Федеральное агенство по образованию
Государственное образовательное учреждение
высшего профессионального образования
Петрозаводский государственный университет
Математический факультет
Кафедра Информатики и математического обеспечения

Кулаков Кирилл Александрович

Генерация систем однородных неотрицательных линейных диофантовых уравнений и ее приложения

510210 — Программное обеспечение вычислительных сетей

Магистерская диссертация

Научные руководители:
к.т.н., доцент
Ю. А. Богоявленский
к.ф.-м.н., ст. преподаватель
Д. Ж. Корзун

Петрозаводск — 2005

Аннотация

Магистерская диссертация направлена на исследование задачи генерации систем однородных неотрицательных линейных диофантовых уравнений (системы одНЛДУ) и ее приложений для тестирования и экспериментального анализа реализаций алгоритмов решения систем одНЛДУ. Исследование сконцентрировано на частном классе систем одНЛДУ — ассоциированных с формальными грамматиками (системы одАНЛДУ).

В работе представлено исследование систем одАНЛДУ. Ключевым результатом является теорема о преобразовании произвольной системы одАНЛДУ. Преобразование существенным образом основано на определенном и исследованном нами частном классе систем одАНЛДУ — симметричные системы одАНЛДУ (системы содАНЛДУ).

Данное преобразование положено в основу пяти предлагаемых нами алгоритмов генерации систем одАНЛДУ, включая и генерацию систем содАНЛДУ. Преобразование также можно использовать и для построения алгоритмов (двойственных) решения систем одАНЛДУ. В этом направлении нами получены два алгоритма: для решения систем содАНЛДУ и для произвольной системы одАНЛДУ.

Практическая часть работы ориентирована на разработку технологии и программного инструментария для автоматизации тестирования, экспериментального и сравнительного анализа реализаций алгоритмов решения систем одАНЛДУ. В качестве базовой реализации алгоритма решения для выполнения тестирования и экспериментов был взят синтаксический алгоритм syntactic solver. Альтернативной реализацией для сравнительного анализа выступала реализация алгоритма slopes.

В работе предложены способы оценки потребляемых ресурсов, таких как затрачиваемое время и используемая память, разработана мобильная программная система alg_analyser для выполнения тестирования, экспериментального и сравнительного анализа алгоритмов решения систем одАНЛДУ. На ее основе были решены поставленные прикладные задачи тестирования и экспериментального анализа предложенной реализации syntactic solver и сравнительный анализ с реализацией slopes. Система alg_analyser также составила ядро алгоритмической части веб-ориентированной системы Web-SynDic, предназначенную для удаленной демонстрации и тестирования алгоритмов решения НЛДУ.

Оглавление

Перечень сокращений и условных обозначений

АНЛДУ система — ассоциированная с грамматикой система НЛДУ.

НЛДУ — неотрицательное линейное диофантово уравнение.

одАНЛДУ система — однородная система АНЛДУ.

одНЛДУ — однородное уравнение НЛДУ.

ПО — програмное обеспечение.

содАНЛДУ система — симметричная система одАНЛДУ.

$\varnothing$ — пустое множество.

$\mathbb{O}$ — нулевой вектор, $\mathbb{O} = (0, 0, \ldots, 0)$.

$\mathrm{E}^m(I^n)$ — булева $(n \times m)$-матрица разбиения, коэффициенты которой определяются как
$$\mathrm{E}^m_{kj}(I^n) = \mathrm{E}^m_{kj}(I_1, \ldots, I_n) = 1 \iff \mathcal{I}_j = k \text{ для } k = \overline{1, n},\ j = \overline{1, m}.$$

e_k — стандартный единичный вектор, $e_k = (0, \ldots, 0, 1, 0, \ldots, 0)$.

I^n — разбиение множества $\mathbb{N}_m$, $I^n = \{I_1, \ldots, I_n\}$.

$\mathcal{I}$ — линейное представление матрицы $\mathrm{E}^m(I^n)$ в виде вектора из $\mathbb{N}^m_n$.

$\mathcal{H}$ — базис Гильберта однородной системы НЛДУ, $\mathcal{H} = \{h^{(1)}, h^{(2)}, \ldots, h^{(q)}\}$.

$\mathbb{N}$ — натуральный ряд, $\mathbb{N} = \{1, 2, \ldots\}$.

$\mathbb{N}_m$ — начальный отрезок натурального ряда длины m, $\mathbb{N}_m = \{1, 2, \ldots, m\}$.

$\mathbb{Z}$ — множество целых чисел, $\mathbb{Z} = \{0, \pm 1, \pm 2, \ldots\}$.

$\mathbb{Z}_+$ — множество неотрицательных целых чисел, $\mathbb{Z}_+ = \{0, 1, 2, \ldots\}$.

Введение

При исследовании математической проблемы может потребоваться генерация набора частных задач при определенных ограничениях на результат и процесс генерации. Такими ограничениями могут служить, например, построение как самой частной задачи, так и ее решения (тестовые и эталонные задачи), ограничения на вид генерируемой задачи (генерация задач некоторого класса), ограничения на время и память процесса генерации (эффективная генерация) и т.д. В общем случае проблемная область может содержать бесконечное число частных задач. В силу этого важной представляется проблема развития методов, позволяющих выполнять эффективную генерацию конечных наборов частных задач при заданных ограничениях.

В данной работе в качестве математической проблемы рассматривается задача решения однородных систем неотрицательных линейных диофантовых уравнений (системы одНЛДУ). Основной задачей решения системы одНЛДУ будем считать задачу нахождения базиса Гильберта системы. В общем случае эта задача вычислительно трудоемка [5].

Одним из перспективных алгоритмов для эффективного решения частного класса систем одНЛДУ является синтаксический алгоритм, предложенный Д. Ж. Корзуном [5, 6]. Алгоритм позволяет решать ассоциированные с формальными грамматиками системы одНЛДУ (системы одАНЛДУ) [2, 3, 5]. В качестве альтернативного синтаксическому в работе рассматривается алгоритм решения произвольной системы одНЛДУ, предложенный M. Filgueiras и A.-P. Tomás [28, 29].

Задача генерации систем одАНЛДУ заключается в построении на основе заданных ограничений как самой системы, так и ее базиса Гильберта. В силу того, что требуется также и построение базиса Гильберта, задача генерации сталкивается с теми же проблемами вычислительной сложности, что и задача решения. Кроме того, ограничения зависят от приложений и могут иметь форму, трудную для непосредственного учета (без решения системы), например, ограничение на размер базиса Гильберта генерируемой системы. Все это требует развития соответствующего теоретического аппарата.

Ключевыми приложениями задачи генерации систем одАНЛДУ являются тестирование, экспериментальный и сравнительный анализ алгоритмов решения.

Тестирование позволяет проверить корректность реализации алгоритма решения на основе конечного набора сгенерированных тестовых систем. В частности, возможно проведение комплексных системных тестов, обеспечивающих качественное покрытие и гарантирующих с приемлемым уровнем значимости отсутствие ошибок в реализации. Это требует разработки специальной технологии и соответствующих программных средств, автоматизирующих процесс генерации и проверки тестов.

Экспериментальный анализ использует сгенерированную выборку систем одНЛДУ из некоторого частного класса для оценки эффективности алгоритма решения на заданном классе. Это также может включать построение карты эффективности алгоритма — определение областей предметной области со сходной экспериментальной сложностью вычислений. Программное обеспечение экспериментального анализа может быть построено на основе ПО для автоматизированного тестирования, расширив его дополнительными средствами вычисления статистических параметров.

Для выполнения сравнительного анализа эффективности нескольких алгоритмов решения необходимо использовать набор эталонных систем, на основе которого и проводится сравнение. Это требует расширения задачи экспериментального анализа на случай нескольких алгоритмов решения и соответствующего статистического инструментария.

Задача генерации систем одАНЛДУ двойственна по отношению к задаче решения. Если в первой требуется построить систему, исходя, в первую очередь, из ограничений на базис Гильберта получаемой системы, то во второй строится решение на основе заданного вида системы. Тем самым, теоретический и алгоритмический аппарат, разработанный для решения задачи генерации, может быть применим и для разработки двойственных алгоритмов решения систем одАНЛДУ, что можно рассматривать как еще одно важное приложение задачи генерации.

Представленные выше приложения задачи генерации позволяют получить "комплексный практический взгляд" на алгоритмическую часть теории систем одАНЛДУ и ее прикладной потенциал. Таким образом, представляется целесообразной интеграция соответствующего программного инструментария в виде Web-ориентированной системы научно-образовательного и прикладного сервиса. Такая система включает ряд реализаций алгоритмов решения систем одАНЛДУ, демонстрирует на удаленной основе через Интернет эффективность их работы, позволяет выполнять распределенное тестирование реализаций алгоритмов, их сравнительный и экспериментальный анализы, а также предоставляет вычислительную услугу решения систем одАНЛДУ, задаваемых пользователем.

Целью данной работы является разработка алгоритмов генерации систем одАНЛДУ, основанных на них (двойственных) алгоритмах решения таких систем, а также, применение разработанных алгоритмов для создания мобильного программного обеспечения комплексного тестирования, экспериментального и сравнительного анализа алгоритмов решения систем одАНЛДУ. Объектом исследования в работе являются алгоритмы решения систем НЛДУ, а предметом исследования — методы генерации тестовых систем НЛДУ для исследуемых алгоритмов.

Для достижения поставленной цели необходимо решить следующие задачи.

1) Выполнить теоретическое исследование систем одАНЛДУ, на основе которого разработать алгоритмы генерации тестовых систем.

2) Исследовать двойственность задач генерации и решения систем одАНЛДУ и ее применимость к созданию алгоритмов решения систем одАНЛДУ.

3) Разработать и реализовать мобильный программный инструментарий, который использует предложенные алгоритмы генерации для выполнения комплексного тестирования, экспериментального и сравнительного анализа алгоритмов решения систем одАНЛДУ.

4) Выполнить комплексное тестирование и экспериментальный анализ реализации syntactic solver синтаксического алгоритма решения однородных систем АНЛДУ [5] и его сравнительный анализ с альтернативным алгоритмом slopes [28, 29].

5) Интегрировать полученные результаты в систему Web-SynDic[1] [30].

Выпускная работа состоит из введения, четырех глав, заключения, библиографического списка использованной литературы (30 наименований), приложения, имеет общий объем 82 машинописные страницы, содержит 30 рисунков и 3 таблицы.

В первой главе приводится необходимый теоретический аппарат систем одАНЛДУ и представлены ключевые оригинальные результаты: (1) определение и свойства симметричных систем одАНЛДУ и (2) преобразование произвольной системы одАНЛДУ. Во второй главе выполнена постановка задачи генерации систем одАНЛДУ и представлены пять разработанных алгоритмов генерации. В третьей главе предлагаются двойственные алгоритмы решения: (1) базовый алгоритм для симметричных систем одАНЛДУ и

[1]Доступна по адресу `http://websyndic.cs.karelia.ru`

(2) для произвольной системы одАНЛДУ (на основе базового). Четвертая глава посвящена тестированию, экспериментальному и сравнительному анализу алгоритмов решения систем одАНЛДУ, предлагаются способы вычисления затрачиваемых алгоритмами ресурсов и представлены результаты соответствующих экспериментов с реализациями syntactic solver и slopes, а также, описываются разработанные программные системы alg_analyser и Web-SynDic.

Результаты выпускной работы докладывались и обсуждались на 54-й, 55-й и 57-й научных студенческих конференциях ПетрГУ, на международном научном семинаре FDPW'03, на конкурсе-конференции студентов и молодых ученых Северо-Запада "Технологии Microsoft в теории и практике программирования", а также на научных семинарах кафедры информатики и математического обеспечения ПетрГУ. По результатам работы опубликованы тезисы [8, 10–12, 14] и статья [13].

1. Однородные системы неотрицательных линейных диофантовых уравнений, ассоциированные с контекстно-свободными грамматиками

В настоящей главе представлен теоретический аппарат, в том числе и оригинальный, который необходим для построения предлагаемых нами в главах 2 и 3 алгоритмов генерации и решения задач линейного диофантова анализа. Текущее состояние теории НЛДУ представлено, например, в работах [17, 18, 20, 23, 25, 26, 29]. Важнейшие результаты исследований систем АНЛДУ можно найти в [2, 5, 7]. Основные положения данной главы опубликованы в [10, 12, 13].

В п. 1.1 представлен используемый понятийный аппарат теории систем одАНЛДУ. В п. 1.2 рассмотрен случай одного уравнения в системе одАНЛДУ. В п. 1.3 исследуется случай симметричных систем одАНЛДУ. В п. 1.4 сформулирована и доказана теорема о преобразовании произвольной системы одАНЛДУ к одному из рассмотренных в предыдущих двух пунктах частных случаев.

1.1. Обозначения и понятийный аппарат

Определение 1.1. *Однородной системой неотрицательных линейных диофантовых уравнений (одНЛДУ)* называется система однородных уравнений, все коэффициенты которой являются произвольными целыми числами, а компоненты решений принимают неотрицательные целые значения:

$$Ax = \mathbb{O}\,, \qquad A \in \mathbb{Z}^{n\times m}\,, \qquad x \in \mathbb{Z}_+^m\,, \tag{1.1}$$

где n — число уравнений системы, m — число неизвестных, A — матрица коэффициентов системы, вектор $\mathbb{O}$ — нулевой вектор из $\mathbb{Z}^n$.

Определение 1.2. Ненулевое решение $h \in \mathbb{Z}_+^m$ системы одНЛДУ (1.1) называется *неразложимым*, если оно не может быть представлено в виде суммы двух ненулевых решений этой же системы.

Определение 1.3. *Базисом Гильберта* системы одНЛДУ (1.1) называется множество всех ее неразложимых решений:

$$\mathcal{H} = \{h^{(1)}, \dots h^{(q)}\} ,$$

где q — число базисных решений.

