>`. В параграфе 3 мы описали дерево вычислений $\mathcal{T}$, определяемое этой парой. Корень дерева $\mathcal{T}$ определяет входные данные вычислительной системы S, соответствующей программе P; листья дерева $\mathcal{T}$ — выходные данные системы S. Легко видеть, что посредством переопределения P можно ограничиться входными точками вида `<Go #e.n>`, не меняя основную часть структуры дерева вычислений $\mathcal{T}$. Для этого достаточно переопределить функцию `Go` к виду:

```
$ENTRY Go {
  s1 patt, <Out arg>: patt1 = right;
}
```

Где `patt` — образец, построенный из `expr` заменой всех параметров на одноименные им переменные; `Out` — вспомогательная функция, передающая управление другим функциям из P посредством одного шага вычисления программы (одного ребра в дереве вычислений программы), $\texttt{patt}_1$ — образец, принимающий результат вычисления вызова функции `Out`. Ниже, если не оговорено противное, мы ограничимся входными точками вида `<Go #e.n>` и определением функции `Go` описанного вида. Назовем программы, определение `Go` которых удовлетворяет описанным синтаксическим свойствам, нормализованными.

Все выходные ребра из `return`–вершин $\texttt{<Out arg>: patt}_1$ дерева вычислений $\mathcal{T}$ являются входными ребрами листов. Причем, для каждого листа b дерева $\mathcal{T}$, метка которого не равна $\perp$, существует `return`–вершина $\texttt{<Out arg>: patt}_1$ такая, что входное ребро листа b является выходным ребром вершины $\texttt{<Out arg>: patt}_1$. Обозначим множество возможных значений вызова `<Go #e.n>` через $\mathtt{Im}_P$. Склеим все листы дерева, метки которых не равны $\perp$, и полученную таким образом вершину пометим как $\texttt{<Out } \gamma_{\mathtt{Im}_P}()\texttt{>}$. Построенный граф назовем графом вычислений пары `(P,<Go #e.n>)`. Для полной симметричности входа и выхода далее входную точку программы будем представлять следующим образом: $\texttt{<Go } \gamma_{\#\mathtt{e.n}}()\texttt{>}$.

Рассмотрим следующее формальное преобразование $\mathcal{W}$ нормализованной Рефал–Н программы P.

$$\mathcal{W}(\texttt{entry func}^*) = \mathcal{W}(\texttt{entry})\ \ \mathcal{W}(\texttt{func})^*$$

$$\mathcal{W}(\texttt{\$ENTRY Go \{ sent}^+\ \texttt{\}}) = \texttt{\$ENTRY Go \{ }\mathcal{W}(\texttt{sent})^+\ \texttt{\}}$$
$$\mathcal{W}(\texttt{fn \{ sent}^+\ \texttt{\}}) = \texttt{fn}'\ \texttt{\{ }\mathcal{W}(\texttt{sent})^+\ \texttt{\}}$$

$\mathcal{W}(\mathtt{S_j\ patt_0}, \mathtt{<fn_1\ arg_1>: patt_1}, \ldots, \mathtt{<fn_k\ arg_k>: patt_k} = \mathtt{right};)$
$= \mathtt{S_j}\ \mathcal{U}(\mathtt{right}, \mathtt{<fn'_k\ patt_k>: arg_k}, \ldots, \mathtt{<fn'_1\ patt_1>: arg_1} = \mathtt{patt_0});$

где отображение $\mathcal{U}$ преобразует некоторые переменные и параметры по правилу:

- Для всех j таких, что $0 \leq j < k$, для всех переменных
$$v \in \mathtt{Vars}(\mathtt{patt}_j) \setminus (\mathtt{Vars}(\mathtt{right}) \cup \bigcup_{m=j+1}^{k} \mathtt{Vars}(\mathtt{arg}_m))$$
преобразовать v в параметр со «свежим» именем и типом, совпадающим с типом переменной v.
- Для всех параметров $p \in \mathtt{Pars}(\mathtt{right})$ преобразовать p в переменную со «свежим» именем и типом, совпадающим с типом параметра p.
- Для всех j таких, что $0 < j \leq k$, для всех параметров $p \in \mathtt{Vars}(\mathtt{arg}_j)$ преобразовать p в переменную со «свежим» именем и типом, совпадающим с типом параметра p.

Здесь каждое преобразование конкретной переменной/параметра происходит: (1) последовательно по убыванию индекса j[7] и (2) одновременно по всем вхождениям этой переменной/параметра в рассматриваемое предложение `sent`.

Программу $P' = \mathcal{W}(P)$ назовем программой обратной/двойственной к программе P.

Пример 6. *Нижеследующая Рефал–Н программа P с входной точкой* `<Go #n>` *является нормализованной версией программы из примера 2 (параграф 2).*

```
$ENTRY Go {  S1 e.n, <Out e.n>: e.out = e.out; }
Out {  S1 e.n, <FibN e.n>: e.fib = e.fib; }

FibN {
S1    = I;
S2  I = I;
S3  I I e.m, <FibN  e.m>: e.x, <FibN I e.m>: e.y = e.x e.y;
}
```

Двойственная ей программа P' выглядит следующим образом:

[7] То есть, слева направо.

```
$ENTRY Go {  S1 e.out, <Out' e.out>: e.n = e.n; }
Out' {  S1 e.fib, <FibN' e.fib>: e.n = e.n; }

FibN' {
S1  I = ;
S2  I = I;
S3  e.x e.y, <FibN'  e.y>: I e.m, <FibN' e.x>: e.m = I I e.m;
}
```

Возможные результаты вычисления входной точки `<Go I>` *программы* P'*:*

- *Путь вычисления* `(Go ((in I) (S1 (e.out := I)))`
  `(Out' ((in I) (S1 (e.fib := I)))) (FibN' ((in I) (S1 ())))`
  `(FibN' (out) (e.n := )) (Out' (out) (e.n := ))` *соответствует результату* `<Go I> =` [8].
- *Путь вычисления* `(Go ((in I) (S1 (e.out := I)))`
  `(Out' ((in I) (S1 (e.fib := I)))) (FibN' ((in I) (S2 ())))`
  `(FibN' (out) (e.n := I)) (Out' (out) (e.n := I))` *соответствует результату* `<Go I> = I`.
- *Все пути вычисления, проходящие через третье предложение* S_3 *функции* `FibN'`*, приводят к результату отождествления со значением пустого выражения либо переменной* `e.x`*, либо переменной* `e.y`*. И, следовательно, к бесконечному пути вычисления; ибо выхода по пустому выражению определение функции* `FibN'` *не содержит.*

Для любой нормализованной Рефал–Н программы P верны следующие утверждения:

Утверждение 1: *С точностью до переименования функций, переменных и параметров выполняется равенство* $P'' = P$.

Утверждение 2: *Пусть* $p = (v_0, v_1), (v_1, v_2), \ldots (v_{(n-1)}, v_n)$ *— путь в дереве вычислений пары* $(\mathtt{P}, <\mathtt{Go\ d}>)$*, выходящий из корня* v_0 *и заканчивающийся в некотором листе* v_n*, не отмеченном как* $\underline{\perp}$*. Тогда* $p' = (v_n, v_{(n-1)}) \ldots (v_2, v_1), (v_1, v_0)$ *есть путь в дереве вычислений пары* $(\mathtt{P}', <\mathtt{Go}\ \gamma_{\mathtt{Im_P}}()>)$*, выходящий из корня* v_n *и заканчивающийся в листе* v_0*, не отмеченном как* $\underline{\perp}$*. (В случае графа вычислений,* p *есть*

[8] Пустое выражение.

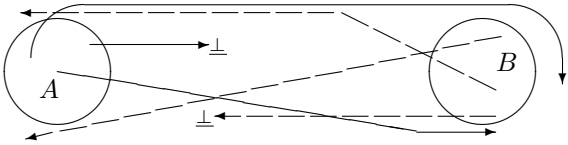


Рис. 1. Пути прямой и двойственной систем, соответствующие прямой и двойственной задачам верификации

путь из вершины <Go d> *в вершину* <Out $\gamma_{\mathtt{Im_P}}()$>, *а* p' *есть путь из вершины* <Out $\gamma_{\mathtt{Im_P}}()$> *в вершину* <Go d>.)

Утверждение 3:
Пара (P′,<Go $\gamma_{\mathtt{Im_P}}()$>) *определяет отношение на* Data × Data *обратное к отношению, определяемому парой* (P,<Go d>).

6. Задача верификации, двойственная к данной

Пусть P — программа, для которой существует двойственная программа P'. Пусть $\mathtt{D}_P$ — область определения программы P, а $\mathtt{Im}_P$ — ее образ. Пусть дано некоторое множество $B \subset \mathtt{Im}_P$. Рассмотрим $\psi_{\underline{\perp}}, \xi_a$ — *имярек*-тождественно характеристические функции множеств $A \subset \mathtt{D}_P$, $\mathtt{Im}_P \setminus B$ соответственно.

Пусть константа $\mathtt{false} \notin \mathtt{D}_P \cup \mathtt{Im}_P$. Пусть дана задача $\mathcal{Z}$ верификации программы P с предусловием $\psi_{\underline{\perp}}$ по постусловию $\xi_{\mathtt{false}}$. Задачей $\mathcal{Z}'$ — двойственной к задаче $\mathcal{Z}$ — назовем задачу верификации программы P' с предусловием $\xi'_{\underline{\perp}}$ по постусловию $\psi'_{\mathtt{false}}$, где $\xi'_{\underline{\perp}}$ есть тождественно характеристическая функция множества B, а $\psi'_{\mathtt{false}}$ — тождественно характеристическая функция множества $\mathtt{D}_P \setminus A$.

Утверждение: *Программа* P *корректна по предусловию* $\psi_{\underline{\perp}}$ *и постусловию* $\xi_{\mathtt{false}}$ *тогда и только тогда, когда программа* P' *корректна по предусловию* $\xi'_{\underline{\perp}}$ *и постусловию* $\psi'_{\mathtt{false}}$.
Доказательство: Предположим обратное — программа P' корректна, а программа P не корректна по указанным в утверждении условиям. Тогда в графе вычислений пары $(P, <\mathtt{Go}\ \gamma_A()>)$ существует путь p из вершины <Go $\gamma_A()$> в вершину <Out $\gamma_B()$>. Но тогда в графе

вычислений пары $(P', <\texttt{Go}\ \gamma_B()>)$ путь p' (двойственный к p) начинается в вершине $<\texttt{Go}\ \gamma_B()>$ и заканчивается в вершине $<\texttt{Out}\ \gamma_A()>$. Что противоречит сделанному предположению. □

Таким образом, решив задачу верификации $\mathcal{Z}$, мы имеем решение двойственной ей задачи $\mathcal{Z}'$ и наоборот. Если задача $\mathcal{Z}$ поставлена перед мета–программой M, анализирующей программу P и ее параметризованную входную точку, тогда цель программы M — доказать, что не существует путей эволюции этой пары из множества A в множество B. Попытка решения двойственной задачи $\mathcal{Z}'$ должна доказать отсутствие путей эволюции соответствующей ей пары из множества B в множество A. Таким образом, в задаче $\mathcal{Z}$, по существу, исследуются существующие пути из A в $\mathtt{Im}_P \setminus B$, а в задаче $\mathcal{Z}'$ — пути из B в $D \setminus A$. Два указанных здесь множества путей, рассматриваемых с точностью до ориентации, не пересекаются. Тем самым для программы M одна из задач $\mathcal{Z}$, $\mathcal{Z}'$ может оказаться значительно проще другой задачи. То есть, построенная нами конструкция может иметь практическое значение.