Пример 1.1. Рассмотрим систему одНЛДУ:

$$\begin{cases} x_1 + x_3 - 2x_4 - x_5 = 0 \\ 5x_1 - x_2 + 7x_3 + 2x_4 + 3x_5 = 0 \\ x_4 - 2x_5 = 0 \end{cases}$$

Она имеет следующий базис Гильберта:

$$\mathcal{H} = \left\{ \begin{pmatrix} 5 \\ 32 \\ 0 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 4 \\ 34 \\ 1 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ 36 \\ 2 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 2 \\ 38 \\ 3 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 40 \\ 4 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 42 \\ 5 \\ 2 \\ 1 \end{pmatrix} \right\} .$$

Отметим, что согласно теореме о базисе Гильберта (см. напр. [17, 24]) он определяется единственным образом, всегда конечен и любое решение x системы (1.1) может быть выражено через базис Гильберта следующим образом:

$$x = \sum_{s=1}^{q} \alpha_s h^{(s)}, \qquad \alpha_1, \alpha_2, \dots, \alpha_q \in \mathbb{Z}_+. \tag{1.2}$$

Отметим, что, вообще говоря, разложение произвольного решения одНЛДУ по элементам базиса Гильберта неоднозначно.

Пример 1.2. Следующее частное решение $x = (5, 74, 5, 4, 2)^{\mathsf{T}}$ системы одНЛДУ из примера 1.1 может быть представлено в виде линейной комбинации элементов базиса Гильберта

$$x = h^{(1)} + h^{(6)} = h^{(2)} + h^{(5)} = h^{(3)} + h^{(4)} .$$

Неразложимость элементов базиса Гильберта эквивалентна их *минимальности*, т. е. любого решения h и произвольного базисного элемента $h' \in \mathcal{H}$, $h' \neq h$ неверно, что $h \leqslant h'$. При этом сравнение здесь покомпонентное:

$$(u_1, u_2, \dots, u_m) \leqslant (v_1, v_2, \dots, v_m) \quad \overset{\text{def}}{\Longleftrightarrow} \quad u_j \leqslant v_j \quad \forall\, j = 1, 2, \dots, m .$$

Таким образом, базисное решение покомпонентно не превосходит любое другое решение системы НЛДУ. Более того, базисные решения несравнимы друг с другом в смысле покомпонентного сравнения векторов.

Определение 1.4. *Разбиением конечного отрезка натурального ряда* $\mathbb{N}_m = \{1, \ldots, m\}$ называется множество подмножеств $I^n = \{I_0, I_1, \ldots, I_n\}$, где

$$I_k \cap I_j = \mathbb{O} \quad \forall\, k, j \in \mathbb{N}_n \cup \{0\}\,,\ k \neq j\,; \qquad \bigcup_{k=0}^{n} I_k = \mathbb{N}_m\,; \qquad I_k \neq \varnothing \quad \forall\, k \in \mathbb{N}_n\,.$$

Подмножество I_0 может быть пусто.

Будем называть *матрицей разбиения* $\mathrm{E}^m(I^n) = \mathrm{E}^m(I_0, I_1, \ldots, I_n)$ матрицу из $\{0,1\}^{n\times m}$:

$$\mathrm{E}^m_{k,i}(I^n) = 1 \overset{\text{def}}{\Longleftrightarrow} i \in I_k\,, \quad k = 1, 2, \ldots, n\,,\ i = 1, 2, \ldots, m\,.$$

Для любой матрицы разбиения $\mathrm{E}^m(I^n)$ выполняется следующее свойство:

$$\sum_{k=1}^{n} \mathrm{E}^m_{ki}(I^n) = 1 \quad \forall\, i = \mathbb{N}_m \setminus I_0\,, \qquad \sum_{k=1}^{n} \mathrm{E}^m_{ki}(I^n) = 0 \quad \forall\, i = I_0\,,$$

т.е. столбец i либо нулевой ($i \in I_0$), либо содержит в точности одну единицу ($i \notin I_0$)[1].

Определение 1.5. *Ассоциированной с контекстно-свободной грамматикой системой одНЛДУ* (системой одАНЛДУ) называется система вида

$$\mathrm{E}^m(I_0, I_1, \ldots, I_n)x = Ax,$$

где $I^n = \{I_0, I_1, \ldots, I_n\}$ — разбиение $\mathbb{N}_m$, n — число уравнений системы, m — число неизвестных, $\mathrm{E}^m(I^n) = \mathrm{E}^m(I_0, I_1, \ldots, I_n)$ — матрица разбиения.

В частности, для примера 1.1 матрица $\mathrm{E}^m(I^n)$ будет следующего вида (см. также пример 1.3 ниже):

$$\mathrm{E}^5(I_0, I_1, I_2, I_3) = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \qquad I_0 = \{5\},\ \ I_1 = \{1,3\},\ \ I_2 = \{2\},\ \ I_3 = \{4\}.$$

Случай $I_0 \neq \varnothing$ легко преобразуется в случай $I_0 = \varnothing$ путем добавления соответствующих I_0-неизвестных в левую и правую части произвольных уравнений. В дальнейшем будем предполагать, что $I_0 = \varnothing$.

Связь систем одАНЛДУ с КС-грамматиками представлена в приложении.

[1] Матрица $\mathrm{E}^m(I^n)$ допускает эффективную реализацию в виде вектора $\mathcal{I}^m \in \mathbb{N}^m_n$, $\mathcal{I}^m_i = l \Longleftrightarrow \mathrm{E}^m_{l,i}(I^n) = 1$. Так, в примере 1.1 вектор $\mathcal{I}^m = (1, 2, 1, 3, 0)$.

Пример 1.3. Система одНЛДУ из примера 1.1 может быть записана в формате одАНЛДУ

$$\begin{cases} x_1 + x_3 &= 2x_4 + x_5 \\ x_2 &= 5x_1 + 7x_3 + 2x_4 + 3x_5 \\ x_4 + x_5 &= 3x_5 \end{cases}$$

где $n = 3$, $m = 5$, $I_0 = \varnothing$, $I_1 = \{1, 3\}$, $I_2 = \{2\}$, $I_3 = \{4, 5\}$,

$\mathrm{E}(I^{3,5}) = \mathrm{E}(I_1, I_2, I_3) = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$ и $A = \begin{pmatrix} 0 & 0 & 0 & 2 & 1 \\ 5 & 0 & 7 & 2 & 3 \\ 0 & 0 & 0 & 0 & 3 \end{pmatrix}$. Отметим, что неизвестная x_5 была добавлена к последнему уравнению, чтобы добиться $I_0 = \varnothing$.

В линейной алгебре (см. напр. [9]) две системы уравнений называются эквивалентными, если их множества решений совпадают. В данной работе нам удобнее использовать другое понятие эквивалентности — с точностью до перенумерации неизвестных.

Определение 1.6. Пусть $I' = (I'_1, \dots, I'_n)$ и $I'' = (I''_1, \dots, I''_n)$ — произвольные разбиения $\mathbb{N}_m$. Две системы одАНЛДУ $\mathrm{E}(I')x = A'x$ и $\mathrm{E}(I'')x = A''x$ называются *эквивалентными* (с точностью до перенумерации), если перенумерацией неизвестных и уравнений их можно привести к системе $\mathrm{E}^m(I^n)x = Ax$, где матрица $\mathrm{E}^m(I^n) = \mathrm{E}^m(I_1, \dots, I_n)$ имеет вид, который мы будем называть *стандартной формой записи системы одАНЛДУ*:

$$\mathrm{E}^m(I_1, \dots, I_n) = \begin{pmatrix} 1 & \dots & 1 & 0 & \dots\dots\dots\dots\dots\dots & & & & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 & 0 & \dots\dots\dots & 0 \\ & & & & & & & & \\ \dots\dots & \dots\dots & \dots\dots & \dots\dots & \dots\dots & \dots\dots & \dots\dots & \dots\dots & \dots\dots \\ & & & & & & & & \\ 0 & \dots\dots\dots\dots\dots\dots & & & & 0 & 1 & \dots & 1 \end{pmatrix},$$

где $I_1 = \{1, \dots, k\} = \mathbb{N}_k$, $I_2 = \{k+1, \dots, j\} = \mathbb{N}_j \setminus \mathbb{N}_k$, $\dots$, $I_n = \{l+1, \dots, m\} = \mathbb{N}_m \setminus \mathbb{N}_l$.

Базисы Гильберта эквивалентных систем одАНЛДУ одинаковы с точностью до перестановки неизвестных. Это свойство эквивалентности будет использоваться нами при построении алгоритмов генерации (гл. 2) и решения (гл. 3). Генерация одной системы в стандартной форме означает фактически генерацию $m! * n!$ тестовых систем, получаемых перестановкой неизвестных. Аналогично, решение одной системы в стандартной форме дает решение и для остальных систем, эквивалентных исходной.

Пример 1.4. Для системы одАНЛДУ из примера 1.3 стандартный вид матрицы $\mathrm{E}^m(I^n)$:

$$\mathrm{E}^m(I_1, I_2, I_3) = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Стандартный вид системы:

$$\begin{cases} y_1 + y_2 & = & 2y_4 + y_5 \\ y_3 & = & 5y_1 + 7y_2 + 2y_4 + 3y_5 \\ y_4 + y_5 & = & 3y_5 \end{cases}$$

Перенумерация неизвестных: $y_1 = x_1$, $y_2 = x_3$, $y_3 = x_2$, $y_4 = x_4$, $y_5 = x_5$.

Определения 1.7-1.8 относятся к приложению преобразования Гаусса для последовательного вычисления базиса Гильберта системы (см. напр. [18]).

Определение 1.7. *Зависимые неизвестные* — неизвестные, значения которых вычисляются через остальные неизвестные. Остальные неизвестные будем называть *свободными.*

Пример 1.5. Рассмотрим следующее уравнение

$$x_1 + x_2 + x_4 + x_5 = 10x_3 + 3x_4 + 2x_6.$$

Зависимыми неизвестными являются x_1, x_2 и x_5. Неизвестные x_3, x_4 и x_6 являются свободными, т.к. при решении данного уравнения (см. п. 1.2) для неизвестных x_3, x_4 и x_6 задаются определенные значения, а неизвестные x_1, x_2 и x_5 вычисляются.

Определение 1.8. Пусть задана система одНЛДУ $S = S' \wedge S''$ с n уравнениями. $\mathcal{H}'$ и $\mathcal{H}''$ — базисы Гильберта систем S' и S'' соответственно, $\mathcal{H}' = \{h_1, h_2, \ldots, h_q\}$. Каждое решение системы S' есть линейная комбинация ее базисных векторов $x = c_1h_1 + c_2h_2 + \cdots + c_qh_q$. Множество решений системы S есть пересечение множеств решений систем S' и S'', следовательно, базисные решения системы S также являются линейными комбинациями базисных решений системы S'.

Под *подстановкой базиса Гильберта* будем понимать подстановку линейной комбинации базисных векторов $\mathcal{H}' = \{h_1, h_2, \ldots, h_q\}$ в систему S'' и с неизвестными $c_1, c_2, \ldots, c_q$. При этом количество уравнений в системе равно $(n - l)$, l — количество уравнений системы S'.

Пример 1.6. Рассмотрим систему одАНЛДУ.

$$\begin{cases} x_1 + x_2 & = & 2x_3 \\ x_3 + x_4 & = & 2x_1 + x_2 + 3x_3 \\ x_5 & = & x_4 + x_5 \end{cases}$$

В качестве системы S' возьмем первое уравнение $x_1 + x_2 = 2x_3$. Остальные уравнения образуют систему S''. Базисом Гильберта для системы S' будет

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} : \ \mathcal{H}' = \left\{ \begin{array}{ccccc} h_1 & h_2 & h_3 & h_4 & h_5 \\ \begin{pmatrix} 0 \\ 2 \\ 1 \\ 0 \\ 0 \end{pmatrix}, & \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}, & \begin{pmatrix} 2 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, & \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, & \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \end{array} \right\}.$$

Любое решение уравнения S' есть $h = c_1h_1 + c_2h_2 + c_3h_3 + c_4h_4 + c_5h_5$. Следовательно, процесс подстановки базиса Гильберта для системы S'' будет выглядеть следующим образом

$$\begin{cases} c_1 + c_2 + c_3 &= 2(c_2 + 2c_3) + (2c_1 + c_2) + 3(c_1 + c_2 + c_3) \\ c_5 &= c_4 + c_5 \end{cases}$$

или

$$\begin{cases} c_1 + c_2 + c_3 &= 5c_1 + 6c_2 + 7c_3 \\ c_5 &= c_4 + c_5 \end{cases}.$$

Подстановка базиса Гильберта будет использована нами при преобразовании системы одАНЛДУ для построения базиса Гильберта конечной системы из базиса Гильберта одного уравнения одАНЛДУ или системы содАНЛДУ.

Определение 1.9. *Уравнение Фробениуса с единичными коэффициентами* (при неизвестных) — это НЛДУ вида

$$\sum_{i=1}^{n} x_i = T\,, \tag{1.3}$$

где T — неотрицательная целая константа.

В случае $T = 0$ уравнение (1.3) имеет только нулевое решение. Для $T > 0$ в [15] показано, что это уравнение имеет конечное множество решений, число которых определяется формулой

$$q = C_{T+n-1}^{n-1} = \frac{(T+n-1)!}{T!(n-1)!}. \tag{1.4}$$

Все решения несравнимы друг с другом в смысле покомпонентного сравнения векторов.

Пример 1.7. Рассмотрим следующее уравнение Фробениуса с единичными коэффициентами:

$$x_1 + x_2 + x_3 = 4.$$

Множество решений этого уравнения состоит из $q = \dfrac{(4+3-1)!}{4!(3-1)!} = \dfrac{6!}{4!2!} = 15$ элементов:

$$\mathcal{N} = \left\{ \begin{pmatrix} 4 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 3 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 3 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 2 \\ 2 \\ 0 \end{pmatrix}, \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 2 \\ 0 \\ 2 \end{pmatrix}, \begin{pmatrix} 1 \\ 3 \\ 0 \end{pmatrix}, \right.$$

$$\left. \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 3 \end{pmatrix}, \begin{pmatrix} 0 \\ 4 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 3 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 3 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 4 \end{pmatrix} \right\}$$

Уравнения Фробениуса будут использованы нами при решении системы одАНЛДУ для $n = 1$.