7. Результаты экспериментов

В работах [3–7] мы показали, что описанная нами в параграфе 4 задача верификации может быть успешно решена для класса параметризованных недетерминированных протоколов кэш когерентности мультипроцессорных систем [8–10] автоматическим специализатором программ [11, 12]. В наших экспериментах[9] мы использовали в качестве такого специализатора суперкомпилятор SCP4 (см. [13–18]) программ, написанных на функциональном языке программирования Рефал–5 [2, 19].

Недетерминированные протоколы кэш когерентности моделировались нами по схеме, данной нами в параграфе 1. Описание этих протоколов и деталей их кодировки в виде программ на языке Рефал–5 можно найти в работах [6, 9, 10]. Отметим, что протоколы кэш когерентности представляют собой вычислительные системы, работающие бесконечно долго. Под параметризацией, в данном случае, понимается параметризация по числу процессоров. Условие корректности формулируется в терминах недостижимости определенного

[9]Результаты этих и других успешных экспериментов представлены на Интернет странице [6].

вида глобальных состояний соответствующей вычислительной системы, работа которой может начинаться из определенного множества ее стартовых состояний. Условие принадлежности к этому множеству и является предусловием задачи верификации. Возможность автоматического решения этой задачи, конечно, зависит от свойств применяемого специализатора, который специализирует рассматриваемую систему по множеству допустимых стартовых конфигураций.

Говоря точнее, для кодировки этих протоколов использовался ограниченный Рефал (см. [19]) — некоторый фрагмент языка Рефал–5. В ограниченном Рефале не допускается использование конструктора *запятой* (см. параграф 2 и [19]) и наложены дополнительные синтаксические ограничения на использование образцов, которые обеспечивают единственность или отсутствие решений уравнений, возникающих при отождествлении с образцами. Кроме того, ограниченный Рефал, как подмножество Рефала–5, конечно, является детерминированным языком программирования. Данные свойства ограниченного Рефала не позволяют использовать напрямую нашу синтаксическую конструкцию построения двойственной программы к данной программе (параграф 5).

Мы закодировали двойственные вычислительные системы к ряду протоколов кэш когерентности вручную, оставаясь в рамках семантической идеи (см. параграф 5). Позволим себе повторить ее еще раз.

Пусть дана некоторая абстрактная недетерминированная вычислительная система S. Под двойственной к S системой S' мы будем понимать вычислительную систему, все шаги эволюции которой противоположны шагам эволюции системы S. То есть, если система S из состояния q переходит в состояние q', тогда система S' из состояния q' переходит в состояние q.

Суперкомпилятор SCP4 смог удачно решить двойственные задачи верификации для следующих параметризованных протоколов кэш когерентности — Synapse N+1 [9, 20], MSI [21], MOSI [22], MESI [9, 20], MOESI [9, 20], The University of Illinois [9, 20], Berkley [9, 20], DEC Firefly [9, 20], IEEE Futurebus+ [9, 20], Xerox PARC Dragon [10]; а также и других параметризованных вычислительных систем — Java Meta–Locking Algorithm [23], Reader–Writer protocol [24], Steve German's directory–based consistency protocol [25], Load Balancing Monitor protocol, Data Race Free Synchronization Model [10], Control Server

Monitor protocol [8].[10] Фактически, эксперименты по автоматическому решению двойственных задач верификации по отношению к данным задачам оказались успешными *для всех закодированных нами двойственных вычислительных систем*. Но ручная кодировка этих систем является трудоемким процессом.

8. Заключение

Насколько нам известно, интерес исследователей к обращению эволюции вычислительных систем проявлялся, в основном, с двух точек зрения. Физики рассматривали задачу построения обращаемых по времени вычислительных машин как наиболее энергосберегаемых (см., например [26]). Программистов интересовал вопрос автоматического построения эффективной программы, вычисляющей обратный алгоритм по отношению к данному алгоритму. Если отбросить требование эффективности, то классическим примером языка программирования, поддерживающего «вычислительно неориентированный» стиль мышления, является логический язык Пролог. Одна и та же программа на языке Пролог описывает одновременно прямое и обратное отношение на декартовом произведении множеств определения ее аргументов. Важно подчеркнуть, что семантика программы совершенно симметрична по отношению к прямому и обратному отношению[11]. Так как Пролог позволяет описывать только предикаты, то оба этих отношения только *выделяются* их характеристическими функциями, а *не вычисляются* — даже тогда, когда отношение на паре множеств (X, Y) представляет собой график взаимно–однозначной функции f.

Интересной задачей является построение по данной программе, реализующей функцию f в терминах функционального языка программирования, программы, вычисляющей обратную к f функцию, если f^{-1} существует, или, иначе, обратное отношение к f — в терминах того же языка программирования. Этой задачей занимались достаточно много авторов. Мы отметим здесь обращаемый по времени[12] язык программирования Janus, разработанный в 1982 году для

[10] Описание всех этих протоколов можно найти также на нашей электронной странице [6].

[11] Мы вынуждены оговориться: если эта программа не использует конструкцию `CUT`.

[12] time–reversible

студенческих экспериментов в Калифорнийском институте технологии (Caltech) [27]. Любые попытки решения этой задачи приводят к симбиозу понятий логических и функциональных языков программирования (см., например, [28]). С нашей точки зрения, большой интерес представляют работы С. Абрамского по обращению вычислений (например, [29]). В контексте языка Рефал в этом направлении активно работал А.Ю. Романенко [30, 31]. По сути дела, он предпринимал попытки построения функционально–логического языка, основанного на понятиях Рефала, включающего в свою трансформационную семантику элементы суперкомпиляции. К сожалению, А. Романенко не имел в своем распоряжении какого–либо приемлемого преобразователя–оптимизатора программ, который бы позволил ему проводить интересные эксперименты. В последнее время интерес к обращению вычислений возник в связи с теорией квантовых вычислений [32].

Нам неизвестны какие–либо работы, связанные с обращением вычислений в контексте верификации недетерминированных вычислительных систем. Недетерминированность вычислений снимает некоторые проблемы обращения вычислений. Конструкция, описанная нами в данной статье, как мы показали на экспериментах, иногда позволяет автоматически верифицировать программные кодировки недетерминированных вычислительных систем; формально обращать их на уровне синтаксиса и ставить задачу верификации обратной вычислительной системы, решение которой приводит к автоматическому решению прямой задачи верификации. При этом задачи верификации решались нами автоматически посредством суперкомпилятора SCP4. Кодировка вычислительной системы, обратной к данной, которую мы назвали двойственной, строилась нами вручную. Задача состоит в автоматизации этой кодировки на основе конструкции, которую мы описали в данной работе.

Список литературы

[1] Лялин И. В. *О решении автоматных уравнений*: Дискрет. матем. Т. **16:2**, 2004, с. 104–116. ↑2

[2] Turchin V. F., Turchin D. V., Konyshev A. P., Nemytykh A. P. Refal–5: sources, executable modules, 2000, http://www.botik.ru/pub/local/scp/refal5/. ↑2, 1, 3, 7

[3] Лисица А. П., Немытых А. П. *Верификация как параметризованное тестирование (эксперименты с суперкомпилятором SCP4)*: «Программирование». — Т. **33(1)**. — Москва, 2007, с. 22–43. ↑7

[4] Lisitsa A. P., Nemytykh A. P. *Verification of parameterized systems using supercompilation. A case study* // Of APPSEM05 ред. Hofmann M., Loidl H. W. — Germany, 2005, Accessible via: ftp://www.botik.ru/pub/local/scp/refal5/appsem_verification2005.ps. ↑

[5] Lisitsa A. P., Nemytykh A. P. *Verification via Supercompilation*: International Journal of Foundations of Computer Science. — Т. **19(4)**, 2008, с. 953–970. ↑

[6] Lisitsa A. P., Nemytykh A. P. Experiments on verification via supercompilation, 2007, http://refal.botik.ru/protocols/. ↑7, 9, 10

[7] Lisitsa A. P., Nemytykh A. P. *Verification as Specialization of Interpreters with Respect to Data* // Of First International Workshop on Metacomputation in Russia. — Pereslavl-Zalessky, 2008, с. 94–112. ↑7

[8] Begin L. The BABYLON Project: A Tool for Specification and Verification of Parameterized Systems to Benchmark Infinite-State Model Checkers, http://www.ulb.ac.be/di/ssd/lvbegin/CST/. ↑7

[9] Delzanno G. *Automatic Verification of Parameterized Cache Coherence Protocols* // Of the 12th Int. Conference on Computer Aided Verification: LNCS. — Т. **1855**: Springer–Verlag, 2000, с. 53–68. ↑7

[10] Delzanno G. Automatic Verification of Cache Coherence Protocols via Infinite-state Constraint–based Model Checking, http://www.disi.unige.it/person/DelzannoG/protocol.html. ↑7

[11] Jones N. D., Gomard C. K., Sestoft P. Partial Evaluation and Automatic Program Generation: Prentice Hall International, 1993. ↑7

[12] Nemytykh A. P. *On the Place of Supercompilation inside Program Specialization* // Of First International Workshop on Metacomputation in Russia. — Pereslavl-Zalessky, 2008, с. 131–144. ↑7

[13] Корлюков А. В. Пособие по суперкомпилятору SCP4, 1999, http://www.refal.net/supercom.htm. ↑7

[14] Немытых А. П. Суперкомпилятор SCP4: общая структура. — Moscow: URSS, 2007. ↑

[15] Nemytykh A. P. *The Supercompiler SCP4: General Structure (extended abstract)* // Of the Perspectives of System Informatics: LNCS. — Т. **2890**: Springer–Verlag, 2003, с. 162–170, Accessible via:ftp://www.botik.ru/pub/local/scp/refal5/nemytykh_PSI03.ps.gz. ↑