1.2. Случай одного уравнения в системе одАНЛДУ

Рассмотрим систему одАНЛДУ, состоящую из одного уравнения ($n = 1$). В этом случае система одАНЛДУ имеет следующий вид[2]:

$$\sum_{k=1}^{m'} x_k \; = \sum_{k=m'+1}^{m} a_k x_k, \qquad a_{m'+1}, \ldots, a_m \in \mathbb{N}, \tag{1.5}$$

Здесь $\mathrm{E}^m(I^1) = (1,1,\ldots,1) \in \mathbb{Z}_+^m$, $A = (0,0,\ldots,0,(a_{m'+1}+1),\ldots,(a_m+1))$. Данное уравнение является одним из частных случаев, к которому сводится произвольная система одАНЛДУ (см. п. 1.4).

Докажем теорему, которая дает способ вычисления базиса Гильберта любого уравнения вида (1.5).

Теорема 1.1. *Базисом Гильберта уравнения* (1.5) *является множество*

$$\mathcal{H} = \left\{ h \in \mathbb{Z}_+^m \;\middle|\; \begin{array}{l} h = (c_1,\ldots,c_{m'},0,\ldots,0,1,0,\ldots,0)^{\mathsf{T}}, \\ \text{1 стоит на } j\text{-й позиции, } m' < j \leqslant m, \\ \sum\limits_{k=1}^{m'} c_k = a_j, \quad c_k \in \mathbb{Z}_+ \end{array} \right\}. \tag{1.6}$$

[2] В варианте записи, когда требуется, чтобы $I_0 = \varnothing$, данное уравнение принимает вид $\sum\limits_{k=1}^{m} x_k = \sum\limits_{k=m'+1}^{m} (a_k+1) x_k$.

Доказательство. Упорядочим элементы множества $\mathcal{H}$ из условия (1.6) теоремы следующим образом:

$$\mathcal{H} = \left\{ \overbrace{\begin{pmatrix} c_{11} \\ \vdots \\ c_{m'1} \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}}^{h^{(1)}}, \overbrace{\begin{pmatrix} c_{12} \\ \vdots \\ c_{m'2} \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}}^{h^{(2)}}, \ldots, \overbrace{\begin{pmatrix} c_{1q} \\ \vdots \\ c_{m'q} \\ 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}}^{h^{(q)}} \right\} = \left\{ \begin{pmatrix} c_{*s} \\ e_l \end{pmatrix} \right\}_{s=1}^{q}$$

При этом, вектора $h^{(s)}$, для которых $h_j^{(s)} = 1$ $(m' < j \leqslant m)$, взаимнооднозначно соответствуют решениям следующего уравнения Фробениуса с единичиными коэффициентами и неизвестными c_{*s}:

$$\sum_{k=1}^{m'} c_{ks} = a_j. \tag{1.7}$$

Очевидно, что любой вектор $h^{(s)} = \begin{pmatrix} c_{*s} \\ e_{j-m'} \end{pmatrix}$ является решением уравнения (1.5). В то же время, любые два решения $h', h'' \in \mathcal{H}$ несравнимы друг с другом: 1) либо $h'_{m'+k} = h''_{m'+l} = 0$ и $h'_{m'+l} = h''_{m'+k} = 1$ (единицы на разных позициях, $1 \leqslant k, l \leqslant m - m'$, $k \neq l$), а значит h' и h'' несравнимы, 2) либо $h'_{m'+k} = h''_{m'+k} = 1$, а тогда h' и h'' несравнимы в силу несравнимости c' и c'' как решений уравнения (1.7) для $j = m' + k$. Тем самым, доказано, что множество (1.6) содержит только несравнимые решения уравнения (1.5).

Докажем теперь, что произвольное решение $x = (x_1, \ldots, x_m)^\mathsf{T}$ уравнения (1.5) может быть выражено через элементы заданного множества. Тем самым будет доказано, что (1.6) содержит все базисные решения уравнения (1.5).

Пусть x — произвольное ненулевое решение уравнения (1.5). Тогда $\sum\limits_{j=m'+1}^{m} x_j > 0$. Следовательно существует j, т. ч. $m' + 1 \leqslant j \leqslant m$ и $x_j > 0$. Докажем, что найдутся такие $x' \in \mathbb{Z}_+^m$ и $h \in \mathcal{H}$, что $x = x' + h$ и $h_j = 1$.

Подставим $x = x' + h$ в левую часть уравнения (1.5):

$$\sum_{k=1}^{m'} (x'_k + h_k) \;=\; \sum_{k=m'+1}^{m} a_k x_k \;.$$

В силу $h_k = 0$ для $k = m' + 1, \ldots, j-1, j+1, \ldots, m$ и $h_j = 1$ должно выполняться следующее уравнение для x' и h:

$$\sum_{k=1}^{m'} x'_k + \sum_{k=1}^{m'} h_k \; = \sum_{\substack{k=m'+1 \\ k \neq j}}^{m} a_k x_k \; + \; a_j(x_j - 1) \; + \; a_j \; .$$

Это уравнение разрешимо в неотрицательных целых, если, например, 1) x' есть решение уравнения

$$\sum_{k=1}^{m'} x'_k \; = C \; , \quad \text{где } C = \sum_{\substack{l=m'+1 \\ l \neq j}}^{m} a_l x_l \; + a_j(x_j - 1) \geqslant 0 \; ,$$

и 2) найдется в (1.6) такое h, что

$$\sum_{k=1}^{m'} h_k = a_j \; , \quad h_{m'+1} = 0 \, , \; \ldots , \; h_{j-1} = 0 \, , \; h_j = 1 \, , \; h_{j+1} = 0 \, , \; \ldots , \; h_m = 0 \, ,$$

Уравнение из условия 1) всегда разрешимо в силу свойств уравнения Фробениуса, а вектор h, удовлетворяющий условию 2), очевидно найдется в силу построения множества (1.6). Тем самым, искомое представление $x = x' + h$ найдено.

Вектор x' также является решением уравнения (1.5). Рассуждая аналогично, можно найти представление $x' = x'' + h$ для некоторого $h \in \mathcal{H}$. Продолжая этот процесс, за конечное число шагов придем нулевому решению:

$$x = x' + h' = x'' + h'' + h' = \cdots = \mathbb{O} + \sum_{s=1}^{q} \alpha_s h^{(s)}.$$

Итак, получаем разложение произвольного решения x в виде неотрицательной линейной целочисленной комбинации векторов из $\mathcal{H}$, что и завершает доказательство теоремы. □

Отметим, что другое доказательство этой теоремы, основанное на методах теории формальных грамматик, дано в [4].

1.3. Симметричные системы одАНЛДУ

В данном параграфе рассматривается частный случай систем одАНЛДУ — системы с симетричной структурой левой и правой частей. Данные системы является одним из частных видов, к которому может быть преобразована произвольная система одАНЛДУ (см. п. 1.4).

Отметим, что в работах [1,16] рассматриваются системы неравенств, аналогичных симметричным системам одАНЛДУ, и исследуется вопрос о разрешимости этих систем.

Однако в случае неравенств, контур в графе системы уравнений не всегда является решением, т.к. он может быть решением только если все неравенства входящие в контур являются нестрогими.

Данные системы неравенств в некотором смысле аналогичны симметричным системам одАНЛДУ, т.к. в обоих случаях исследуется вопрос о существовании контуров в графах.

1.3.1. Свойства симметричных систем одАНЛДУ

Определение 1.10. *Симметричной системой одАНЛДУ* (система содАНЛДУ) будем называть следующую систему из n уравнений и с m неизвестными:

$$\mathrm{E}^m(I^n)x = \mathrm{E}^m(J^n)x\,, \tag{1.8}$$

где $\mathrm{E}^m(I^n)$ и $\mathrm{E}^m(J^n)$ — матрицы разбиения.

Пример 1.8. Рассмотрим систему содАНЛДУ:

$$\begin{cases} x_1 & = & x_1 \\ x_2 + x_4 & = & x_3 + x_4 + x_5 \\ x_3 + x_5 & = & x_2 \end{cases} \tag{1.9}$$

Для данной системы содАНЛДУ матрицы разбиения выглядят следующим образом:

$$\mathrm{E}^m(I^n) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}, \quad \mathrm{E}^m(J^n) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

Базис Гильберта системы содАНЛДУ (1.9):

$$\mathcal{H} = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}.$$

Теорема 1.2. *Любое уравнение системы содАНЛДУ* (1.8) *равносильно сумме всех остальных уравнений.*

Доказательство. Рассмотрим систему содАНЛДУ (1.8). Выберем произвольное уравнение k. Остальные уравнения образуют систему одНЛДУ

$$\sum_{i\in I_p} x_i = \sum_{j\in J_p} x_j, \quad p = 1, \ldots, k-1, k+1, \ldots, n. \tag{1.10}$$

Сложим все уравнения системы (1.10). По свойству матриц разбиения получаем

$$\sum_{i\in \mathbb{N}_m \setminus I_k} x_i = \sum_{j\in \mathbb{N}_m \setminus J_k} x_j. \tag{1.11}$$

Вычитая уравнение (1.11) из тождественного равенства $\sum\limits_{i\in\mathbb{N}_m} x_i = \sum\limits_{j\in\mathbb{N}_m} x_j$ получаем

$$\sum_{i\in J_k} x_i = \sum_{j\in I_k} x_j.$$

Последнее полученное уравнение есть в точности уравнение k системы (1.8). $\square$

Следствие 1.2.1. *Множества решений систем (*1.8*) и (*1.10*) совпадают.*

Доказательство. Пусть x — произвольное решение системы (1.8). Следовательно x удовлетворяет каждому уравнению этой системы. По построению системы (1.10) x удовлетворяет каждому уравнению системы (1.10). Следовательно x — решение системы (1.10).

Пусть теперь x — произвольное решение системы (1.10), т.е. x удовлетворяет каждому уравнению этой системы, а значит и сумме всех уравнений (1.11) системы (1.10). Следовательно, x удовлетворяет каждому уравнению системы (1.8). $\square$

Очевидно, что базисы Гильберта систем (1.8) и (1.10) также совпадают, в силу свойства единственности базиса Гильберта.

1.3.2. Графическое представление симметричных систем одАНЛДУ

Представим произвольную систему содАНЛДУ в виде конечного ориентированного графа (*граф системы содАНЛДУ*) следующим образом:

— вершинам графа соответствуют уравнения;

— дугам графа соответствуют неизвестные системы.

Дуга “x_i” выходит из вершины “k” и входит в вершину “j”, если переменная x_i встречается в левой части уравнения k и в правой части уравнения j. По свойству матрицы разбиения каждое неизвестное встречается в точности один раз в левой и правой частях системы. Используем ориентацию “слева направо” (в противном случае можно поменять местами левую и правую части системы содАНЛДУ).

Пример 1.9. Ориентированный граф для системы (1.9) из примера 1.8 представлен на рис. 1. В данном примере вершинам “1”, “2” и “3” соответствуют уравнения 1, 2 и 3

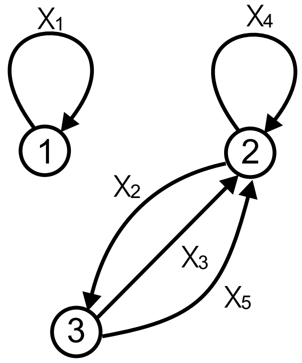


Рис. 1. Графическое представление содАНЛДУ.

системы (1.9):

$$\begin{cases} 1: & x_1 & = & x_1 \\ 2: & x_2 + x_4 & = & x_3 + x_4 + x_5 \\ 3: & x_3 + x_5 & = & x_2 \end{cases}$$

Аналогично, решения системы содАНЛДУ также можно представить в виде конечного ориентированного мультиграфа (*мультиграф решения системы содАНЛДУ*) следующим способом:

— вершины графа соответствуют уравнениям;

— каждой компоненте решения соответствуют дуги графа в количестве равном величине компоненты решения.

Дуги “x_i” выходят из вершины “k” и входят в вершину “j”, если переменная x_i встречается в левой части уравнения k и в правой части уравнения j. Без потери общности полагаем ориентацию “слева направо”.

Таким образом, построение аналогично случаю графа системы, однако число кратных дуг определяется значениями компонент решения.

Пример 1.10. Решению $x = (1, 3, 2, 0, 1)^{\mathsf{T}}$ системы содАНЛДУ (1.9) соответствует мультиграф, представленный на рис. 2.

Лемма 1.1. *Число входящих дуг для вершины мультиграфа решения системы содАНЛДУ равно числу исходящих.*

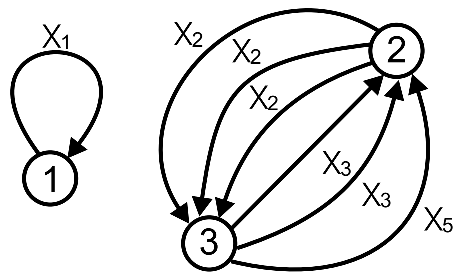


Рис. 2. Графическое представление решения $x = (1, 3, 2, 0, 1)^{\mathsf{T}}$.

Доказательство. Пусть k — произвольное уравнение системы содАНЛДУ, x — решение системы. Подставим решение x в уравнение k. По построению мультиграфа решения в правой части уравнения k записано число входящих дуг, а в левой части — число выходящих дуг. Следовательно число входящих дуг вершины мультиграфа решения равно числу выходящих дуг. □

Лемма 1.2. *Мультиграф решения системы содАНЛДУ содержит хотя бы один простой контур.*

Доказательство. Допустим противное: пусть x — решение системы содАНЛДУ и мультиграф решения x не содержит контуров.

Пусть вершина a_1 мультиграфа решения x имеет выходящую дугу p_1. Так как мультиграф решения x не содержит контуров, то дуга p_1 входит в вершину a_2, $a_2 \neq a_1$. По лемме 1.1 существует дуга p_2 выходящая из вершины a_2. Так как мультиграф решения x не содержит контуров, то дуга p_2 входит в вершину a_3, $a_3 \neq a_1$, $a_3 \neq a_2$. По лемме 1.1 существует дуга p_3 выходящая из вершины a_3. И так далее.

В итоге мы получим цепь с бесконечным числом различных дуг и вершин, что противоречит конечности мультиграфа. □

Теорема 1.3. *Решение x системы содАНЛДУ является неразложимым тогда и только тогда, когда мультиграф решения x — простой контур.*

Доказательство. Допустим противное: существует решение x, т.ч. мультиграф решения x не является простым контуром. Докажем, что решение x не является неразложимым.

По лемме 1.2 мультиграф решения x содержит хотя бы один простой контур. Выделим в мультиграфе решения x простой контур. Данный контур является мультиграфом решения y системы содАНЛДУ, т.к.:

— дуги контура соответствуют единичным значениям соответствующих неизвестных;

— данное решение удовлетворяет всем уравнениям системы (число выходящих дуг равно числу входящих или, другими словами, левые части уравнений равны правым).

Так как x и y являются решениями системы содАНЛДУ и $y < x$ (сравнение покомпонентное), то $x - y$ тоже является решением системы содАНЛДУ. Следовательно x не является неразложимым решением системы содАНЛДУ, что противоречит предположению. □

Теорема 1.4. *Базис Гильберта системы содАНЛДУ содержит только* $\{0,1\}$*-вектора, т.е.* $\mathcal{H} \subset \{0,1\}^{m\times q}$*, где* m *— число неизвестных,* q *— число компонент базиса Гильберта.*

Доказательство. Базис Гильберта — это множество неразложимых решений содАНЛДУ. Мультиграф неразложимого решения является простым контуром. Простой контур содержит не более одного вхождения для каждой дуги неизвестной. □

Теорема 1.5. *Минимальная размерность базиса Гильберта для множества систем содАНЛДУ* (1.8) *с заданной матрицей разбиения* $\mathrm{E}^m(I^n)$ *равна*

$$\min_{k\in\mathbb{N}_n} \sum_{i\in I_k} \mathrm{E}^m_{k,i}(I^n).$$

Доказательство. На основе графовой интерпретации систем содАНЛДУ данная теорема звучит следующим образом: на множестве ориентированных графов с n вершинами, m дугами и с заданным распределением выходов дуг минимальное количество простых контуров равно минимальному числу выходов у вершин графа.

Пусть минимальное количество выходов у вершин графа равно t. Докажем, что невозможно построить заданный граф с количеством различных простых контуров меньше t.

Допустим противное: существует заданный граф $G(k)$ с числом простых контуров $k < t$. Тогда у каждой вершины графа $G(k)$ существует выходящая дуга, по которой не проходит ни одно решение.

Возьмем вершину A_1 графа $G(k)$. Из вершины A_1 выходит дуга p_1, по которой не проходит ни одно решение. Дуга p_1 входит в вершину A_2, $A_2 \neq A_1$ (иначе мы построили новый простой контур). Из вершины A_2 выходит дуга p_2, по которой не проходит ни одно решение. Дуга p_2 входит в вершину A_3, $A_3 \neq A_1$, $A_3 \neq A_2$ (иначе мы построили новый контур). И так далее.

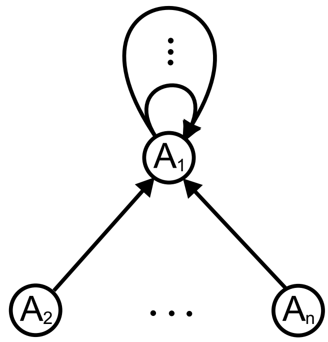


Рис. 3. Граф с минимальным количеством простых контуров.

В итоге мы построили простую цепь с бесконечным числом вершин и дуг, что противоречит конечности графа. Следовательно не существует заданного графа с числом простых контуров $k < t$.

Построим явно граф $G(t)$ с числом простых контуров $k = t$.

Пусть вершина A_1 имеет число выходов равное t. Тогда сделаем для всех дуг входящей вершиной вершину A_1 (см рис. 3). Система содАНЛДУ будет выглядеть следующим образом:

$$\begin{cases} x_1 + x_2 + \cdots + x_t & = & x_1 + x_2 + \cdots + x_m \\ x_{t+1} + x_{t+2} + \cdots + x_p & = & 0 \\ x_{p+1} + x_{p+2} + \cdots + x_z & = & 0 \\ \ldots \\ x_{z+1} + x_{z+2} + \cdots + x_m & = & 0 \end{cases},$$

а базис Гильберта — $\{e_1, e_2, \ldots, e_t\}$.

Докажем, что у данного графа $G(t)$ ровно t простых контуров. Дуги, выходящие из вершины A_1 образуют простые контуры $A_1 \to A_1$. Количество таких контуров равно t. Любые дуги, выходящие не из вершины A_1 входят в вершину A_1. Следовательно они не участвуют в образовании новых контуров (отсутствуют переходы из вершины A_1 на другие вершины). Следовательно граф $G(t)$ имеет ровно t простых контуров. $\square$

1.4. Преобразование произвольной системы одАНЛДУ

Рассмотрим следующее преобразование произвольной системы одАНЛДУ $\mathrm{E}^m(I^n)x = Ax$. Пусть $S^{(0)}(I^n, A)$ — исходная система одАНЛДУ. Ее можно записать как $\widetilde{A}^{(0)}x = \mathbb{O}$, где $\widetilde{A}^{(0)} = A - \mathrm{E}^m(I^n) \in \{\mathbb{Z}_+ \cup \{-1\}\}^{n \times m}$. Без потери общности будем предполагать, что $S^{(0)}$ не содержит тождественных уравнений $0 = 0$, поскольку их всегда можно удалить без изменения множества решений системы.

Преобразование будем выполнять последовательно $S^{(0)} \to S^{(1)} \to \dots \to S^{(r)}, r < n$, исключая на каждом шаге одно уравнение и не менее одной неизвестной. Система $S^{(l)}$ содержит $n-l$ уравнений и $m_l < m$ неизвестных $(l = 1, 2, \dots, r)$. В итоге получим систему $S^{(r)}$, для которой дальнейшее исключение невозможно.

Пусть $S^{(l)}$ — текущая система $\widetilde{A}^{(l)}x = \mathbb{O}$, $A^{(l)} \in \{\mathbb{Z}_+ \cup \{-1\}\}^{(n-l)\times m_l}$. Сложим все уравнения:

$$\sum_{j=1}^{m_l} c_j x_j = 0, \qquad \text{где } c_j = \sum_{i=1}^{n-l} \widetilde{A}^{(l)}_{ij}. \tag{1.12}$$

Преобразование $S^{(l)} \to S^{(l+1)}$ выполняется, если есть хотя бы один коэффициент $c_k = -1$. В этом случае найдется уравнение i_0 т. ч.

$$K = \left\{k \mid \widetilde{A}^{(l)}_{i_0 k} = -1 \text{ и } \widetilde{A}^{(l)}_{ik} = 0 \text{ для всех } i \neq i_0\right\} \neq \varnothing. \tag{1.13}$$

Без потери общности полагаем $i_0 = n-l$, т.е. последнее уравнение системы $S^{(l)}$. В противном случае это можно получить путем перенумерации уравнений. Аналогично полагаем $K = \{r_l+1, \dots, m_l\}$, т.е. последние неизвестные. В противном случае этого можно достичь путем перенумерации неизвестных.

Используя уравнение n_l выразим неизвестные x_k, $k \in K$:

$$\sum_{k=r_l+1}^{m_l} x_k = \sum_{k=1}^{r_l} \widetilde{A}^{(l)}_{n_l k} x_k = T_{l+1}(x_1, \dots, x_{r_l}). \tag{1.14}$$

Поскольку, помимо данного уравнения неизвестные x_k, $k \in K$ в системе $S^{(l)}$ больше не встречаются, то, при заданных значениях неизвестных $x_1, \dots, x_{r_l}$, используя последнее уравнение, можно подобрать соответствующие значения неизвестных x_k, $k \in K$. Следовательно, при решении системы $S^{(l)}$ требуется определить значения неизвестных $x_1, \dots, x_{r_l}$, а, затем, с помощью уравнения (1.14) найти значения остальных неизвестных. То есть, неизвестные $x_1, \dots, x_{r_l}$ являются свободными, а неизвестные $x_{r_l+1}, \dots, x_{m_l}$ — зависимые.

Очередная система $S^{(l+1)}$ строится из $S^{(l)}$ исключением уравнения n и неизвестных x_k, $k \in K$. Система $S^{(l+1)}$ является системой одАНЛДУ, т.к. она была получена из системы одАНЛДУ путем исключения одного уравнения и неизвестных, которые входили только в это уравнение.

В данном переходе мы разбиваем систему одАНЛДУ на две системы, одна из которых состоит из одного уравнения.

Преобразование завершается через конечное число шагов, поскольку на каждом шаге из системы удаляется в точности одно уравнение. В результате получаем систему $S^{(r)}$, где $0 \leqslant r < n$, n — число уравнений в исходной системе $S^{(0)}$.

Если система $S^{(r)}$ состоит из одного уравнения, то она имеет следующий вид:

$$\sum_{j\in I_1} x_j = \sum_{j=1}^{m_r} a_{1j}x_j. \tag{1.15}$$

Данное уравнение не является пустым, т.е. не равно $0 = 0$, в силу начального предположения.

Если $p = n - r > 1$, то все коэффициенты $c_j \geqslant 0$. Неотрицательный вектор x' может выступать в качестве решения уравнения (1.12) тогда, когда при коэффициентах $c_j > 0$ стоят нулевые компоненты решений, т.е. $x_j = 0$. Следовательно, можно исключить из рассмотрения неизвестные x_j при $c_j > 0$ и положить их равными 0.

Для коэффициентов $c_j = 0$ система имеет следующий вид:

$$\sum_{i\in I_k} x_i = \sum_{j\in J_k} x_j,\ k = \overline{1,p}, \tag{1.16}$$

где $\bigcup\limits_{k=\overline{1,p}} I_k = \bigcup\limits_{k=\overline{1,p}} J_k$. Разбиения I, J — разбиения для системы $S^{(r)}$ с учетом того, что $x_j = 0$ для $c_j > 0$. Система (1.16) является системой содАНЛДУ.

Базис Гильберта $\mathcal{H}^{(0)}$ исходной системы $S^{(0)}$ может быть вычислен по базису Гильберта $\mathcal{H}^{(r)}$ итоговой системы $S^{(r)}$ (уравнение (1.15) или система (1.16)) в порядке, обратном выполненному преобразованию:

$$\mathcal{H}^{(r)} \to \mathcal{H}^{(r-1)} \to \cdots \to \mathcal{H}^{(0)}.$$

Пусть $\mathcal{H}^{(l+1)}$ — текущий базис Гильберта, соответствующий системе $\widetilde{A}^{(l+1)}x = \mathbb{O}$. Тогда существует уравнение (1.14) которое было исключено из системы $\widetilde{A}^{(l)}x = \mathbb{O}$ для преобразования в систему $\widetilde{A}^{(l+1)}x = \mathbb{O}$. Компоненты решения x_k, $k \in K$ можно получить используя подстановку базиса Гильберта $\mathcal{H}^{(l+1)}$ в уравнение (1.14). В силу свойств подстановки (см. п. 1.1) полученные решения составляют базисом Гильберта $\mathcal{H}^{(l)}$ системы $\widetilde{A}^{(l)}x = \mathbb{O}$.

Теорема 1.6. *Пусть задана произвольная система одАНЛДУ из n уравнений с m неизвестными:*

$$\mathrm{E}^m(I_1,\ldots,I_n)x = Ax. \tag{1.17}$$

Задача нахождения базиса Гильберта системы (1.17) *сводится к задаче нахождения базиса Гильберта либо уравнения вида* (1.15)*, либо системы содАНЛДУ вида* (1.16)*.*

Доказательство. Выполним описанное выше преобразование системы (1.17) либо к виду (1.15), либо к виду (1.16). Решаем задачу нахождения базиса Гильберта полученной системы. Вычисляем базис Гильберта исходной системы, используя подстановку базисов в порядке, обратном выполненному преобразованию. □

С точки зрения генерации системы одАНЛДУ теорема 1.6 сводит задачу к построению уравнения (1.15) или системы содАНЛДУ (1.16), а, затем к преобразованию к системе искомого вида.

С точки зрения решения системы одАНЛДУ теорема 1.6 сводит задачу к преобразованию исходной системы к виду (1.15) или (1.16), решению этих более простых систем и возврату к исходной системе с получением ее базиса Гильберта.

Если исходная система (1.17) сводится к уравнению (1.15), то нахождение базиса Гильберта последнего может быть получено на основе теоремы 1.1 (см. с. 16). Задача нахождения базиса Гильберта системы содАНЛДУ (1.16) будет решена нами в п. 3.1.

Замечание 1.1. В ряде случаев исходные системы одАНЛДУ могут иметь базис Гильберта меньшей размерности, нежели базисы Гильберта промежуточных систем одАНЛДУ. С вычислительной точки зрения это может приводить к большим расходам памяти в процессе преобразования.

Пример 1.11. рассмотрим систему одАНЛДУ

$$\begin{cases} x_1 + x_2 + x_3 &= 0 \\ x_4 &= x_2 + x_3 \end{cases}.$$

Входе преобразований системы одАНЛДУ получаем уравнение $x_4 = x_2 + x_3$. Данное уравнение имеет базис Гильберта

$$\mathcal{H}^{(1)} = \left\{ \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} \right\} \begin{array}{l} : x_2 \\ : x_3 \\ : x_4 \end{array}$$

В то же время исходная система одАНЛДУ имеет только тривиальное решение $x = \mathbb{O}^m$.

Замечание 1.2. Более типичными представляются случаи, когда исходная система одАНЛДУ имеет базис Гильберта, который не меньше, чем у промежуточных систем. Это следует из свойств уравнения Фробениуса с единичными коэффициентами (см. п. 1.1).