[16] Nemytykh A. P. *The Supercompiler SCP4: General Structure*: Программные системы: теория и применение. — Т. **1**. — Москва: Физматлит, 2004, с. 448–485, Available at ftp://ftp.botik.ru/pub/local/scp/refal5/GenStruct.ps.gz. ↑

[17] Nemytykh A. P., Turchin V. F. The Supercompiler SCP4: sources, on–line demonstration, 2000, http://www.botik.ru/pub/local/scp/refal5/. ↑

[18] Turchin V. F. *Theconceptofasupercompiler*: ACM Transactions on Programming Languages and Systems. — Т. **8**: ACM Press, 1986, с. 292–325. ↑7

[19] Turchin V. F. Refal–5, Programming Guide and Reference Manual. — Holyoke, Massachusetts: New England Publishing Co., 1989,
(electronic version: http://www.botik.ru/pub/local/scp/refal5/ ,2000). ↑7

[20] Delzanno G. Automatic Verification of Parameterized Cache Coherence Protocols, 2000, ftp://ftp.disi.unige.it/person/DelzannoG/papers/ccp.ps.gz. ↑7

[21] Emerson E. A., Kahlon V. *Rapid Parameterized Model Checking of Snoopy Cache Coherence Protocols* // Of the 9th International Conference on Tools and Algorithms for the Construction and Analysis of Systems: LNCS. — T. **2619**: Springer–Verlag, 2003, с. 114–159. ↑7

[22] Martin M. M. K. Token Coherence: Ph.D. dissertation: University of Wisconsin, 2003. — Page 19 с., http://www.botik.ru/pub/local/scp/refal5/. ↑7

[23] Roychoudhury A., Ramakrishnan I. V. *Inductively Verifying Invariant Properties of Parameterized Systems*: Automated Software Engineering. — T. **11**, 2004, с. 101–139. ↑7

[24] Fribourg L. *Petri Nets, Flat Languages and Linear Arithmetic* // Of the 9th Int. Workshop. on Functional and Logic Programming, 2000, с. 344–365. ↑7

[25] Delzanno G., Bultan T. *Constraint-based Verification of Client-Server Protocols* // Of the 7th International Conference on Principles and Practice of Constraint Programming: LNCS. — T. **2239**: Springer–Verlag, 2001, с. 286–301. ↑7

[26] Bennett C. H. *Logical reversibility of computation*: IBM J. Res. Develop. T. **17**, 1973, с. 525–532. ↑8

[27] Lutz Ch., Derby H. Janus: a time-reversible language: Ph.D. dissertation: California Institute of Technology, 1982, Unpublished report. ↑8

[28] Klimov Yu. A., Orlov A. Y. *XSG: Fair Language with Built-in Equality* // Of First International Workshop on Metacomputation in Russia. — Pereslavl-Zalessky, 2008, с. 85–93. ↑8

[29] Abramsky S. *A Structural Approach To Reversible Computation*: Theoretical Computer Science. — T. **347(3)**, 2005, с. 441–464. ↑8

[30] Romanenko A. Y. *The generation of inverse functions in Refal*: Partial Evaluation and Mixed Computation ред. Bjørner D., Ershov A. P., Jones N. D. — North-Holland, 1988, с. 427–444. ↑8

[31] Romanenko A. Y. *Inversion and metacomputation*: ACM SIGPLAN Notices. — T. **26(9)**: ACM, 1991, с. 12–22. ↑8

[32] Vitányi P. *Time, space, and energy in reversible computing* // Of the 2nd Conference on Computing Frontiers. — Ischia, Italy: ACM, 2005, с. 435–444. ↑8

The University of Liverpool

Исследовательский центр мультипроцессорных систем ИПС РАН

A. P. Lisitsa, A. P. Nemytykh. *On one application of computations with oracle* // Proceedings of Program Systems institute scientific conference "Program systems: Theory and applications". — Pereslavl-Zalesskij, v. **1**, 2009. — p. 245–265. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. Let a program $P(d)$ implementing a partial recursive function φ be given. Let us consider a function $\mathcal{O}_P$ defined on the domain of φ. Fixed a data d_0, let the function $\mathcal{O}_P$ map d_0 to the path of computation of the program P on the fixed data d_0. Let a program $Q(p, d)$ be defined iff $p = \mathcal{O}_P(d)$ and let the value of the program be $Q(\mathcal{O}_P(d), d) = P(d)$. The program $Q(p, d)$, which is totally absurd with point of view of its practical computation on concrete input data, might be practically useful when it analyzed by a metaprogram. In the paper we show how the program $Q(p, d)$ can be used with a goal to verify a postcondition imposed on the program $P(d)$. The method suggested in the paper was tested on the verification tasks of cache coherence protocols and other distributed computing systems.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 517

И. В. Гордин

Компьютеризация общества как фактор разрешения экологических противоречий

Аннотация. Экологическая кибернетика активно изучает роль компьютеризации в формировании антропогенной нагрузки на биосферу. Множество потребностей человечества сегодня удовлетворяется с использованием компьютерных технологий, ослабляя негативные экологические воздействия (от транспорта, целлюлозно-бумажной промышленности и т.д.). Удовлетворение многих потребностей переходит в виртуальное пространство, полностью исключая ущерб для биосферы и ресурсов планеты. Компьютер становится реальным экологизатором жизни на Земле.

Экологическая кибернетика плодотворно использует компьютерные технологии для решения своих стратегических и оперативных задач, начиная с имитационных математических моделей мировой динамики (климат, демография, экология и экономика) и заканчивая включением компьютерных блоков в системы автоматического регулирования эксплуатационных режимов очистных сооружений. Вместе с этими техническими аспектами все более актуальным становится гуманитарное осмысление процесса компьютеризации как эколого-цивилизационного феномена.

Компьютеризация является одной из ведущих составляющих научно-технического прогресса, играя важную роль как в негативном, так и позитивном влиянии последнего на экологию планеты. Негативное влияние технократической цивилизации исторически первично и остается главным фактором деградации и разрушения биосферы. Вместе с тем, осознав эту опасность, человечество сумело найти и рычаги защиты окружающей среды с использованием технических средств.

Сегодня научно-техническому прогрессу отводится огромная роль в сдерживании и компенсации глобальных и локальных экологических угроз [1–5]. Именно благодаря ему мы вытесняем из производственных процессов, а значит из окружающей среды и своего организма, смертельно опасные вещества (рис. 1). Именно он позволяет обезвредить производственные и хозяйственно-бытовые отходы,

которые ещё совсем недавно повсеместно убивали природу и человека (рис. 2, рис. 3).

Рис. 1. Физиологические реакции населения на экологизацию производственных процессов в промышленности и сельском хозяйстве

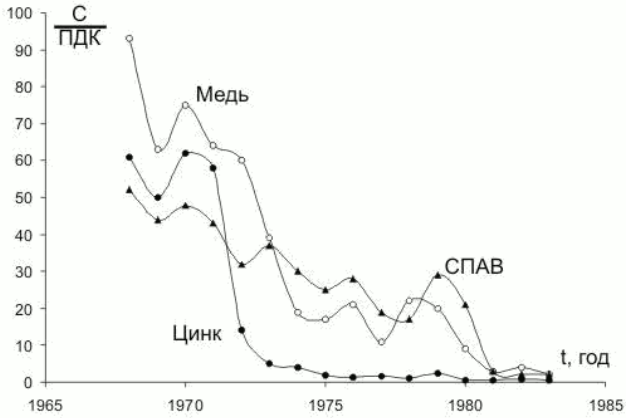


Рис. 2. Радикальное улучшение качества волжской воды ниже Тверского промузла в период наиболее интенсивного водоохранного строительства

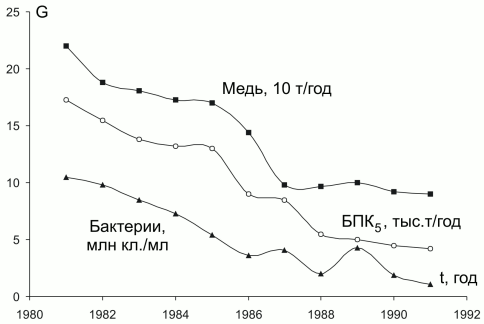


Рис. 3. Сокращение сброса загрязнений в водные объекты Санкт-Петербурга и соответствующее снижение бактериальной загрязненности Невы

И компьютер играет всё более важную роль в оптимизации стратегий защиты природной среды, конкретных проектных решений, становится функционально необходимым звеном АСУ ТП природоохранных комплексов. При этом нельзя не заметить поразительный разрыв между фантастическими достижениями человечества в области компьютерной техники и беспомощностью в решении самых насущных экологических проблем (включая обеспечение даже минимальных условий здоровой жизни, начиная с чистоты воды и воздуха).

На современном этапе развития цивилизации в философской оценке баланса экологически негативного и позитивного воздействия научно-технического прогресса главным является то, что именно он породил экологический кризис, а решает возникшие проблемы выборочно и частично. В этом глобальном аспекте данный пессимизм относится и к процессу компьютеризации в той мере, в которой она способствовала технико-экономическому развитию экологически опасных отраслей. Но в последнее время компьютеризация многое сделала для того, чтобы человечество глубоко осознало гибельность неукротимого технического прогресса.

Непрерывно нарастает роль компьютерных исследований в прогнозировании экологических последствий развития техники и выборе оптимальных путей предотвращения экологических катаклизмов. Так, именно компьютеризация экологических исследований позволила найти аргументы, заставившие сверхдержавы отказаться от соблазна ядерного выяснения отношений. Несколько подробнее остановимся на этом примере выдающегося компьютерного исследования, предотвратившего вполне вероятную экологическую катастрофу.

Исследования проводились в секторе климатических моделей ВЦ АН под руководством академика Н.Н.Моисеева, который так характеризовал их значение [3]: «К сожалению, в большинстве публикуемых учеными материалов о возможных последствиях ядерной войны до сих пор преобладали качественные оценки. Но в наш рациональный век людей больше всего убеждают именно цифры, и их-то порой не хватало. Недостаточная определённость в оценках последствий войны открыла путь различным домыслам о её допустимости, о возможности победы в ней, служила оправданием политики военно-промышленных комплексов, направленной на расширение ядерных арсеналов...

Результаты расчетов заставляют взглянуть на всю проблему по-новому... До сих пор можно было рассчитывать, что ужасы ядерной войны обойдут какие-то уголки, что потери будут громадными, но все же ограниченными, что какие-то регионы вообще не будут затронуты войной и смогут даже что-то выиграть от взаимного ослабления или уничтожения участников конфликта. Цифры показывают, что надеяться на это не приходится. В отличие от непосредственных факторов ядерного поражения факторы климатические носят глобальный характер. «Ядерная зима» и «ядерная ночь», отсутствие света, пищи, пресной воды, отравление атмосферы токсичными газами затронут всю планету в равной степени. В этой войне не может быть не только победителей и побежденных, но даже нейтральных. Причем роковым может оказаться и сравнительно небольшой ядерный конфликт.

В ноябре 1983 года в вашингтонском отеле «Шератон» состоялась международная конференция «Мир после ядерной войны». В центре внимания политиков и ученых оказались впервые вынесенные на публичное обсуждение американская и российская математические модели климатических последствий ядерного столкновения. Предельно ясная интерпретация результатов математического моделирования и практически полное совпадение результатов расчетов, выполненных двумя изолированно работающими в разных странах группами специалистов, произвели на участников конференции огромное впечатление.

Вскоре группа советских ученых во главе с вице-президентом Академии наук Е. П. Велиховым была приглашена на специальные слушания в сенат США.

Обсуждение началось с яркого, эмоционального выступления сенатора Э. Кеннеди [3]: «Я, как и многие американцы, обеспокоен непрекращающимися свидетельствами того, что кое-кто в нынешней администрации считает возможным выжить и победить в ядерной войне. Так, всего за неделю до этой встречи Управление по мобилизации в чрезвычайных обстоятельствах направило в Белый дом оптимистичный доклад о перспективах сельскохозяйственного производства и обеспечения продовольствием во время ядерной войны».

Кеннеди с возмущением цитировал фразы о радужных перспективах сбора урожая после начала войны, обоснованных в докладе тем, что сельское население должно пострадать от взрывов меньше городского, тем, что беженцы из разбомблённых городов создадут на селе избыток рабочей силы, благоприятствующий быстрому сбору

урожая. Государственный деятель откровенно издевался над потугами дилетантов в системном анализе ситуации.

Закрывавший конференцию академик Е. П. Велихов так подытожил обсуждение [3]: «Военные средства превратились в триггер, включающий цепную реакцию перестройки природы. И если даже кто-то, включив этот триггер, сумеет отсидеться в каком-нибудь немыслимом убежище, выйдя из него, он найдет совсем другую планету, где ему уже не будет места. Из всего сказанного на слушаниях явственно следует одно: ядерному оружию больше невозможно приписывать какой бы то ни было военный или политический смысл: это оружие просто нельзя применять ни в каких целях, кроме самоубийства».

И еще долго на всех языках мира звучали слова глубокой признательности ученым, сумевшим доказать, что компьютерное прогнозирование может быть таким ясным и убедительным, таким необходимым для принятия стратегических решений.

В последние десятилетия экологической кибернетикой выявляется фундаментально позитивное влияние компьютеризации на экологию, выражающееся в постепенной трансформации и виртуализации процессов удовлетворения физиологических и духовных потребностей человечества, которые всегда имели определяющее влияние на формирование антропогенной нагрузки на биосферу [4, 5].

Категория индивидуальных и популяционных потребностей является одной из центральных в экологии. Удовлетворение потребности индивида — это переходный процесс из одного психического состояния (неудовлетворенности) к другому (состоянию удовлетворения). Переходные процессы от голода к его утолению, от замерзания к тепловому комфорту можно осуществить только в физическом пространстве с определенными затратами материальных и энергетических ресурсов планеты. Но возьмем другой пример: зоолог на определенном этапе работы осознает необходимость ознакомления с жизнью львиных прайдов. Он может отправиться в африканскую экспедицию с огромными затратами на транспорт, провизию, экипировку, риском для жизни, риском заболеть, оплатой страховки, нанесением дополнительного экологического вреда флоре и фауне и т.д. А может выйти на соответствующий сайт Интернета с текстовыми описаниями и видеофильмами по интересующей теме, не затратив никаких ресурсов, не нанеся никакого ущерба. Переходный процесс

состоялся в информационном пространстве и дал (возможно, гораздо больше) информации практически без материальных затрат.

Рассмотрим общество в целом. Пример общественной потребности: всем необходимы контакты с коллегами и партнерами, родственниками и друзьями. Каждодневные миллиарды переходных процессов по удовлетворению этой потребности могут реализовываться транспортной инфраструктурой с соответствующими затратами. А могут в значительной своей части — телекоммуникациями с несоизмеримо меньшими затратами, и сегодня — компьютерными телекоммуникациями с небывалыми возможностями и технико-экономической эффективностью. Компьютер — реальное средство ликвидации всего экологически негативного комплекса перенаселенных мегаполисов. Уже сегодня происходит «рассасывание» городской толчеи и автомобильных пробок за счет трансформации очных деловых контактов в компьютерные, замены офисного взаимодействия производственных коллективов во взаимодействие персональных домашних компьютеров сотрудников. Огромные возможности предоставляет развитие компьютерного взаимодействия и для расселения за пределы городов, экологическая обстановка в которых катастрофически усугубляется.

Другой пример экологизации посредством компьютеризации. Человечество много веков совершенствует способы записи, хранения и передачи информации. Сравнительно недавно мы жили в эпоху победного доминирования бумажных носителей. Сегодня эти функции во все нарастающих объеме и качестве исполнения берут на себя электронные информносители, развивающиеся фантастическими темпами. Для экологии это новая эра: спасены колоссальные лесные массивы, приговоренные к исчезновению целлюлозно-бумажной промышленностью.

Возьмем такую индивидуальную и общественную потребность, как повышение эффективности научных исследований. Ввиду ее бескрайности сосредоточимся на роли компьютера в задачах гидро- и аэродинамики, где методом описания процессов являются системы дифференциальных уравнений. Колоссальные аналитические трудности встретила наука, пытаясь решать системы этих уравнений с учетом реальных нелинейностей и нестационарностей многомерных объектов. Сложность, громоздкость, невозможность аналитического интегрирования широчайшего практически важного класса систем

дифференциальных уравнений способствовала созданию целых отраслей экспериментального физического моделирования. Компьютер с его неограниченными возможностями численного интегрирования (без каких бы то ни было ограничений по многомерности, нелинейности, нестационарности математических блоков) блестяще разрешил проблемы. Упразднены целые направления физического моделирования и экспериментирования, исключительно дорогостоящие и экологически вредные.

Мы привели примеры удовлетворения индивидуальных и общественных потребностей, не ставя под сомнение их важность. Однако тема шире. По мере развития цивилизации и экологической деградации планеты все актуальнее становится мировоззренческая дифференциация потребностей на действительные (первичные, функционально необходимые) и мнимые (искусственные, вторичные). Проиллюстрируем разницу на очевидном примере.

Человек купается в воде потому, что это полезно и приятно его телу. Особенно полезно и приятно мыться в подогретой воде, израсходовав энергию. Для нормального человека переход к подогреву объективно ценнее воды уличной температуры, допустим, на 50%. Хотя легендарные целители с этим бы не согласились, и были бы абсолютно правы. Затратив дополнительную энергию, можно перейти к гидромассажу, что ещё полезнее и приятнее и даст еще 10% физиологического эффекта. Наверное, 20% физиоэффекта и удовольствия даст нефальсифицированная бытовая химия. И больше ничего действительно полезного для организма не изобрести. В молоке и шампанском аристократ купается не потому, что это полезнее и приятнее физически (может быть в чем-то на 0,5%), а потому, что это не дано другим, потому, что «он этого достоин», потому, что это только «для тех, кто чувствует разницу». Реализуется нефизиологическая потребность социального превосходства. Более того, мизантропу приятно, что молоко отнято у других, голодных. Если бы можно было неограниченно пить, он бы выпил эти отнятые стаканы, непосредственно вливая в себя чужое здоровье и красоту. Но приходится выдумывать купание в молоке и тысячи других ничем естественным не обусловленных потребностей-развлечений.

Безысходность цивилизации в том, что одновременно растут и уровень планки индивидуально-потребительских «капризов», и численность особей. Сколько мог потреблять султан позапрошлого века? Сотню наложниц и сотню убитых тигров. Да, его роль в исчезновении

тигров значительна. И всё-таки это очень узкий фронт наступления на природу в сравнении с агрессией современного человека. Сегодня индивид хочет и может иметь горы движимых и недвижимых вещей. Их произвели с колоссальными затратами и отходами (ежегодно по 48 тонн отходов на каждого жителя Земли, считая и тех, кому кроме отходов ничего не достается). А яхта губернатора Чукотки с вертолётами и подводными лодками просто не могла присниться не то что султану позапрошлого века, а и миллиардеру прошлого. Современный человек хочет и может жить во дворце, напичканном инженерной инфраструктурой и бытовой супертехникой, поглощающими огромную энергию, производимыми опять же с колоссальными затратами и отходами и обновляющимися каждый год. Он хочет и может иметь сказочную кухню и сказочный гардероб, он хочет иметь все, что можно представить, и менять это каждый день. Таких людей на планете уже гораздо больше, чем султанов. Только миллиардеров 1162. И остальные 6 миллиардов мечтают ими стать именно для того, чтобы потреблять с тем же размахом. И нельзя не заметить, что компьютер в качестве информационно-оргтехнического средства эффективно участвует в развитии общества массового потребления. То есть играет роль экологически негативную. Однако есть основания полагать и существование противоположных тенденций. Рассмотрим их, опираясь для наглядности гипотез на символы и модели философского антропоморфизма.

Создатель (антропоморфистская фигура, символизирующая генезис и эволюцию человечества без привязки к какой-либо религиозной или естественнонаучной гипотезе) возвысил человека над животными, подарив ему Разум, который должен был стать пространством бескрайнего саморазвития. Но разум стал всего лишь средством изобретения избыточных потребностей. И в их горниле стремительно уничтожаются ресурсы планеты. И как будто ни чем не скорректировать эту гибельную для планеты траекторию.

Но есть серьезные основания полагать, что обнадеживающим регулятором становится именно Его Величество Компьютер, уводящий уже сегодня сотни миллионов землян в виртуальное пространство. Человечество стремительно и необратимо уходит туда двумя мощными потоками. Первый, как будто бы задуманный Создателем, — творцы, усиливающие компьютером свои способности и ускоряющие

свое движение по бескрайним просторам Разума. Второй, «обманувший Создателя», — ненасытные потребители земных благ, соблазненные теперь потреблять виртуально. С теми же ощущениями и даже изощреннее, но не нанося ущерба природе и обществу. И оживает непостижимая формула изнуряющих уроков марксизма-ленинизма: «От каждого по способностям, каждому по потребностям». В «Мертвом сезоне» бесчеловечный биолог мечтает создать породу счастливых людей-рабов, которым благодаря нейрохирургии и генной инженерии нужна только ежедневная миска похлебки. Компьютер никого не покалечит хирургией, не подавит ничьих желаний, никого не заставит работать. Он уведет всех в волшебный мир и провозгласит вожделенное: «На всех всего хватит».