Пример 1.12 (Приведение системы одАНЛДУ к виду (1.15)). Пусть задана система одАНЛДУ:

$$\begin{cases} x_1 + x_3 &= 3x_4 + x_5 \\ x_2 &= 5x_1 + 7x_3 + 2x_4 + 3x_5 \\ x_4 + x_5 &= 3x_5 \end{cases}$$

Строим уравнение вида (1.12):

$$4x_1 - x_2 + 6x_3 + 4x_4 + 6x_5 = 0.$$

Применим преобразование системы ($c_2 = -1$). Следовательно, получаем правило вычисления значения переменной x_2:

$$x_2 = 5x_1 + 7x_3 + 2x_4 + 3x_5 = T_1(x_1, x_3, x_4, x_5).$$

Исключаем уравнение 2 и неизвестное x_2 из рассмотрения и получаем систему вида:

$$\begin{cases} x_1 + x_3 & = & 3x_4 + x_5 \\ x_4 + x_5 & = & 3x_5 \end{cases}$$

Пересчитываем уравнение (1.12):

$$-x_1 - x_3 + 2x_4 + 3x_5 = 0.$$

Применим преобразование системы ($c_1 = -1$ и $c_3 = -1$). Получаем правило для вычисления переменных x_1 и x_3:

$$x_1 + x_3 = 3x_4 + x_5 = T_2(x_4, x_5).$$

Исключаем уравнение 1 и неизвестные x_1, x_3 из рассмотрения и в итоге у нас остается только одно уравнение. Следовательно, мы привели систему одАНЛДУ к виду (1.15):

$$\begin{cases} x_4 + x_5 & = & 3x_5 \\ x_1 + x_3 & = & T_2(x_4, x_5) \\ x_2 & = & T_1(x_1, x_3, x_4, x_5) \end{cases}$$

Нахождение базиса Гильберта для данной системы выглядит следующим образом:

$$x_4 = 2x_5.$$

Базис Гильберта для данного уравнения:

$$\mathcal{H}^{(2)} = \left\{ \begin{pmatrix} 2 \\ 1 \end{pmatrix} \right\} \begin{array}{l} : x_4 \\ : x_5 \end{array}$$

Применяем правило T_2:

$$x_1 + x_3 = T_2(x_4, x_5) = 3 * 2 + 1 * 1 = 7.$$

Это уравнение Фробениуса, которое имеет 7 решений:

$$\mathcal{H}^{(1)} = \left\{ \begin{pmatrix} 7 \\ 0 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 6 \\ 1 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 5 \\ 2 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 4 \\ 3 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ 4 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 2 \\ 5 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 6 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 7 \\ 2 \\ 1 \end{pmatrix} \right\} \begin{array}{l} : x_1 \\ : x_3 \\ : x_4 \\ : x_5 \end{array}$$

Применяем правило T_1:

$$x_2 = T_1(x_1, x_3, x_4, x_5) = 5x_1 + 7x_3 + 2x_4 + 3x_5.$$

Подставляя $\mathcal{H}^{(1)}$, получаем базис Гильберта для исходной системы одАНЛДУ.

$$\mathcal{H} = \left\{ \begin{pmatrix} 7 \\ 42 \\ 0 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 6 \\ 44 \\ 1 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 5 \\ 46 \\ 2 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 4 \\ 48 \\ 3 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ 50 \\ 4 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 2 \\ 52 \\ 5 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 54 \\ 6 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 56 \\ 7 \\ 2 \\ 1 \end{pmatrix} \right\}$$

Пример 1.13 (Приведение системы одАНЛДУ к виду (1.16)). Пусть задана система одАНЛДУ:

$$\begin{cases} x_1 + x_3 & = & x_1 + 3x_3 \\ x_2 & = & x_2 + 5x_3 \\ x_4 & = & 2x_1 + 3x_2 + 7x_3 \end{cases}$$

Строим уравнение вида (1.12):

$$2x_1 + 3x_2 + 14x_3 - x_4 = 0.$$

Применим преобразование системы ($c_4 = -1$). Получаем правило для вычисления значения переменной x_4:

$$x_4 = 2x_1 + 3x_2 + 7x_3 = T_1(x_1, x_2, x_3).$$

Исключаем уравнение 3 и переменную x_4 из рассмотрения и получаем систему вида:

$$\begin{cases} x_1 + x_3 & = & x_1 + 3x_3 \\ x_2 & = & x_2 + 5x_3 \end{cases}$$

Пересчитываем уравнение (1.12):

$$7x_3 = 0.$$

Больше нельзя применять преобразование системы. Полагаем переменную $x_3 = 0$, т.к. $c_3 > 0$. В итоге данная система приведена к виду (1.16):

$$\begin{cases} x_1 &= x_1 \\ x_2 &= x_2 \\ x_3 &= 0 \\ x_4 &= T_1(x_1, x_2, x_3) \end{cases}$$

Нахождение базиса Гильберта выглядит так: решаем систему уравнений

$$\begin{cases} x_1 = x_1 \\ x_2 = x_2 \end{cases}$$

Базис Гильберта для этой системы:

$$\mathcal{H}^1 = \left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\} \begin{array}{l} : x_1 \\ : x_2 \end{array}$$

Учитывая, что $x_3 = 0$ применяем правило для нахождения x_4 и получаем базис Гильберта для системы одАНЛДУ:

$$\mathcal{H} = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 2 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \\ 3 \end{pmatrix} \right\}$$

Выводы к главе

В этой главе были представлены следующие основные положения по теории систем одАНЛДУ.

1. Выполнено развитие понятийного аппарата систем АНЛДУ. Введены компактные обозначения для однородных систем АНЛДУ и симметричных систем одАНЛДУ.

2. Исследован вид базиса Гильберта для системы одАНЛДУ, состоящей из одного уравнения (теорема 1.1).

3. Получен ряд свойств для систем содАНЛДУ. На основе графового представления исследован вопрос о разрешимости таких систем, получены и доказаны основные свойства их базисных решений (теоремы 1.2–1.5).

4. Предложено преобразование произвольной системы одАНЛДУ к виду системы содАНЛДУ или к случаю системы одАНЛДУ с одним уравнением. Это ключевой результат данной главы, он был сформулирован и доказан в виде теоремы 1.6.

2. Генерация систем одАНЛДУ

С точки зрения ключевых приложений генерации систем одАНЛДУ — тестирования, экспериментального и сравнительного анализа, необходимо наличие библиотеки алгоритмов генерации. Это позволяет учесть разнообразие ограничений, накладываемых приложением на процесс и результат генерации, выбирая наиболее подходящий алгоритм из этой библиотеки. Единый теоретический аппарат, на основе которого возможна разработка таких алгоримов, был представлен в предыдущей главе. В настоящей главе нами предлагаются и обосновываются пять алгоритмов генерации систем одАНЛДУ, позволяющих решать широкий класс задач для упомянутых приложений.

В п. 2.1 выполняется постановка задачи генерации и представлена ее специфика для приложений тестирования и экспериментального анализа алгоритмов решения систем одАНЛДУ. В пп. 2.2, 2.3 и 2.4 предлагаются три алгоритма генерации частных классов систем одАНЛДУ. В п. 2.5 представлен алгоритм генерации систем содАНЛДУ. Алгоритм генерации произвольной системы одАНЛДУ дается в п. 2.6.

2.1. Задача генерации и ее приложения

Впервые системы одАНЛДУ были определены и исследованы в [5] на основе взаимосвязи систем НЛДУ и формальных грамматик. Там же был предложен и обоснован синтаксический алгоритм решения произвольной системы одАНЛДУ и показана полиномиальная сложность для наихудшего случая. Проведенные эксперименты с реализацией syntactic solver этого алгоритма также подтверждают его существенную эффективность для случая систем большой размерности.

В то же время, оставался открытым вопрос о качестве данной реализации алгоритма с точки зрения наличия в ней ошибок. Необходимо также комплексное экспериментальное исследование алгоритма для получения более точных оценок эффективности синтаксического алгоритма на практике и сравнение его с альтернативными решателями.

Таким образом, на примере данной практической задачи видно, что в общем случае необходим программный инструментарий для автоматической генерации:

1) тестовых систем одАНЛДУ с соответствующими базисами Гильберта;

2) выборки систем одАНЛДУ из заданного частного класса.

Первое необходимо для успешного решения задачи покрывающего системного тестирования реализации некоторого алгоритма решения системы одАНЛДУ, используя проверку правильности полученного алгоритмом решения. При этом тестовые системы могут генерироваться как для некоторых частных классов (тестирование на специфичных тестах), так и для общего случая — потенциально возможна генерация любой системы одАНЛДУ. Отметим, что используя понятие эквивалентных систем одАНЛДУ можно увеличить эффективность тестирования за счет быстрого получения новых тестовых систем путем перенумерации неизвестных в сгенерированной системе.

Второе важно при выполнении экспериментального анализа эффективности алгоритма решения и построения карты его эффективности для различных классов систем одАНЛДУ. Также это может использоваться для сравнительного анализа эффективности нескольких алгоритмов решения систем одАНЛДУ.

Тем самым, мы приходим к постановке следующей общей задачи генерации системы одАНЛДУ вида

$$\mathrm{E}^m(I^n)x = Ax \tag{2.1}$$

при заданных ограничениях на процесс и результат генерации. В простейшем случае, в качестве ограничений могут выступать лишь число уравнений n и неизвестных m. В более сложных случаях необходимо учитывать, например, ограничения на число базисных решений q или на элементы матрицы A. Основными ограничениями на сам процесс генерации обычно выступают требования на максимально допустимое время генерации и используемый объем памяти.

С вычислительной точки зрения данная задача не является такой простой, как может показаться на первый взгляд. Она не может быть сведена к непосредственной генерации матриц Е и A. Во-первых, в таком случае будут порождаться системы одАНЛДУ, которые, как правило, несовместны (имеют пустой базис Гильберта) [5]. Во-вторых, для тестирования надо знать базис Гильберта генерируемой системы одАНЛДУ и использовать его при проверке решений, полученных тестируемым алгоритмом. В-третьих, экспериментальный и сравнительный анализы требуют системы из заданного частного класса. Соответствующие ограничения требуют информации, например, о форме базиса Гильберта и матрицы A, числе базисных решений и т.п.

В силу этого, задача генерации систем одАНЛДУ сталкивается с таким же проблемами вычислительной сложности, что и задача решения. В п. 1.4 было предложено

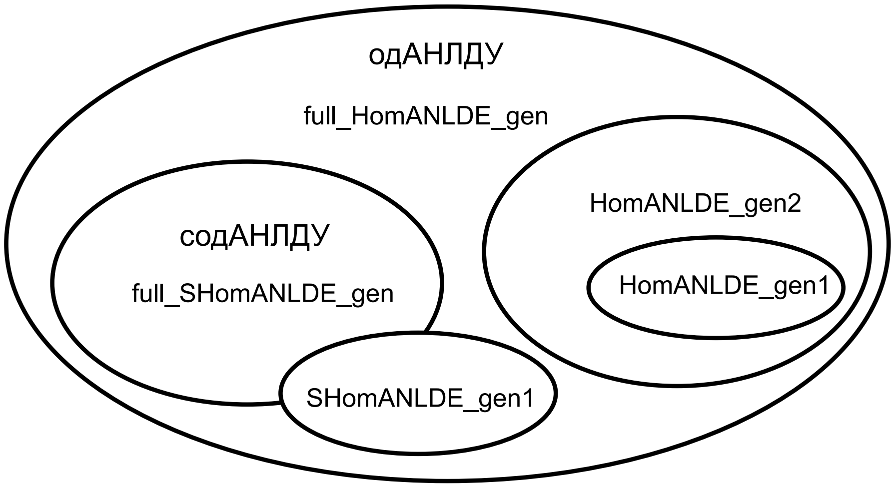


Рис. 4. Структура разбиения множества систем одАНЛДУ на классы, в соответствии с предлагаемыми алгоритмами генерации.

преобразование систем одАНЛДУ, которое позволяет решить задачу генерации в общем случае. Нами разработаны четыре алгоритма генерации частных классов систем одАНЛДУ (включая такой класс как системы содАНЛДУ) и алгоритм генерации произвольной системы одАНЛДУ. Общая схема взаимоотношения этих алгоритмов в зависимости от генерируемых ими классов систем представлена на рис. 4. Алгоритмы генерации частных классов предлагают более простой метод генерации по сравнению с алгоритмом генерации произвольной системы одАНЛДУ.

В простейшем случае алгоритм генерации систем получает в качестве входных параметров лишь размерности генерируемой системы (число уравнений n, число неизвестных m, $0 < n \leqslant m$). Выходными параметрами являются разбиение множества неизвестных I^n, матрица коэффициентов правой части A, матрица коэффициентов левой части $\mathrm{E}^m(I^n)$ и базис Гильберта $\mathcal{H}$.

В качестве дополнительных ограничений на результат генерации могут выступать следующие:

— Максимально возможное значение коэффициентов матрицы A (в общем случае, ограничение на некоторую норму $||A||$);

— Максимально возможное значение компонент базисных решений (в общем случае, ограничение на некоторую норму $||\mathcal{H}||$);

— Максимально допустимое число базисных решений q.

Отметим, что возможны и другие ограничения, например, задание точного значения для числа базисных решений. Однако, ограничения в форме верхних границ существенно более просты для учета при построении алгоритма и, в то же время, достаточно гибкие, чтобы полученный алгоритм можно было использовать в приложениях.

Далее, при описании алгоритмов действия “генерируем случайным образом матрицу” и “выбираем случайным образом число” будут означать построение соответствующих объектов на основе стандартного датчика псевдослучайных чисел.

2.2. Алгоритм генерации систем одАНЛДУ с полностью единичным базисом Гильберта SHomANLDE_gen1 (аналог преобразования Гаусса — Жордано)

Пусть n и m фиксированы, причем $0 < n \leqslant m$. Идея алгоритма заключается в построении системы одАНЛДУ, базис Гильберта которой состоит из n стандартных единичных векторов пространства $\mathbb{Z}^m$:

$$\mathcal{H} = \{e_1, e_2, \ldots, e_n\}. \tag{2.2}$$

Далее такой базис будем называть *единичным базисом Гильберта*.

Матрицу $\mathrm{E}^m(I^n)$ коэффициентов левой части строим как $\mathrm{E}^m(I^n) = (\mathbb{I} \mid B)$, где $\mathbb{I}$ — стандартная единичная $(n \times n)$-матрица ($\mathbb{I} = \mathrm{E}^n(\{1\}, \{2\}, \ldots, \{n\})$), а $B \in \{0,1\}^{n\times(m-n)}$ — произвольная $(0,1)$-матрица, для которой выполняется следующее условие:

$$\forall k \quad \sum_{i=1}^{n} B_{ik} = 1. \tag{2.3}$$

Матрицу A коэффициентов правой части строим как $A = (\mathbb{I} \mid B + \Delta)$, где матрица $\Delta \in \mathbb{Z}_+^{n\times(m-n)}$ удовлетворяет следующему условию:

$$\forall\, i \quad \sum_{j=1}^{m-n} \Delta_{ij} x_{j+n} = 0 \quad \Longleftrightarrow \quad x = \mathbb{O}. \tag{2.4}$$

Очевидно, что условие (2.4) гарантируется при выполнении следующего условия:

$$\forall j \quad \sum_{i=1}^{n} \Delta_{ij} > 0\,, \tag{2.5}$$

использование которого проще с точки зрения генерации матрицы Δ.