И наступает эра, когда никому для утоления своей «охоты к перемене мест» не нужен автомобиль, теплоход и авиалайнер. Ньютоновское механическое перемещение по планете, наконец-то, названо примитивной формой движения. Пока экологи доказывали, что это расточительное потребление редких металлов, энергии, кошмар сжигания нефти и кислорода, программисты доказали, что это неэффективное, унизительное, немодное времяпровождение, отбрасываемое историей, как унылость дилижанса.

Уже вчера телевизор (усилиями развивающих программ) позволил путешествовать, не перемещаясь в пространстве, и многое снятое профессиональной камерой даже заядлый турист и охотник сами никогда не увидели бы. Какие сказочные подводные съемки, сколько ошеломляющего подсмотрено в жизни диких животных! Сколько нефти надо бы было потратить, чтобы всех телезрителей (а не группу зоологов) реально отправить в эти путешествия, и что осталось бы от этих экзотических мест? И неразвивающие программы тоже очень экологичны. Часами сидят миллионы телезрителей, надрываясь от смеха перед «Кривыми зеркалами», и не наносят никакого ущерба природе в процессе обретения полного удовлетворения жизнью. Илья Ильф иронизировал: «Вот уж и радио изобрели, а счастья всё нет». Сегодня без иронии следует продолжить: «И только с изобретением телевидения, компьютера. . . ».

В мире компьютерных фантазий особое место отведут любимой игре Homo sapiens — получению прибыли. Любите зарабатывать на эксплуатации человека человеком — пожалуйте в программу N. Обожаете изничтожать природные ресурсы — просим в N + 1. А какой

простор для жаждущих славы и творящих столько бед. Теперь честолюбцы триумфально побеждают на виртуальных выборах, деспотично вершат судьбы миллионов граждан виртуальных стран, штыками созидают и рушат виртуальные империи, снимают и снимаются в виртуальном Голливуде, виртуально выигрывают Олимпийские игры и бои без правил. Сколько гибельных для природы и общества процессов каждый день «улетучиваются» в виртуальное пространство. Особо позаботился компьютер об изоляции маньяков, которые, вместо того чтобы вредить реально, играют у монитора в киллеров и порнографических героев. И не исключено, что их атавистические выходы в реальный мир — досадные издержки недолгого переходного процесса к более совершенным компьютерным системам. В перспективе благодаря компьютеризации процессы утоления первичных потребностей будут оптимизированы по всем параметрам. Изобретенные цивилизацией вторичные потребности будут виртуально удовлетворены даже более полноценно, чем сейчас. Ведь поскольку они неестественны, то и утолить их в полной мере может только неограниченное виртуальное пространство. В нем, несомненно, эти желания и разовьются и утончатся, и новые возникнут.

И, возможно, когда-нибудь историк напишет, в каких примитивных формах человечество удовлетворяло свои потребности, каким убогим на протяжении веков эволюции было это удовлетворение, и какой страшный ущерб оно наносило планете. Он напишет, что компьютер не мог возникнуть, минуя интеллектуальные фазы изобретения кочегарки и дизеля, а значит, и того экологического вреда, который обусловила технократическая цивилизация. Напишет, что человечество предавалось океанским круизам и охоте на носорогов только для того, чтобы изобрести и совершенствовать ружья и пароходы — жалкие, но необходимые технические ступени к компьютеризации жизни, к райским вратам виртуального мира.

Не получилось у землян разумного материального потребления. Но вдруг Создатель добьется от нас экологической чистоты, заселяя нами царство Разумного потребления. Мы всегда понимали, что строить очистные сооружения — это экологически неоптимально. Оптимально — создавать безотходные технологии. Однако о такой радикальной безотходности, экологичности, которую обеспечило бы погружение человечества в виртуальную реальность, никто, конечно, не мечтал.

Но это уже другая планета. А может быть, так и замкнется круг? Ведь, наблюдая экологическую конфликтность человека на Земле, вполне логично поверить в его инопланетное происхождение.

Не развивая космологических и футурологических гипотез, можно утверждать, что Компьютер уже сегодня становится реальным Экологизатором жизни на Земле [4,5]. Системный анализ экологической функции компьютерных технологий и синтез систем управления компьютерными воздействиями на процессы экологизации должны стать одним из приоритетных направлений экологической кибернетики.

Работа выполнена при финансовой поддержке РГНФ, проект № 07-02-00043а.

Список литературы

[1] Бертокс П., Радд Д. Стратегия защиты окружающей среды от загрязнения. — М.: Мир, 1980. — 606 с. ↑(document)

[2] Медоуз Д.Х., Рендерс Й., Медоуз Д. За пределами роста. — М.: Прогресс-Пангея, 1994. — 304 с. ↑

[3] Моисеев Н.Н. Междисциплинарные исследования глобальных проблем. — М.: Тайдекс Ко, 2003. — 264 с. ↑(document)

[4] Гордин И.В. Игнорирование экологических угроз. — М.: Физматлит, 2007. — 120 с. ↑(document)

[5] Гордин И.В. Экология: проблемы и программы. — М.: Зеленый мир, 2008. — 112 с. ↑(document)

I. V. Gordin. *Society computerisation as the factor of the solution of ecological contradictions* // Proceedings of Program Systems institute scientific conference "Program systems: Theory and applications". — Pereslavl-Zalesskij, v. **1**, 2009. — p. 265–276. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. The ecological cybernetics actively studies a computerisation role in formation of anthropogenous pressure on biosphere. The set of requirements of mankind is satisfied today with use of computer technologies, decreasing negative ecological influences (from transport, a pulp and paper industry etc.). The satisfaction of many requirements passes in virtual space, completely excepting a damage for biosphere and planet resources. The computer helps to improve ecology of the Earth.

ISBN 978-5-901795-16-3 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ. Переславль-Залесский, 2009

УДК 517

И. В. Гордин

Неконтролируемость и неидентифицируемость загрязнения — ключевые категории экологической информатики

Аннотация. Мониторинг источников загрязнения — важнейшее направление экологической кибернетики и информатики. Однако значительная часть источников не контролируется. Программы природоохранных мероприятий оказываются сосредоточенными только на управлении теми потоками загрязнений, которые оказались под контролем. В результате эффективность программ существенно снижается. Проблема изучается на примере охраны водных ресурсов.

Универсальным принципом синтеза оптимальных систем управления на всех иерархических уровнях решения задачи устойчивого развития регионов и планеты в целом является надежность экологического мониторинга по всем значимым параметрам. Однако для ряда объектов не только отсутствуют системы надежного мониторинга, но неадекватны сами представления о необходимом и достаточном множестве координат, которые должны идентифицироваться и контролироваться. Рассмотрим эту проблему экологической информатики на примере систем управления качеством водных ресурсов.

Со второй половины прошлого века все развитые страны предприняли колоссальные усилия с целью радикального оздоровления экологической и санитарно-эпидемиологической ситуации на своих водных объектах. Пик водоохранного строительства в России пришелся на 1970–1980-е годы и ознаменовался выдающимися достижениями в деле сокращения сброса сточных вод за счет повсеместного строительства городских и поселковых очистных сооружений, мощных систем очистки и оборотного водоснабжения во всех отраслях промышленности, замкнутых систем водопользования для целых промрайонов с городскими жилыми массивами и прилегающими сельхозугодиями [1–3].

И, естественно, главный научный и практический интерес представляет конечная эффективность проводимых мероприятий, их фактическое влияние на качество воды в бассейнах. Однако даже в период наиболее интенсивного водоохранного строительства экологический мониторинг не фиксировал улучшения качества воды по целому ряду важнейших показателей. Как объяснение этого факта неэффективности колоссальных капиталовложений в гидроэкологии возникли три гипотезы, сыгравшие значительную роль в развитии природоохранной теории и практики [3], [4].

Первая гипотеза: строящиеся очистные сооружения по ряду ингредиентов неэффективны. Самым масштабным доказательством этого положения является неспособность станций механико-биохимической очистки, т.е. подавляющего большинства городских, поселковых и общезаводских очистных сооружений, задерживать азот и фосфор. При этом водоохранное строительство оказывается индифферентным к предельно актуальной проблеме эвтрофирования (цветения) водоемов.

Таблица 1. Эффективность автоматизированной системы доочистки

Загрязняющие вещества	**Вход системы**	**Выход системы**	
		Первая ступень	Вторая ступень
Взвешенные, мг/л	7,0-14,0	4,8-7,0	0,5-2,6
Органические, мг $БПК_{П}$/л	6,0-18,0	4,0-8,0	1,6-4,0
Микробное число, тыс.клеток/мл	36-85	14-23	6-9
Аммоний, мг/л	5,5-9,2	3,0-4,9	0,3-1,0
Фосфаты, мг/л	5,1-7,7	4,8-7,2	4,6-7,0

Практическим выводом первой гипотезы является необходимость совершенствования локальных, общезаводских, городских очистных сооружений (ГОС), строительство станций доочистки сточных вод, прошедших механико-биохимическую очистку. И на этом направлении достигнуты выдающиеся технологические результаты. В частности, в табл. 1 представлены характеристики одной из отечественных систем доочистки [1]. Данная автоматизированная установка эффективно удаляет из сточных вод взвешенные вещества, органику, патогенную микрофлору, аммоний. Фосфаты проходят установку практически беспрепятственно, и для их удаления система дополнена

специальным автоматизированным блоком физико-химической обработки.

Радикальным решением экологических проблем полагается создание оборотных систем водоснабжения и замкнутых систем водопользования промрайонов на базе повторного использования в промышленном и сельскохозяйственном производстве кондиционированных городских и промышленных сточных вод.

Вторая гипотеза: действующая система мониторинга не способна зафиксировать все изменения, которые происходят в бассейнах под влиянием негативных и позитивных антропогенных воздействий. С этим нельзя не согласиться, когда из тысяч примесей нормируются только сотни и только 20–30 из них надежно измеряются. Причем пространственно-временная сетка этих измерений назначается не из соображений регистрации тех или иных переходных процессов по теореме Котельникова, а исключительно из возможностей химлабораторий.

Практический вывод: необходимость создания технически совершенных систем мониторинга, позволяющих всесторонне отследить и проанализировать последствия проводимых водоохранных мероприятий.

Надо заметить, что если в технологических проработках (включая и аспект автоматического регулирования технологических процессов) мы долгое время соперничали с развитыми странами, то в области гидроэкологического мониторинга отставание всегда было явным. И главная причина - аналитическая база. Уже начиная с 1970-х годов США надежно идентифицировали в природных водах более 1500 органических соединений.