Таким образом, в матричном представлении генерируемая система одАНЛДУ выглядит следующим образом

$$(\mathbb{I} \mid B)x = (\mathbb{I} \mid B + \Delta)x. \tag{2.6}$$

Алгоритм 1 Генерация системы одАНЛДУ с полностью единичным базисом Гильберта SHomANLDE_gen1

Вход: n — число уравнений, m — число неизвестных, $n \leqslant m$.

Выход: $(\mathrm{E}^m, A, \mathcal{H})$ для системы (2.6).

1: $\mathcal{H} = \{e_1, e_2, \ldots, e_n\}$;

2: генерируем случайным образом матрицу $B \in \{0,1\}^{n\times(m-n)}$, удовлетворяющую (2.3);

3: генерируем случайным образом матрицу $\Delta \in \mathbb{Z}_+^{n\times(m-n)}$, удовлетворяющую (2.5);

4: $\mathrm{E}^m := (\mathbb{I} \mid B)$; $\quad A := (\mathbb{I} \mid B + \Delta)$. $\qquad \triangleright$ $\mathbb{I} \in \{0,1\}^{n\times n}$ — единичная матрица

Формальное описание шагов генерации представлено в Алгоритме 1. Отметим, что для генерации матрицы B (шаг 2) достаточно лишь для каждого из $m-n$ столбцов случайным образом выбрать номер строки, в которой будет стоять единица, остальные элементы — нулевые. Для генерации матрицы Δ (шаг 3) необходимо обеспечить положительность хотя бы одного элемента в каждом из $m-n$ столбцов, остальные элементы полагаются произвольными нетрицательными целыми.

Теорема 2.1. *Базис Гильберта системы* (2.6) *есть* (2.2)*.*

Доказательство. Переписывая систему (2.6) получаем

$$(\mathbb{I} \mid B)x = (\mathbb{I} \mid (B+\Delta))x \quad \Rightarrow \quad (\mathbb{O} \mid \Delta)x = \mathbb{O}.$$

Тем самым, $\mathbb{O}x' + \Delta x'' = \mathbb{O}$, где $x' = (x_1, \ldots, x_n)^\mathsf{T}$, $x'' = (x_{n+1}, \ldots, x_m)^\mathsf{T}$. Система $\Delta x'' = \mathbb{O}$ имеет только нулевое решение $x'' = \mathbb{O} \in \mathbb{Z}^{m-n}$ в силу условия (2.4). Решениями системы $\mathbb{O}x' = \mathbb{O}$ очевидно являются любые вектора из Z_+^n, а, значит, минимальные решения определяются базисом (2.2). $\square$

Согласно теореме 1.6 задача нахождения базиса Гильберта для системы (2.6) сводится к задаче нахождения базиса Гильберта для системы вида (1.16)

$$\mathbb{O}x = \mathbb{O}.$$

Данный алгоритм является аналогом преобразования Гаусса — Жордано, т.к. матрица коэффициентов A приведена к матрице, содержащей единичную подматрицу.

Пример 2.1. **Вход:** $n = 3, \quad m = 4$

Шаг 1: $\mathcal{H} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$. **Шаг 2:** $B = (1,0,0)^\mathsf{T}$. **Шаг 3:** $\Delta = (2,2,3)^\mathsf{T}$.

Шаг 4: $\mathrm{E}^4 = (\mathbb{I} \mid B) = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad A = (\mathbb{I} \mid B + \Delta) = \begin{pmatrix} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{pmatrix}.$

2.3. Алгоритм генерации систем одАНЛДУ с частично единичным базисом Гильберта HomANLDE_gen1 (аналог преобразования Гаусса)

Пусть n и m фиксированы, причем $0 < n < m$. Идея алгоритма заключается в построении системы одАНЛДУ, базис Гильберта которой частично состоит из единичных векторов:

$$\mathcal{H} = \text{ множество столбцов матрицы } (F \mid \mathbb{I})^{\mathsf{T}}, \tag{2.7}$$

где $\mathbb{I}$ — стандартная единичная $((m-n) \times (m-n))$-матрица ($\mathbb{I} = \mathrm{E}^{m-n}(\{1\}, \{2\}, \ldots, \{m-n\})$), а $F \in \mathbb{Z}_+^{(m-n)\times n}$. Элементы матрицы F будут определяться после генерации коэффициентов системы одАНЛДУ на основе подстановки базиса Гильберта. Далее такой базис будем называть *частично единичным базисом Гильберта.*

Матрицу E^m коэффициентов левой части строим как $\mathrm{E}^m = (\mathbb{I} \mid B)$, где $\mathbb{I}$ — стандартная единичная $(n \times n)$-матрица ($\mathbb{I} = \mathrm{E}^n(\{1\}, \{2\}, \ldots, \{n\})$), а $B \in \{0,1\}^{n\times(m-n)}$ — произвольная $(0,1)$-матрица, удовлетворяющая условию (2.3). Матрицу A коэффициентов правой части строим как $A = (D^+ \mid B + \Delta)$, где $D^+ \in \mathbb{Z}_+^{n\times n}$, т. ч.

$$D^+_{ij} = \begin{cases} 0 & \text{, если } i \geqslant j, \\ d_{ij} \in \mathbb{Z}_+ & \text{, если } i < j+1, \\ d_{ij} \in \mathbb{N} & \text{, если } i = j+1, \end{cases} \tag{2.8}$$

а матрица $\Delta \in \mathbb{Z}_+^{n\times(m-n)}$ удовлетворяет следующему условию:

$$\forall j \quad \Delta_{nj} > 0. \tag{2.9}$$

Таким образом, в матричном представлении система одАНЛДУ выглядит следующим образом

$$(\mathbb{I} \mid B)x = (D^+ \mid B + \Delta)x. \tag{2.10}$$

Формальное описание шагов генерации представлено в Алгоритме 2. Реализация шагов 1–3 генерации матриц D^+, B и Δ очевидна. Реализация шага 4 основана на постановке базиса Гильберта, начиная с единичного базиса, и представлена в доказательстве теоремы 2.2.

Алгоритм 2 Генерация системы одАНЛДУ с частично единичным базисом Гильберта HomANLDE_gen1

Вход: n — число уравнений, m — число неизвестных, $n < m$.

Выход: $(\mathrm{E}^m, A, \mathcal{H})$ для системы (2.10).

1: генерируем случайным образом матрицу $D^+ \in \mathbb{Z}_+^{n\times n}$, удовлетворяющую (2.8);

2: генерируем случайным образом матрицу $B \in \{0,1\}^{n\times(m-n)}$, удовлетворяющую (2.3);

3: генерируем случайным образом матрицу $\Delta \in \mathbb{Z}_+^{n\times(m-n)}$, удовлетворяющую (2.9);

4: вычисляем матрицу $F \in \mathbb{Z}_+^{(m-n)\times n}$;

5: $\mathrm{E}^m := (\mathbb{I} \mid B)$; $A := (D^+ \mid B + \Delta)$; $\mathcal{H} := \{F \mid \mathbb{I}\}^\mathsf{T}$. $\triangleright$ $\mathbb{I} \in \{0,1\}^{n\times n}$ — единичная матрица

Теорема 2.2. *Базис Гильберта системы* (2.10) *есть* (2.7)*.*

Доказательство. Согласно теореме 1.6 произведем преобразование системы (2.1)

$$(\mathbb{I} \mid B)x = (D^+ \mid (B+\Delta))x \quad \Rightarrow \quad ((D^+ - \mathbb{I}) \mid \Delta)x = \mathbb{O}.$$

Тем самым, $(D^+ - \mathbb{I})x' + \Delta x'' = \mathbb{O}$, где $x' = (x_1, \dots, x_n)^\mathsf{T}$, $x'' = (x_{n+1}, \dots, x_m)^\mathsf{T}$. В силу специфики построения матрицы $(D^+ - \mathbb{I})$ и условия (2.9) с помощью преобразования произвольной системы одАНЛДУ система будет преобразована к уравнению вида (1.15):

$$x_n = \sum_{j=1}^{m-n} \Delta_{nj} x_{j+n}.$$

Согласно теореме 1.1 (см. п. 1.2, с. 16) базис Гильберта данного уравнения есть

$$\mathcal{H} = \text{ множество столбцов матрицы } \{F' \mid \mathbb{I}\}^\mathsf{T},$$

где F' определяется решениями уравнений Фробениуса с единичными коэффициентами. Вычисляя базис Гильберта в обратном порядке получаем (2.7). $\square$

Согласно теореме 1.6 задача нахождения базиса Гильберта для системы (2.10) сводится к задаче нахождения базиса Гильберта для уравнения вида (1.15)

$$x_n = \sum_{j=1}^{m-n} \Delta_{nj} x_{j+n}.$$

Данный алгоритм является аналогом преобразования Гаусса, т.к. матрица коэффициентов приведена к ступенчатому виду.

Пример 2.2. **Вход:** $n = 3, \quad m = 5$

Шаг 1: $D^+ = \begin{pmatrix} 0 & 2 & 3 \\ 0 & 0 & 5 \\ 0 & 0 & 0 \end{pmatrix}$. **Шаг 2:** $B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$. **Шаг 3:** $\Delta = \begin{pmatrix} 0 & 0 \\ 4 & 2 \\ 3 & 4 \end{pmatrix}$.

Шаг 4: $F = \begin{pmatrix} 47 & 19 & 3 \\ 52 & 20 & 4 \end{pmatrix}$. **Шаг 5:** $\mathrm{E}^5 = (E \mid B) = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$,

$$A = (D^+ \mid B + \Delta) = \begin{pmatrix} 0 & 2 & 3 & 1 & 0 \\ 0 & 0 & 5 & 4 & 3 \\ 0 & 0 & 0 & 3 & 4 \end{pmatrix}, \quad \mathcal{H} = (F \mid E)^\mathsf{T} = \begin{pmatrix} 47 & 52 \\ 19 & 20 \\ 3 & 4 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

2.4. Алгоритм генерации систем одАНЛДУ с обобщенным частично единичным базисом Гильберта HomANLDE_gen2

Пусть n и m фиксированы, причем $0 < n < m$. Формальное описание алгоритма представлено в Алгоритме 3.

Алгоритм 3 Генерация системы одАНЛДУ с обобщенным частично единичным базисом Гильберта HomANLDE_gen2

Вход: n — число уравнений, m — число неизвестных, $n < m$.

Выход: $(\mathrm{E}^m, A, \mathcal{H})$.

1: генерируем случайным образом число $p \in \{0, \ldots, m - n - 1\}$ и последовательность $0 < k_1 < \cdots < k_n = n + p + 1$;

2: генерируем случайным образом матрицу $B \in \{0, 1\}^{n \times (m-n-p)}$, удовлетворяющую (2.3);

3: генерируем случайным образом матрицу $\Delta \in \mathbb{Z}_+^{n \times (m-n-p)}$, удовлетворяющую (2.9);

4: строим матрицу E^m вида (2.11);

5: генерируем случайным образом матрицу A вида (2.12);

6: с помощью преобразования теоремы 1.6 строим базис $\mathcal{H}$.

Идея алгоритма заключается в построении системы одАНЛДУ, которая может быть преобразована (теорема 1.6) к уравнению вида (1.15). Как и в предыдущем алгоритме, базис Гильберта вычисляется после генерации коэффициентов системы. Структура базиса будет представлена ниже.

Матрицу E^m коэффициентов левой части строим как

$$\mathrm{E}^m = \left(\ \mathbb{I}' \ \middle| \ B \ \right), \tag{2.11}$$

где $\mathbb{I}' = \mathrm{E}^{n+p}(\{1,\ldots,k_1-1\},\{k_1,\ldots,k_2-1\},\ldots,\{k_{n-1},\ldots,k_n-1\})$ или

$$\mathbb{I}' = \begin{pmatrix} 1 & \ldots & 1 & 0 & \ldots\ldots\ldots\ldots\ldots\ldots\ldots & & & & 0 \\ 0 & \ldots & 0 & 1 & \ldots & 1 & 0 & \ldots\ldots\ldots\ldots & & 0 \\ & & & & & & & & & \\ \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots \\ & & & & & & & & & \\ 0 & \ldots\ldots\ldots\ldots\ldots\ldots\ldots & & & & & 0 & 1 & \ldots & 1 \end{pmatrix},$$

$B \in \{0,1\}^{n\times(m-n-p)}$ — произвольная $(0,1)$-матрица, удовлетворяющая условию (2.3), а $0 < k_1 < k_2 < \cdots < k_n = n+p+1$ — некоторая возрастающая последовательность положительных целых чисел ($0 \leqslant p \leqslant m-n-1$ — произвольная константа).

Матрицу A коэффициентов правой части строим как

$$A = \left(\ \mathbb{O} \ \middle| \ A_+ \ \middle| \ B+\Delta \ \right), \tag{2.12}$$

где

$$A_+ = \begin{pmatrix} a_{1k_1} & \ldots\ldots\ldots\ldots & & & a_{1(n+p)} \\ 0 & \ldots & 0 & a_{2k_2} \ \ldots & a_{2(n+p)} \\ & & & & \\ \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots \\ 0 & \ldots\ldots\ldots\ldots & & & 0 \end{pmatrix},$$

$a_{ij} \geqslant 0$, $a_{ik_i} > 0$ и матрица $\Delta \in \mathbb{Z}_+^{n\times(m-n-p)}$ удовлетворяет условию (2.9).

Таким образом, в матричном представлении система одАНЛДУ выглядит следующим образом

$$\left(\ \mathbb{I}' \ \middle| \ B \ \right) x = \left(\ \mathbb{O} \ \middle| \ A_+ \ \middle| \ B+\Delta \ \right) x. \tag{2.13}$$

Базис Гильберта $\mathcal{H}$:

$$\mathcal{H} = \text{ множество столбцов матрицы } (F|\mathbb{I}')^{\mathsf{T}} \tag{2.14}$$

может быть представлен в виде $\{h^{(s)}\}_{s=1}^{q}$, где q — размерность базиса. Каждый вектор $h^{(s)}$ удовлетворяет следующиему набору условий:

$$\begin{cases} h_i^{(s)} \in \{0,1\}, \text{ если } i \geqslant k_n \text{ и } \sum\limits_{j=k_n}^{m} h_j^s = 1, \\ h_i^{(s)} = \sum\limits_{j=k_r}^{m} a_{rj} h_j^{(s)} - \sum\limits_{j\in\overline{\{k_{r-1},k_r-1\}}\setminus i} h_j, \text{ если } i < k_n, \text{ где } r \text{ т.ч. } i \in I_r. \end{cases}$$

Далее такой базис будем называть *обобщенным частично единичным базисом Гильберта.*

Теорема 2.3. *Базис Гильберта системы* (2.13) *есть* (2.14).