Третья гипотеза, разрабатываемая с позиций экологической информатики, главной причиной неэффективности природоохранных программ считала неадекватность их исходной информационной базы. И в первую очередь, неполноту блока «Источники загрязнения» [1], [4].

Источники загрязнения всегда были одним из главных предметов теоретической и практической экологии. Многообразие источников требовало самых разных классификационных признаков: гидравлическая и массовая мощность; физико-химический и биохимический

состав стока; принадлежность к генерирующей отрасли, предприятию; степень токсикологической, санитарной, эпидемической, экологической опасности; точечность, канализованность, распределенность, диффузность; стационарность (временная), нестационарность, стационарность (пространственная), мобильность, штатность, аварийность и т.д.

Но системный анализ показывал, что для объяснения сложившейся в водоохранном проектировании и строительстве ситуации в названное классификационное пространство необходимо ввести принципиально новые координаты из области информатики. При определении источников загрязнения бассейна разработчики водоохранных проектов используют известную форму 2тп-водхоз, достаточно детально учитывающую контролируемые канализационные выпуски. При этом действия неконтролируемых источников загрязнения бассейна как бы не существуют. Нарушаются важнейшие положения системного анализа (определение полной группы событий, определение полного вектора возмущающих воздействий).

Опираясь на общие положения кибернетики и информатики, третья гипотеза не является вместе с тем теоретической абстракцией. Она разрабатывается по мере все более детального экспериментального изучения невидимых, неучитываемых поступлений загрязняющих веществ в природную гидросферу (и канализованных, и диффузных). Приведем два эпизода анализа этих латентных процессов, которыми автор занимался в Кузбассе в докризисный период.

На заводе «Электромашина» (Прокопьевск) синтезировалась автоматизированная станция нейтрализации сточных вод гальваноцеха. Здесь возникали интересные кибернетические задачи управления потоками. Кислотные стоки использовались взамен товарных реагентов в реакторах восстановления шестивалентного хрома бисульфитом натрия (оптимум проведения реакции pH = 2,5), щелочные направлялись в реакторы окисления цианидов (оптимум проведения реакции pH = 11,5) и т.д. Создание таких АСУ ТП является одной из самых творческих задач автоматического регулирования процессов очистки сточных вод.

В ходе исследования системы водоотведения в карту экспериментального контроля наряду с производственными выпусками был

включен и замыкающий колодец неконтролируемой ливневой канализации промплощадки. Концентрации металлов колебались примерно одинаково как в производственных сточных водах, так и в условно-чистом стоке ливневой канализации. Выделялись только аварийные прорывы технологического раствора меди через автоматизированный электрокоагулятор. По ХПК, БПК, нефтепродуктам ливневой сток оказался несоизмеримо хуже производственного. В то время как максимумы нефтезагрязненности производственных сточных вод не поднимались выше 5-6 мг/л, в ливневой канализации зафиксированы концентрации до 9600 мг/л. Первым потоком предстояло заняться на базе АСУ ТП, чтобы сделать его еще менее опасным для природной гидросферы. Вторым вообще не надо заниматься, поскольку он проходит в отчетности 2тп-водхоз по графе «условно-чистый». Эти локальные сопоставления заставили серьезно усомниться вообще в правильности путей, на которых техническая кибернетика ищет выход из гидроэкологического кризиса.

Другой промплощадкой исследования феномена неконтролируемого загрязнения стал Томь-Усинский ремонтно-механический завод (Междуреченск). Территория завода занимает 28,5 га, в том числе: застройка 5 га, твердые покрытия 0,7 га, зеленые насаждения 1,5 га. Источники водоснабжения предприятия: городской водопровод — 68 тыс.м^3/год, технический водозабор из Томи — 80 тыс.м^3/год. Сброс хозяйственно-бытовых сточных вод в городскую канализацию — 57 тыс. м^3/год, производственных на заводские очистные сооружения — 30 тыс.м^3/год. Таким образом, безвозвратные потери составляют 61 тыс.м^3/год, т.е. 41% общего водопотребления.

Такой водохозяйственный баланс вызвал серьезные сомнения. Были проведены детальные расчеты реальных потерь воды. Потери на испарение и капельный унос в градирнях оценены в 7,5 тыс.м^3/год, расход на полив территории и зеленых насаждений 3 тыс.м^3/год, прочие расходы 0,5 тыс.м^3/год. Итого 11, но никак не 61 тыс.м^3/год. К неизвестно куда девающимся 50 тыс.м^3/год сточных вод необходимо прибавить как минимум 65 тыс.м^3/год тало-дождевых вод, формирующихся на площадке. Таким образом, из 202 тыс.м^3/год годового объема водоотведения 30 тыс.м^3/год реально контролируются и очищаются на заводских сооружениях, 57 тыс.м^3/год контролируются только количественно (перекачиваясь на ГОС) и 115 тыс.м3/год (57%) оказываются абсолютно неконтролируемыми. Как они уходят в природную гидросферу и сколько привносят в нее загрязняющих примесей, неизвестно.

Графики загрязненности выходящих с завода потоков изображены на рис. 1.

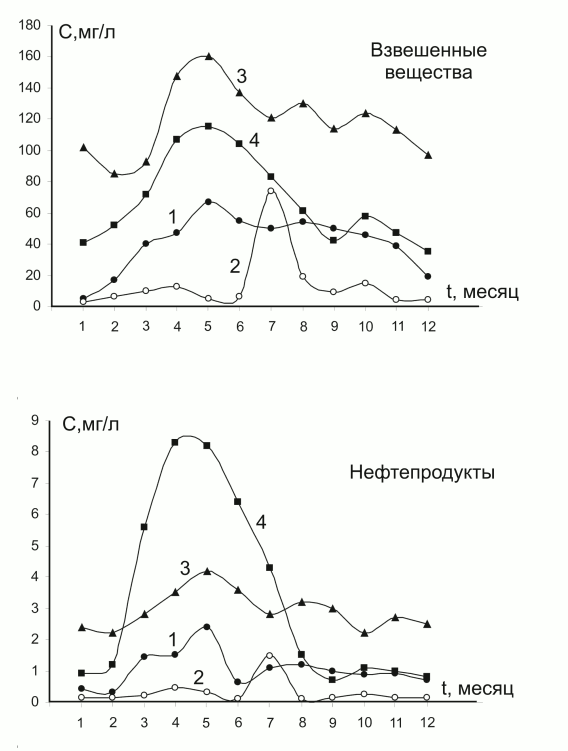


Рис. 1. Динамика среднемесячных концентраций загрязняющих веществ в сточных водах Томь–Усинского завода: 1 — производственный сток, 2 — очищенный производственный сток, 3 — хозяйственно-бытовой сток, 4 — сток дренажного канала-коллектора

Сезонный ход концентраций взвешенных и нефтепродуктов говорит о несомненном проникновении в производственную канализацию талых и дождевых смывов промплощадки. Свой вклад вносит пост мойки автотранспорта, особенно загрязняемого в весеннюю распутицу. Достаточно эффективная очистка в условиях весенней перегрузки

дается ценой избыточного зашламовывания грязеотстойника и флотаторов, забивания фильтров. В работоспособное состояние систему возвращает июльская промывка, характеризующаяся резким всплеском выходных концентраций. Менее заметный, но тоже зафиксированный след оставляет осенняя и летне-осенняя грязь площадки и в хозяйственно-бытовой канализации. Спецификой хозбытового стока завода являются высокие концентрации нефтепродуктов, не снижающиеся ниже 2 мг/л. Это объясняется изобилием ГСМ в цехах и на территории завода, интенсивной работой моек, душевых кабин, прачечной спецодежды. Весной резко нарастает загрязненность стока от мойки полов в цехах и бытовых помещениях. Главным путем проникновения грязи в хозбытовую канализацию является вынужденное открытие ее колодцев для ликвидации подтопления территории. В колодцы хозяйственно-бытовой и производственной канализации уходит около 20 тыс.м^3 годового объема тало-дождевых вод. Значит, остается неясной судьба 95 тыс.м^3 образующегося в промзоне стока.

Ответ на этот вопрос дали специально организованные круглогодичные измерения загрязненности дренажного канала-коллектора завода. Как и большинство дренажных систем, канал полагается условно-чистым, пр опускающим 130 тыс.м^3/год фильтруемых в него природных болотных вод. Нет никаких сомнений, что именно через этот канал в Томь уходят ненайденные 95 тыс.м^3/год сточных вод завода, включая основной объем поверхностного смыва с территории.

Подведем итоги действительного водохозяйственного баланса. Контролируемо с завода уходят только 30 тыс.м^3/год очищенных производственных сточных вод с остаточной, официально фигурирующей в отчетах 2тп-водхоз, концентрацией после сорбционных фильтров 5-10 мг/л по взвешенным и 0,1-0,2 мг/л по нефтепродуктам. Всего за год этот официальный сброс составляет по самой завышенной оценке 300 кг взвеси и 6 кг нефтепродуктов. Реально же с учетом тало-дождевой перегрузки имеем 420 кг и 9 кг соответственно. Но это мелкие погрешности технологического контроля в сравнении с тем, что в городскую канализацию уходит неконтролируемо 6,7 т взвеси и 165 кг нефтепродуктов. Этот поток осложняет работу ГОС, но все-таки он подпадает под контроль Росприроднадзора, хотя бы на этом «обезличенном» этапе. Воды дренажного канала поступают в Томь без всякого контроля. А это еще 8,8 т взвешенных и 429 кг нефтепродуктов. Ее нетрудно рассчитать. Степень бесконтрольности процесса генерации взвешенных примесей: $1 - 0{,}3/(0{,}42 + 6{,}7 + 8{,}8) = 1 -$

0,3/15,9 = 98%. По нефтепродуктам степень неконтролируемости источника загрязнения равна: 1 — 6/(9 + 165 + 429) = 1 — 6/603 = 99%. Эти объективные соотношения не попадают в поле зрения водохозяйственных и природоохранных органов, Росприроднадзор полагает, что он контролирует ситуацию и способен к эффективным управляющим воздействиям.

Современную остаточную загрязненность сточных вод Кузбасса, контролируемо выходящих с ГОС, и загрязненность малых рек, протекающих в городской черте и являющихся фактически коллекторами неконтролируемых сбросов, отражает рис. 2 ([1]).

Загрязняющие вещества		г. Новокузнецк		г. Кемерово	
		выход ГОС	р. Аба	выход ГОС	р. Искитимка
Взвешенные		10,7	32	12,5	17,6
ХПК		20,8	55,9	15,7	22,2
Азот	аммонийный	1,3	1,54	0,24	0,86
	нитритный	0,14	0,11	0,15	1,6
	нитратный	15,6	10,5	18,7	1,3
Фосфаты		1,3	0,6	5,5	2,7
Железо		0,29	0,84	0,15	0,82
Медь		0	0,014	0	0,015
Нефтепродукты		0,09	1,25	0,08	0,72
Фенолы		0,001	0,008	0,003	0,004
СПАВ		0,15	1,6	0,15	1,1

Рис. 2. Концентрации загрязняющих веществ в очищенных сточных водах и в малых реках, протекающих через города Кузбасса

Как видим, по большинству показателей сток ГОС значительно чище рек, в которые он сбрасывается. В этой экологически тупиковой ситуации глубокая очистка стока ГОС ошибочно считается Росприроднадзором стратегическим достижением.

Повышенная загрязненность малых рек в черте промышленных городов России повсеместно формируется в результате функционирования санкционированных и несанкционированных прямых канализационных выпусков с предприятий, не имеющих локальных очистных сооружений, скрытых врезок производственных сточных вод в

ливневку, аварийных технологических сбросов, ночных сбросов жидких отходов из накопителей суточного регулирования и т.д. И даже если в каком-нибудь городе не останется никакой промышленности, что для ряда городов с 2009 года становится реальной перспективой, то, несомненно, останутся стационарные утечки и аварии хозяйственно-бытовой канализации, которые также создают дополнительное неконтролируемое загрязнение водных объектов. И, конечно, всегда присутствует поверхностный, диффузный смыв с территорий.

Много усилий было предпринято для привлечения внимания к проблеме неконтролируемого загрязнения и его влияния на конечную эффективность водоохранных программ и систем управления состоянием водных ресурсов. Однако, когда феномен неконтролируемого загрязнения получил всеобщее признание, былое невнимание к нему стало восприниматься как экологический нонсенс. Поэтому целесообразно дать комментарии, уточняющие историю вопроса.

Во-первых, отнесение на протяжении десятилетий некоторых классов сточных вод к условно-чистым не следует считать однозначным признаком невнимания. Скорее, это интуитивное формирование водоохранной стратегии с оптимальной очередностью мероприятий. Первоочередные — хозяйственно-бытовые и производственные сточные воды, перспективные — ливневая канализация, термальные воды теплоэнергетики, сбросные воды ирригационных систем и др. Надо помнить, что игнорирование проблемы зарождалось на фоне колоссального канализованного сброса абсолютно неочищаемых сточных вод городов и промрайонов.

Во-вторых, многие неконтролируемые источники загрязнения не игнорировались, а считались временным явлением, которое будет ликвидировано по тем же схемам, что и канализованные потоки. Так, экологическим кошмаром стала гигантомания животноводческих комплексов, приведшая к беспрецедентным диффузным смывам в гидросферу. Но эта опасность возникла не вследствие игнорирования проблемы, а, напротив, как попытка ухода от антиэкологичности малых ферм. Разработчики проектов были уверены, что высокотехнологичная очистка и утилизация сконцентрированных отходов крупных животноводческих комплексов — вполне решаемая задача.

В-третьих, хотя концептуально проблема неконтролируемого загрязнения загонялась в тень, водоохранная практика немало делала для ее решения. Во всяком случае, больше, чем сейчас, когда все

теоретически прояснено и практически доказано, но разрушена технологическая база решений.

Эколого-экономической драмой рассматриваемой фазы развития природоохранной стратегии явилось то обстоятельство, что водоохранные программы по инерции ориентировались на «ужас» канализованных и контролируемых сточных вод, а качество воды в бассейнах уже определяли неконтролируемые, главным образом диффузные, источники.

Казалось, что наметившаяся тенденция в дальнейшем будет только усугубляться вследствие прогресса в очистке сосредоточенных потоков и неуклонного расширения диффузно загрязняемых территорий, особенно в условиях все более «небрежной» разработки нефтегазовых месторождений Сибири. Промышленный спад 1990-х сделал эту тенденцию еще более выраженной, сельскохозяйственный — несколько сгладил.

Но с началом некоторого производственного подъема 2000-х годов вследствие обвального разрушения очистных сооружений лидерство опять захватил канализованный производственный и хозяйственно-бытовой сток, ввергая нас в пучину экологического кризиса образца середины прошлого века, когда потоки неочищенных хозяйственнобытовых и производственных сточных вод прямо из канализационных труб сливались в природные водоприемники. Промышленные системы очистки при дополнительных ремонтных затратах как-то продержались только за счет пониженной нагрузки. Но городские, поселковые очистные сооружения, нагрузка на которые не снижалась в период производственного спада 1990-х годов и продолжает увеличиваться, находятся в состоянии критическом.

Эту фазу может сменить еще более опасная. Начинается она с того, что типовые канализованные потоки в отличие от своих аналогов прошлого века уже никак нельзя считать надежно контролируемыми ни по расходу, ни по составу сточных вод. Одним из ярчайших примеров новейшего вида неконтролируемого загрязнения водных объектов, включая источники централизованного водоснабжения, является потайной канализованный сброс массовой дачно-коттеджной застройки [1], [2], [5].

Ситуация усугубляется тем, что из-за физического износа и разрушения не только очистных сооружений, но и подводяще-отводящих сетей непрерывно увеличивающуюся часть потоков уже нельзя считать и канализованными. К великому сожалению, мы наблюдаем, как

целые классы канализованных сточных вод, ранее надежно утилизируемых или глубоко очищаемых перед сбросом, во всяком случае надежно контролируемых, присоединяются к экологически коварной категории неконтролируемых источников загрязнения.

Сегодня экологическую ситуацию «поддерживает на плаву» усугубление общего экономического кризиса. Однако по опыту 1990-х гг. можно уверенно сказать, что экологически позитивное снижение производственной нагрузки будет компенсироваться нарастанием бесконтрольности сброса отходов оставшимися производствами и жилищно-коммунальным хозяйством.

Несмотря на официальное признание третьей гипотезы научной общественностью и природоохранными ведомствами, до сих пор встречаются отдельные попытки занижения научно-методологического значения информационных категорий «контролируемость-неконтролируемость», попытки отождествления их с физическим признаком «точечность-диффузность». Некоторые специалисты видят методологическую плодотворность информационного подхода только в том, что к одному классу («неконтролируемые») отнесены все исходно распределенные потоки независимо от степени их последующей канализации. Другие полагают, что категории информатики полезны только тем, что содействует обоснованию мониторинговых разработок, компьютеризации природоохранных технологий. В этих оценках не улавливается главный смысл перехода к информационному признаку классификации, к классификации по признаку влияния на результативность природоохранных программ через их исходную информационную базу. А этот аспект следует считать важнейшим, принципиальным ровно настолько, насколько принципиален вопрос о конечной эффективности природоохранных мероприятий, об их реальном влиянии на качество окружающей среды.

Сегодня, когда природоохранное строительство недопустимо свернуто, как будто бы не время размышлять об оптимизации проектирования. Однако даже в этих условиях принцип системной оценки всей совокупности источников загрязнения должен быть основополагающим принципом разработки программ строительства и синтеза систем управления. При этом тенденция снижения информационной обеспеченности неизбежно заставляет переходить к укрупненным, вариантным, вероятностным расчетам и алгоритмам управления. Но ни

в коем случае не к детальным проработкам и регулирующим воздействиям по информационно обеспеченным блокам с игнорированием системного взгляда на формирование антропогенной нагрузки.

Экология должна распрощаться со стереотипом «потерянного кошелька»: «теряем там, где темно» (загрязняем биосферу неконтролируемыми, неидентифицируемыми, латентными потоками), «а ищем, где светло» (пытаемся выправить ситуацию все более совершенными решениями в области технологии очистки и утилизации контролируемых потоков загрязнения).

Работа выполнена при финансовой поддержке РГНФ, проект № 07-02-00043а.

Список литературы

[1] Гордин И.В. Кризис водоохранных зон России. — М.: Физматлит, 2006. — 196 с. ↑(document)

[2] Гордин И.В. *Современная динамика загрязнения окружающей среды* // Экономика природопользования, № 3, 2003, с. 26-34. ↑(document)

[3] Гордин И.В. *Фазы развития и актуальные проблемы эколого-экономической оптимизации водоохранной системы РФ* // Экономика и математические методы, № 4, 2003, с. 17-27. ↑(document)

[4] Гордин И.В. *Методологическое значение классификации источников загрязнения биосферы по информационному признаку* // Вестник IASS «Информатика, экология, экономика», № 1, 2001, с. 12-18. ↑(document)

[5] Гордин И.В. *Система мониторинга процесса застройки водоохранных зон* // Вестник IASS «Информатика, экология, экономика», № 1, 2007, с. 48-52. ↑(document)

I. V. Gordin. *Uncontrolled and unidentificated pollution — key categories of ecological computer science* // Proceedings of Program Systems institute scientific conference “Program systems: Theory and applications”. — Pereslavl-Zalesskij, v. **1**, 2009. — p. 277–288. — ISBN 978-5-901795-16-3 (*in Russian*).

Abstract. Monitoring of sources of pollution — the major direction of ecological cybernetics and computer science. However the considerable part of sources is not supervised. Programs of nature protection actions appear those streams of pollution concentrated only on management which have appeared under the control. As a result efficiency of programs essentially decreases. The problem is studied on an example of protection of water resources.