Доказательство. Согласно теореме 1.6 произведем преобразование системы (2.13)

$$(\mathbb{I}' \mid B)x = (\mathbb{O} \mid A_+ \mid B + \Delta)x \quad \Rightarrow \quad ((\mathbb{O} \mid A_+) - \mathbb{I}' \mid \Delta)x = \mathbb{O}.$$

Тем самым, $((\mathbb{O} \mid A_+) - \mathbb{I})x' + \Delta x'' = \mathbb{O}$, где $x' = (x_1, \ldots, x_{n+p})^\mathsf{T}$, $x'' = (x_{n+p+1}, \ldots, x_m)^\mathsf{T}$. В силу специфики построения матрицы $((\mathbb{O} \mid A_+) - \mathbb{I})$ и условия (2.9) с помощью преобразования произвольной системы одАНЛДУ система будет преобразована к уравнению вида (1.15):

$$\sum_{i=k_{n-1}}^{k_n - 1} x_i = \sum_{j=1}^{m-n-p} \Delta_{nj} x_{j+n+p}.$$

Согласно теореме 1.1 (см. п. 1.2, с. 16) базис Гильберта данного уравнения есть

$$\mathcal{H} = \text{ множество столбцов матрицы } \{F' \mid \mathbb{I}'\}^\mathsf{T},$$

где F' определяется решениями уравнений Фробениуса с единичными коэффициентами. Вычисляя базис Гильберта в обратном порядке получаем (2.14). □

Согласно теореме 1.6 задача нахождения базиса Гильберта для системы (2.13) сводится к задаче нахождения базиса Гильберта для уравнения (1.15)

$$\sum_{i=k_{n-1}}^{k_n - 1} x_i = \sum_{j=1}^{m-n-p} \Delta_{nj} x_{j+n+p}.$$

Данный алгоритм является расширенным аналогом преобразования Гаусса, т.к. матрица коэффициентов A приведена к более общему ступенчатому виду.

Пример 2.3. **Вход:** $n = 3, \quad m = 6$.

Шаг 1: $p = 1$, $k_1 = 3$, $k_2 = 4$, $k_3 = 5$. **Шаг 2:** $B = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$. **Шаг 3:** $\Delta = \begin{pmatrix} 3 & 2 \\ 3 & 3 \\ 3 & 3 \end{pmatrix}$.

Шаг 4: $\mathrm{E}^6 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}$. **Шаг 5:** $A = \begin{pmatrix} 0 & 0 & 0 & 1 & 3 & 2 \\ 0 & 0 & 0 & 5 & 4 & 3 \\ 0 & 0 & 0 & 0 & 3 & 4 \end{pmatrix}$.

Шаг 6: $\mathcal{H} = \begin{pmatrix} 6 & 5 & 4 & 3 & 2 & 1 & 0 & 5 & 4 & 3 & 2 & 1 & 0 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 0 & 1 & 2 & 3 & 4 & 5 \\ 18 & 18 & 18 & 18 & 18 & 18 & 18 & 18 & 18 & 18 & 18 & 18 & 18 \\ 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$.

2.5. Алгоритм генерации симметричных систем одАНЛДУ full_SHomANLDE_gen

Пусть m и n фиксированы, причем $0 < n < m$. Формальное описание алгоритма представлено в Алгоритме 4.

Алгоритм 4 Генерация системы содАНЛДУ с базисом Гильберта full_SHomANLDE_gen

---

Вход: n — число уравнений, m — число неизвестных, $0 < n < m$, оганичения на размерность базиса Гильберта.

Выход: $(\mathrm{E}^m(I^n), \mathrm{E}^m(J^n), \mathcal{H})$.

1: Генерируем случайным образом матрицы $\mathrm{E}^m(I^n)$ и $\mathrm{E}^m(I^n)$.

2: Находим базис Гильберта $\mathcal{H}$.

3: **if** базис Гильберта пуст **then**

4: Выбираем случайным образом неизвестное x_i, $1 \leqslant i \leqslant m$.

5: p т.ч. $J_i^n = p$, z т.ч. $I_i^n = 1$. ▷ номера уравнений, в которых встречается неизвестное x_i.

6: $\mathrm{E}^m(J^n)_{pk} = 0$, $\mathrm{E}^m(J^n)_{zk} = 1$, $\mathcal{H} = \{e_i\}$.

7: **end if**

8: **if** количество базисных векторов > максимума **then**

9: уравнение p с минимальным номером среди уравнений i, для которых выполняется условие $\forall j \sum I_j > \sum I_i$.

10: **repeat**

11: Выбираем произвольным образом $0 < z < m$ т.ч. $z \in J_k$, $k \neq p$.

12: $\mathrm{E}^m(J^n)_{kz} = 0$, $\mathrm{E}^m(J^n)_{pz} = 1$.

13: **until** размерность базиса Гильберта больше максимума.

14: **end if**

Данный метод генерирует систему содАНЛДУ (1.8) с учетом накладываемых на размерность базиса Гильберта ограничений. Алгоритм 4 позволяет генерировать системы содАНЛДУ, даже в случае, если одна из матриц разбиения (например, $\mathrm{E}^m(I^n)$) фиксирована.

Для реализации шага 2 используется алгоритм решения систем содАНЛДУ, предложеный в п. 3.1. В случае пустого базиса Гильберта, система преобразуется для получения базисного решения в виде единичного вектора (шаги 3–7). Дальнейшие действия (шаги 8–14) основаны на методе построения системы содАНЛДУ с минимальным базисом Гильберта (см. доказательство теоремы 1.5).

Согласно теореме 1.4 базис Гильберта состоит из $\{0,1\}$-векторов.

Пример 2.4. **Вход:** $n = 3$, $m = 7$, максимум $= 4$.

Шаг 1. $\mathrm{E}^7(I^3) = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$, $\mathrm{E}^7(J^3) = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}$.

Шаг 2. $\mathcal{H} = \left\{ \begin{pmatrix} 0\\0\\1\\0\\0\\0\\0 \end{pmatrix}, \begin{pmatrix} 0\\0\\0\\1\\0\\0\\0 \end{pmatrix}, \begin{pmatrix} 1\\0\\0\\0\\1\\0\\0 \end{pmatrix}, \begin{pmatrix} 1\\0\\0\\0\\0\\0\\1 \end{pmatrix}, \begin{pmatrix} 0\\1\\0\\0\\1\\1\\0 \end{pmatrix}, \begin{pmatrix} 0\\1\\0\\0\\0\\1\\1 \end{pmatrix} \right\}$.

Шаг 3. Базис Гильберта не пуст. Переходим на Шаг 8.

Шаг 8. Базис Гильберта больше максимума.

Шаг 9. $p = 2$. **Шаг 11.** $z = 3$, $k = 3$. **Шаг 12.** $\mathrm{E}^7(J^3) = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$,

$$\mathcal{H} = \left\{ \begin{pmatrix} 0\\0\\0\\1\\0\\0\\0 \end{pmatrix}, \begin{pmatrix} 1\\0\\0\\0\\1\\0\\0 \end{pmatrix}, \begin{pmatrix} 1\\0\\0\\0\\0\\0\\1 \end{pmatrix}, \begin{pmatrix} 0\\1\\0\\0\\1\\1\\0 \end{pmatrix}, \begin{pmatrix} 0\\1\\0\\0\\0\\1\\1 \end{pmatrix}, \begin{pmatrix} 0\\0\\1\\0\\0\\1\\0 \end{pmatrix} \right\}.$$

Шаг 13. Базис Гильберта больше максимума. Переходим на Шаг 11.

Шаг 11. $z = 6$, $k = 3$. **Шаг 12.** $\mathrm{E}^7(J^3) = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$,

$$\mathcal{H} = \left\{ \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \right\}.$$

Шаг 13. Размер базиса Гильберта равен максимуму. Выход из цикла.

2.6. Алгоритм генерации произвольной системы одАНЛДУ full_SHomANLDE_gen

Пусть n и m фиксированы, причем $0 < n \leqslant m$. Формальное описание алгоритма генерации представлено в Алгоритме 5.

Алгоритм 5 Генерация произвольной системы одАНЛДУ с базисом Гильберта full_HomANLDE_gen

Вход: n — число уравнений, m — число неизвестных, $n \leqslant m$.

Выход: $(\mathrm{E}^m, A, \mathcal{H})$.

1: генерируем случайным образом числа $0 \leqslant q \leqslant n$, $0 \leqslant p < m - q$, $0 \leqslant t < m - q - p$ и последовательность $0 < k_1 < \dots < k_q = q + p + 1$;

2: генерируем случайным образом матрицы $B \in \{0,1\}_+^{q \times (m-q-p)}$, $B' \in \{0,1\}_+^{(n-q)\times(m-q-p-t)}$, $B'' \in \{0,1\}_+^{(n-q)\times t}$, удовлетворяющие условию (2.17);

3: генерируем случайным образом матрицу $B''' \in \{0,1\}^{(n-q)\times(m-q-p-t)}$, удовлетворяющую условию (2.20);

4: генерируем случайным образом матрицу $\Delta \in \mathbb{Z}_+^{n\times(m-q-p)}$, удовлетворяющую (2.21);

5: генерируем случайным образом матрицу $\Delta' \in \mathbb{Z}_+^{(n-q)\times t}$, удовлетворяющую условию (2.22);

6: генерируем случайным образом матрицу $A' \in \mathbb{Z}^{q\times(q+p)}$, удовлетворяющую (2.19);

7: генерируем случайным образом матрицу $E' \in \mathbb{Z}^{q\times(q+p)}$, удовлетворяющую (2.16);

8: строим матрицу E^m вида (2.15);

9: строим матрицу A вида (2.18);

10: с помощью преобразования теоремы 1.6 строим матрицу $\mathcal{H}$.

Данный метод строит произвольную систему одАНЛДУ, т.к. в основе алгоритма лежит преобразование произвольной системы одАНЛДУ.

Матрицу E коэффициентов левой части строим как

$$\mathrm{E}^m = \left(\begin{array}{c|c|c} E' & \multicolumn{2}{c}{B} \\ \hline \mathbb{O} & B'' & B' \end{array}\right), \tag{2.15}$$

где $E' = \mathrm{E}^{q+p}(\{1,\ldots,k_1-1\},\{k_1,\ldots,k_2-1\},\ldots,\{k_{n-1},\ldots,k_q-1\})$ или

$$E' = \begin{pmatrix} 1 & \ldots & 1 & 0 & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & & & & & 0 \\ 0 & \ldots & 0 & 1 & \ldots & 1 & 0 & \ldots\ldots\ldots\ldots & & & 0 \\ & & & & & & & & & & \\ \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots & \ldots\ldots \\ & & & & & & & & & & \\ 0 & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & & & & & & 0 & 1 & \ldots & 1 \end{pmatrix}, \tag{2.16}$$

$e_{ij} = 1$, $B \in \{0,1\}^{q\times(m-q-p)}$, $B'' \in \{0,1\}^{(n-q)\times t}$ и $B' \in \{0,1\}^{(n-q)\times(m-q-p-t)}$ — произвольные $(0,1)$-матрицы, для которых выполняется следующее условие:

$$\begin{gathered} \forall j = 1,\ldots,m-q-p-t \quad \sum_{i=1}^{q} B_{i(t+j)} + \sum_{i=1}^{n-q} B'_{ij} = 1, \\ \forall j = 1,\ldots,t \quad \sum_{i=1}^{q} B_{ij} + \sum_{i=1}^{n-q} B''_{ij} = 1. \end{gathered} \tag{2.17}$$

$\{k_1, k_2, \ldots, k_{q-1}, k_q = q+p+1\}$ произвольная последовательность возрастающих чисел: $0 < k_1 < k_2 < \cdots < k_q = q+p+1$, $e_{ij} = 1$, $0 \leqslant q \leqslant n$, $0 \leqslant p < m-q$, $0 \leqslant t < m-q-p$ — произвольные константы. При этом q — количество допустимых преобразований системы одАНЛДУ, p — количество переменных, которые дополнительно выражаются в ходе преобразований системы, t — количество переменных, которые приравниваются к 0 при приведении системы к виду содАНЛДУ.