Author Index

Esin, Grigoriy Igorevich
Research Center for Multiprocessor Sysems PSI RAS
grisha@skif.botik.ru (*v.* **1**: 225–243)

Fralenko, Vitaliy Petrovich
Artificial Intelligence Research Center PSI RAS
alarmod@pereslavl.ru (*v.* **1**: 025–041, 133–143)

Gorbunov, Pavel A.
Medical Informatics Research Center PSI RAS
gorpa@vologda.cbr.ru (*v.* **2**: 121–131)

Gordin, Igor V.
Control Processes Research Center PSI RAS
ivgordin@mail.ru (*v.* **1**: 265–276, 277–288)

Grishina, Maria Alexandrovna
Chelyabinsk State University
maria˙grishina@csu.ru (*v.* **1**: 217–223)

Guliev, Azer. Yadullaevich.
Medical Informatics Research Center PSI RAS
Azer˙Guliev@UniverLab.ru (*v.* **2**: 165–174)

Guliev, Yadulla Iman-Ogly
Medical Informatics Research Center PSI RAS
upis@yag.botik.ru
(*v.* **2**: 013–025, 027–036, 071–096, 121–131, 145–163, 175–206)

Gulieva, Irina Fashitdinovna
Medical Informatics Research Center PSI RAS
upis@irina.botik.ru (*v.* **2**: 133–143)

Gurman, Vladimir Iosiphovich
Control Processes Research Center PSI RAS
vlad@head.botik.ru (*v.* **1**: 025–041)

Ivshina, Nadezhda Nikolaevna
Postovsky Institute of Organic Synthesis of Ural Branch of RAS
inn@ios.uran.ru (*v.* **1**: 217–223)

Magsumov, Dmitriy Rustemovich
Medical Informatics Research Center PSI RAS
dimam@interin.ru (*v.* **2**: 107–120, 217–226, 277–298)

Mashtakov, Alexei Pavlovich
Control Processes Research Center PSI RAS
alexey.mashtakov@gmail.com (*v.* **1**: 005–023)

Matveev, German Anatol'evich
Research Center for Multiprocessor Sysems PSI RAS
gera@prime.botik.ru (*v.* **1**: 217–223)

Miheev, Alexandr Evgen'evich
Medical Informatics Research Center PSI RAS
alexander@medcenter.msk.ru (*v.* **2**: 121–131)

Morzhin, Oleg Vasilievich
Control Processes Research Center PSI RAS
oleg˙morzhin@yahoo.com (*v.* **1**: 043–058)

Moskovskiy, Alexander Alexandrovich
Research Center for Multiprocessor Sysems PSI RAS
moskov@lcc.chem.msu.ru (*v.* **1**: 193–216)

Nazarenko, Gerasim Igorevich
Medical Informatics Research Center PSI RAS
general@medcenter.msk.ru (*v.* **2**: 121–131)

Nemytykh, Andrei Petrovich
Research Center for Multiprocessor Sysems PSI RAS
nemytykh@math.botik.ru (*v.* **1**: 245–264)

Pfaf, Vctor Fransovich
The Central Clinic Hospital of Russian Railways
V.Pfaf@ckb.rzd.ru (*v.* **2**: 027–036)

Pinzhin, Alexey Evgenyevich
Tomsk Polytechnic University
alex˙pinjin@tpu.ru (*v.* **1**: 145–152)

Pogosov, Alexey Olegovich
Medical Informatics Research Center PSI RAS
alexeypogosov@yandex.ru (*v.* **2**: 259–276)

Авторский указатель

Блинов, Александр Олегович
Исследовательский центр процессов управления ИПС РАН
sarmat@pereslavl.ru (т. **1**: 025–041)

Горбунов,Павел Александрович
Исследовательский центр медицинской информатики ИПС РАН
gorpa@vologda.cbr.ru (т. **2**: 121–131)

Гордин, Игорь Викторович
Исследовательский центр процессов управления ИПС РАН
ivgordin@mail.ru (т. **1**: 265–276, 277–288)

Гришина, Мария Александровна
Челябинский Государственный Университет
maria_grishina@csu.ru (т. **1**: 217–223)

Гулиев, Азер Ядуллаевич
Исследовательский центр медицинской информатики ИПС РАН
Azer_Guliev@UniverLab.ru (т. **2**: 165–174)

Гулиев, Ядулла Иман-Оглы
Исследовательский центр медицинской информатики ИПС РАН
upis@yag.botik.ru
(т. **2**: 013–025, 027–036, 071–096, 121–131, 145–163, 175–206)

Гулиева, Ирина Фасхитдиновна
Исследовательский центр медицинской информатики ИПС РАН
upis@irina.botik.ru (т. **2**: 133–143)

Гурман, Владимир Иосифович
Исследовательский центр процессов управления ИПС РАН
vlad@head.botik.ru (т. **1**: 025–041)

Емельянова, Юлия Геннадиевна
Исследовательский центр искусственного интеллекта ИПС РАН
tajra@mail.ru (т. **1**: 133–143)

Есин, Григорий Игоревич
Исследовательский центр мультипроцессорных систем ИПС РАН
grisha@skif.botik.ru (*т.* **1**: 225–243)

Заднепровский, Вадим Федорович
ИПС имени А. К. Айламазяна РАН
v.f.z_skii@mail.ru (*т.* **1**: 193–216)

Знаменский, Сергей Витальевич
Исследовательский центр системного анализа ИПС РАН
svz@latex.pereslavl.ru (*т.* **1**: 123–132)

Зубов, Дмитрий Владимирович
Московский государственный университет инженерной экологии
zubov@mguie.ru (*т.* **2**: 207–216)

Ившина, Надежда Николаевна
Институт Органического Синтеза им. И. Я. Постовского УрО РАН
inn@ios.uran.ru (*т.* **1**: 217–223)

Казаков, Владимир Александрович
Исследовательский центр системного анализа ИПС РАН
kazakov@svp.polnet.botik.ru (*т.* **1**: 085–103)

Казаков, Илья Федорович
Исследовательский центр медицинской информатики ИПС РАН
kazakov@interin.ru (*т.* **2**: 107–120, 217–226)

Козадой, Юрий Владимирович
Исследовательский центр медицинской информатики ИПС РАН
watergad@interin.ru (*т.* **2**: 227–240)

Комаров, Сергей Иванович
Исследовательский центр медицинской информатики ИПС РАН
ksi@interin.ru (*т.* **2**: 013–025, 027–036)

Кузнецов, Антон Александрович
Исследовательский центр мультипроцессорных систем ИПС РАН
tonic@pereslavl.ru (т. **1**: 225–243)

Куликов, Дмитрий Евгеньевич
Исследовательский центр медицинской информатики ИПС РАН
kulikov@interin.ru (т. **2**: 097–106, 241–257)

Лебедев, Алексей Викторович
ЦКБ №1 ОАО РЖД
A.Lebedev@ckb.rzd.ru (т. **2**: 027–036)

Лисица, Алексей Петрович
Университет Ливерпуля
A.Lisitsa@liverpool.ac.uk (т. **1**: 245–264)

Магсумов, Дмитрий Рустэмович
Исследовательский центр медицинской информатики ИПС РАН
dimam@interin.ru (т. **2**: 107–120, 217–226, 277–298)

Матвеев, Герман Анатольевич
Исследовательский центр мультипроцессорных систем ИПС РАН
gera@prime.botik.ru (т. **1**: 217–223)

Маштаков, Алексей Павлович
Исследовательский центр процессов управления ИПС РАН
alexey.mashtakov@gmail.com (т. **1**: 005–023)

Михеев,Александр Евгеньевич
Исследовательский центр медицинской информатики ИПС РАН
alexander@medcenter.msk.ru (т. **2**: 121–131)

Моржин, Олег Васильевич
Исследовательский центр процессов управления ИПС РАН
oleg_morzhin@yahoo.com (т. **1**: 043–058)

Московский, Александр Александрович
Исследовательский центр мультипроцессорных систем ИПС РАН
moskov@lcc.chem.msu.ru (т. **1**: 193–216)

Назаренко,Герасим Игоревич
Исследовательский центр медицинской информатики ИПС РАН
general@medcenter.msk.ru (т. **2**: 121–131)

Немытых, Андрей Петрович
Исследовательский центр мультипроцессорных систем ИПС РАН
nemytykh@math.botik.ru (т. **1**: 245–264)

Пинжин, Алексей Евгеньевич
Томский Политехнический Университет
alex_pinjin@tpu.ru (т. **1**: 145–152)

Погосов, Алексей Олегович
Исследовательский центр медицинской информатики ИПС РАН
alexeypogosov@yandex.ru (т. **2**: 259–276)

Потемкин, Владимир Александрович
Челябинский Государственный Университет
pva@csu.ru (т. **1**: 217–223)

Пфаф, Виктор Франсович
ЦКБ №1 ОАО РЖД
V.Pfaf@ckb.rzd.ru (т. **2**: 027–036)

Роганов, Владимир Александрович
Исследовательский центр мультипроцессорных систем ИПС РАН
var@pereslavl.ru (т. **1**: 225–243)

Рюмина, Елена Викторовна
Исследовательский центр мультипроцессорных систем ИПС РАН
ryum50@mail.ru (т. **2**: 133–143)

Сачков, Юрий Леонидович
Исследовательский центр процессов управления ИПС РАН
sachkov@sys.botik.ru (т. **1**: 005–023)

Сачкова, Елена Федоровна
Исследовательский центр процессов управления ИПС РАН
elenas@u-pereslavl.botik.ru (т. **1**: 059–075)

Сергеева, Алла Владимировна
Московский государственный университет инженерной экологии
wirt@mguie.ru (т. **2**: 207–216)

Талалаев, Александр Анатольевич
Исследовательский центр искусственного интеллекта ИПС РАН
arts@arts.botik.ru (т. **1**: 133–143)

Толчёнов, Алексей Андреевич
Московский государственный университет инженерной экологии
wirt@mguie.ru (т. **2**: 207–216)

Трушкова, Екатерина Александровна
Исследовательский центр процессов управления ИПС РАН
katerina@tea.pereslavl.ru (т. **1**: 025–041)

Фохт, Ольга Анатольевна
Исследовательский центр медицинской информатики ИПС РАН
olya@interin.ru (т. **2**: 175–206)

Фохт, Игорь Анатольевич
Исследовательский центр медицинской информатики ИПС РАН
vogt@interin.ru (т. **2**: 175–206)

Фраленко, Виталий Петрович
Исследовательский центр искусственного интеллекта ИПС РАН
alarmod@pereslavl.ru (т. **1**: 025–041, 133–143)

Хаткевич, Марк Иванович
Исследовательский центр медицинской информатики ИПС РАН
mark@interin.ru (т. **2**: 121–131)

Хаткевич, Юрий Иванович
Исследовательский центр медицинской информатики ИПС РАН
yuriy@interin.ru (т. **2**: 055–070)

Хачумов, Вячеслав Михайлович
Институт системного анализа РАН
vmh48@mail.ru (т. **1**: 133–143)

Цирлин, Анатолий Михайлович
Исследовательский центр системного анализа ИПС РАН
tsirlin@sarc.botik.ru (т. **1**: 085–103)

Шмелев, Алексей Борисович
Исследовательский центр мультипроцессорных систем ИПС РАН
alexey.shmelev@rsk-skif.ru (т. **1**: 193–216)

Юмагужин, Валерий Афтахович
Исследовательский центр системного анализа ИПС РАН
yuma@diffiety.botik.ru (т. **1**: 105–121)

Юмагужина, Валерия Николаевна
Исследовательский центр системного анализа ИПС РАН
course@u.pereslavl.ru (т. **1**: 105–121)

Contents of volume 1

System server software and parallel computing systems

Socio-ecological-economic systems modeling

Содержание тома 1

Интеллектуальные Интернет–технологии

Системное программное обеспечение вычислительных серверов и параллельные вычислительные системы

Моделирование социо-эколого-экономических систем

Contents of volume 2

Large Information Systems

Содержание тома 2

Большие информационные системы