Матрицу A коэффициентов правой части строим как

$$A = \left(\begin{array}{c|c|c} A' & \multicolumn{2}{c}{B+\Delta} \\ \hline \mathbb{O} & B''+\Delta' & B''' \end{array}\right), \tag{2.18}$$

$$A' = \begin{pmatrix} d_{11} & \ldots & d_{1(k_1-1)} & a_{1k_1} & \ldots & a_{1(k_2-1)} & a_{1k_2} & \ldots\ldots\ldots\ldots\ldots & & & a_{1(q+p)} \\ 0 & \ldots & 0 & d_{2k_1} & \ldots & d_{2(k_2-1)} & a_{2k_2} & \ldots\ldots\ldots\ldots\ldots & & & a_{2(q+p)} \\ 0 & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & & & & & & & d_{q(k_q-1)} & \ldots & d_{q(q+p)} \end{pmatrix} \tag{2.19}$$

где $a_{ij} \geqslant 0$, $a_{ij} > 0$ если $k_i \leqslant j < k_{i+1}$, $d_{ij} \in \{0,1\}$ и $\forall i\ \exists j$ т.ч. $d_{ij} = 0$. $B''' \in \{0,1\}^{(n-q)\times(m-q-p-t)}$ т.ч.

$$\forall j \quad \sum_i B'''_{ij} = 1. \tag{2.20}$$

Матрица $\Delta \in \mathbb{Z}_{+}^{q\times(m-q-p)}$ удовлетворяет следующему условию:

$$\forall\, j \quad \Delta_{qj} > 0. \tag{2.21}$$

Матрица $\Delta' \in \mathbb{Z}_{+}^{(n-q)\times(t)}$ удовлетворяет следующему условию:

$$\forall\, j \quad \sum_{i} \Delta'_{ij} > 0. \tag{2.22}$$

В матричном представлении система одАНЛДУ выглядит следующим образом

$$\left(\begin{array}{c|cc} E' & \multicolumn{2}{c}{B} \\ \hline \mathbb{O} & B'' & \multicolumn{1}{|c}{B'} \end{array}\right) x = \left(\begin{array}{c|cc} A' & \multicolumn{2}{c}{B+\Delta} \\ \hline \mathbb{O} & B''+\Delta' & \multicolumn{1}{|c}{B'''} \end{array}\right) x. \tag{2.23}$$

Теорема 2.4. *С помощью преобразования теоремы* 1.6 *система (*2.23*) может быть приведена либо к виду (*1.15*), если $q = n$; либо к виду (*1.16*), если $q < n$.*

Доказательство. 1). Пусть $q = n$. Следовательно мы получаем систему

$$(E' \mid B)x = (A' \mid B+\Delta)x \quad \Rightarrow \quad (A'-E')x' + \Delta x'' = \mathbb{O}$$

$x' = (x_1, \dots, x_n)^{\mathsf{T}}$, $x'' = (x_{n+1}, \dots, x_m)^{\mathsf{T}}$. В силу специфики построения матрицы $(A' - E')$ и условия (2.21) с помощью преобразования теоремы 1.6 система может быть преобразована к уравнению вида (1.15). Согласно теореме 1.1 находим базис Гильберта данного уравнения. Вычисляя базис Гильберта в обратном порядке получаем базис Гильберта для системы (2.1).

2). Пусть $q < n$. Следовательно получаем систему

$$\left(\begin{array}{c|cc} E' & \multicolumn{2}{c}{B} \\ \hline \mathbb{O} & B'' & \multicolumn{1}{|c}{B'} \end{array}\right) x = \left(\begin{array}{c|cc} A' & \multicolumn{2}{c}{B+\Delta} \\ \hline \mathbb{O} & B''+\Delta' & \multicolumn{1}{|c}{B'''} \end{array}\right) x \Rightarrow \left(\begin{array}{c} A'-E' \\ \hline \mathbb{O} \end{array}\right) x' + \left(\begin{array}{cc} \multicolumn{2}{c}{\Delta} \\ \hline \Delta' & \multicolumn{1}{|c}{B'''-B'} \end{array}\right) x'' = \mathbb{O}$$

С помощью преобразования теоремы 1.6 данная система приводится к системе

$$(\Delta' \mid B''' - B')\, x'' = \mathbb{O}.$$

В силу свойства матрицы Δ' решение данной системы сводится к решению системы

$$(B''' - B')\, x''' = \mathbb{O}.$$

что соответствует системе вида (1.16). Находим базис Гильберта данной системы, вычисляем базис в обратном порядке и получаем базис Гильберта для системы (2.1). $\square$

Согласно теоремы 1.6 задача нахождения базиса Гильберта для системы (2.1) сводится к задаче нахождения базиса Гильберта либо для системы вида (1.15), либо для системы вида (1.16).

Пример 2.5. **Вход:** $n = 5, \quad m = 7$.

Шаг 1: $q = 2$, $p = 0$, $t = 0$, $k_1 = 2$, $k_2 = 3$. **Шаг 2:** $B = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$, $B' = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$, $B'' = \begin{pmatrix} \\ \\ \end{pmatrix}$. **Шаг 3:** $B''' = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$.

Шаг 4: $\Delta = \begin{pmatrix} 3 & 2 & 1 & 4 & 2 \\ 2 & 3 & 5 & 4 & 1 \end{pmatrix}$. **Шаг 5:** $\Delta' = \begin{pmatrix} \\ \\ \end{pmatrix}$. **Шаг 6:** $A' = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$.

Шаг 7: $E' = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. **Шаг 8:** $\mathrm{E}^m = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}$.

Шаг 9: $A = \begin{pmatrix} 0 & 1 & 3 & 2 & 1 & 4 & 2 \\ 0 & 0 & 2 & 3 & 5 & 4 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$. **Шаг 10:** $\mathcal{H} = \begin{pmatrix} 11 & 3 & 5 \\ 8 & 1 & 2 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$.

Выводы к главе

В этой главе была исследована задача генерации систем одАНЛДУ и содАНЛДУ.

Выполнена постановка задачи генерации систем одАНЛДУ с точки зрения таких ее приложений, как тестирование, экспериментальный и сравнительный анализы реализаций алгоритмов решения систем одАНЛДУ.

На основе преобразования произвольной системы одАНЛДУ (см. теорему 1.6 из главы 1) предложены пять алгоритмов генерации систем одАНЛДУ. Первый алгоритм генерирует частный случай систем (1.16). Следующие два из них ориентированы на случай уравнения (1.15) в теореме 1.6. Это позволяет генерировать системы, базис Гильберта которых обладает заранее заданными свойствами. Четвертый алгоритм генерирует произвольные системы содАНЛДУ. Это позволяет учесть и случай системы (1.16) в теореме 1.6, что и выполняется в последнем алгоритме, который предназначен для генерации произвольной системы одАНЛДУ.

3. Решение систем одАНЛДУ

В данной главе рассматриваются алгоритмы решения систем одАНЛДУ. В п. 3.1 представлен алгоритм находжения базиса Гильберта для систем содАНЛДУ. В п. 3.2 описан алгоритм нахождения базиса Гильберта для систем одАНЛДУ.

Алгоритмы решения могут быть разбиты на несколько классов: алгоритмы проверки совместности системы, алгоритмы нахождения частного решения, алгоритмы нахождения базисного решения, алгоритмы нахождения базиса Гильберта.

В общем случае задача нахождения базиса Гильберта систем одАНЛДУ двойственна задаче генерации систем одАНЛДУ и соответствующего базиса Гильберта, т.к. в самом простом случае генерируется произвольная система одАНЛДУ и находится соответствующий базис Гильберта.

Задача нахождения базиса Гильберта для системы одАНЛДУ заключается в следующем. Пусть задана система одАНЛДУ

$$\mathrm{E}^m(I^n)x = Ax, \tag{3.1}$$

где m — количество неизвестных, n — количество уравнений. Требуется найти базис Гильберта $\mathcal{H}$ для системы (3.1).

Для случая систем содАНЛДУ система (3.1) выглядит следующим образом

$$\mathrm{E}^m(I^n)x = \mathrm{E}^m(J^n)x. \tag{3.2}$$

3.1. Алгоритм решения симметричных систем одАНЛДУ

Согласно теореме 1.4 базис Гильберта систем содАНЛДУ состоит из $\{0,1\}$-векторов. Следовательно, существует 2^m комбинаций векторов, которые могут быть базисными решениями. Формальное описание алгоритма решения представлено в Алгоритме 6.

Алгоритм решения систем содАНЛДУ заключается в поиске простых контуров в графе системы содАНЛДУ. Для поиска контура будем строить дерево обхода графа системы. При этом вершинами дерева служат текущие неизвестные, ребрами — операции присвоения неизвестным значений 0 или 1.

Алгоритм 6 Алгоритм нахождения базиса Гильберта для систем содАНЛДУ

Вход: $\mathrm{E}^m(J^n)$ — матрица коэффициентов правой части, $\mathrm{E}^m(I^n)$ — матрица коэффициентов левой части.

Выход: $\mathcal{H}$ — базис Гильберта.

1: для каждой неизвестной x_i добавляем в очередь вектор $x : \{\forall j < i \ x_j = 0,\ x_i = 1,\ \forall j > i \ x_j = n/d\}$, номер текущей неизвестной i и начальную вершину $start = i$.

2: **repeat** ▷ Обход всех ветвей дерева

3: Текущий вектор $x = \{x_k\}_{k=1}^m$, текущая неизвестная i и начальная вершина $start$. j — номер уравнения в левой части которой встречается неизвестная x_i.

4: Если в правой части уравнения j есть $x_k = 1, k \neq start$, то переход к следующему элементу очереди. Если в уравнении есть неизвестная x_{start}, то добавляем вектор x в список базисных решений и переходим к следующему элементу очереди.

5: для всех неизвестных $x_p = n/d$ из правой части уравнения j добавляем в очередь вектор x, т.ч. $x_p = 1$, а $x_w = 0$ для всех остальных x_w, присутствующих в правой части уравнения j, текущую неизвестную p и начальную вершину $start$.

6: **until** очередь не пуста.

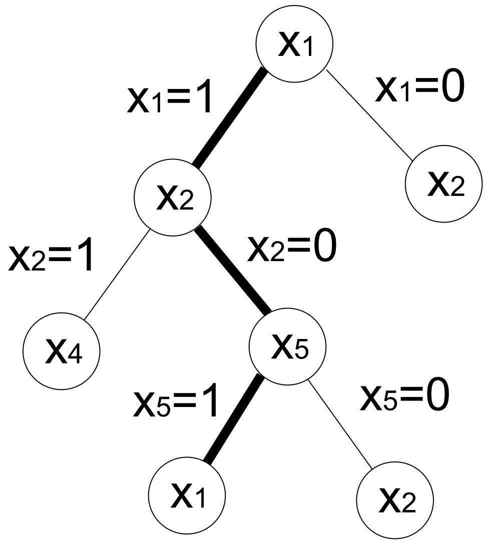


Рис. 5. Пример фрагмента дерева поиска с найденым простым контуром графа системы содАНЛДУ

Для обхода дерева воспользуемся методом поиска в ширину с использованием очереди для хранения текущих состояний. Каждое состояние характеризуется вектором значений неизвестных x, текущей неизвестной i и начальной неизвестной $start$, которая обозначает первую ненулевую неизвестную в текущей цепи. Неизвестные, которые еще не были пройдены помечаются символом n/d.

Базисным решением содАНЛДУ является путь в дереве обхода, который начинается и заканчивается в вершинах с одинаковой переменной (см. рис. 5).

В случае, если появляются контуры не содержащие все неизвестные цепи с ненулевыми значениями, то данная ветвь дерева не содержит новых решений.

Пример 3.1.

Вход: $\mathrm{E}^7(I^3) = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$, $\mathrm{E}^7(J^3) = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}$.

Шаг 1. Добавляем в очередь: $(x = \{1, n/d, n/d, n/d, n/d, n/d, n/d\}$, $i = 1$, $start = 1)$, $(x = \{0, 1, n/d, n/d, n/d, n/d, n/d\}$, $i = 2$, $start = 2)$, $(x = \{0, 0, 1, n/d, n/d, n/d, n/d\}$, $i = 3$, $start = 3)$, $(x = \{0, 0, 0, 1, n/d, n/d, n/d\}$, $i = 4$, $start = 4)$, $(x = \{0, 0, 0, 0, 1, n/d, n/d\}$, $i = 5$, $start = 5)$, $(x = \{0, 0, 0, 0, 0, 1, n/d\}$, $i = 6$, $start = 6)$, $(x = \{0, 0, 0, 0, 0, 0, 1\}$, $i = 7$, $start = 7)$. **Шаг 3.** Из очереди $(x = \{1, n/d, n/d, n/d, n/d, n/d, n/d\}$, $i = 1$, $start = 1)$, $j = 2$. **Шаг 5.** В очередь $(x = \{1, n/d, n/d, n/d, 1, n/d, 0\}$, $p = 5$, $start = 1)$, $(x = \{1, n/d, n/d, n/d, 0, n/d, 1\}$, $p = 7$, $start = 1)$. **Шаг 3.** Из очереди ($x = \{0, 1, n/d, n/d, n/d, n/d, n/d\}$, $i = 2$, $start = 2$), $j = 3$. **Шаг 5.** В очередь ($x = \{0, 1, 1, n/d, n/d, 0, n/d\}$, $p = 3$, $start = 2$), ($x = \{0, 1, 0, n/d, n/d, 1, n/d\}$, $p = 6$, $start = 2$). **Шаг 3.** Из очереди ($x = \{0, 0, 1, n/d, n/d, n/d, n/d\}$, $i = 3$, $start = 3$), $j = 3$. **Шаг 4.** В правой части есть x_3. Базисное решение $x = \{0, 0, 1, 0, 0, 0, 0\}$. **Шаг 3.** Из очереди ($x = \{0, 0, 0, 1, n/d, n/d, n/d\}$, $i = 4$, $start = 4$), $j = 1$. **Шаг 4.** В правой части есть x_4. Базисное решение $x = \{0, 0, 0, 1, 0, 0, 0\}$. **Шаг 3.** Из очереди ($x = \{0, 0, 0, 0, 1, n/d, n/d\}$, $i = 5$, $start = 5$), $j = 1$. **Шаг 3.** Из очереди ($x = \{0, 0, 0, 0, 0, 1, n/d\}$, $i = 6$, $start = 6$), $j = 2$. **Шаг 5.** В очередь ($x = \{0, 0, 0, 0, 0, 1, 1\}$, $p = 7$, $start = 6$). **Шаг 3.** Из очереди ($x = \{0, 0, 0, 0, 0, 0, 1\}$, $i = 7$, $start = 7$), $j = 1$. **Шаг 3.** Из очереди ($x = \{1, n/d, n/d, n/d, 1, n/d, 0\}$, $i = 5$, $start = 1$), $j = 1$. **Шаг 4.** В правой части есть x_1. Базисное решение $x = \{1, 0, 0, 0, 1, 0, 0\}$. **Шаг 3.** Из очереди ($x = \{1, n/d, n/d, n/d, 0, n/d, 1\}$, $i = 7$, $start = 1$), $j = 1$. **Шаг 4.** В правой части есть x_1. Базисное решение $x = \{1, 0, 0, 0, 0, 0, 1\}$. **Шаг 3.** Из очереди ($x = \{0, 1, 1, n/d, n/d, 0, n/d\}$, $i = 3$, $start = 2$), $j = 3$. **Шаг 4.** В правой части есть $x_3 = 1$. Пропускаем. **Шаг 3.** Из очереди ($x = \{0, 1, 0, n/d, n/d, 1, n/d\}$, $i = 6$, $start = 2$), $j = 2$. **Шаг 5.** В очередь ($x = \{0, 1, 0, n/d, 1, 1, 0\}$, $p = 5$, $start = 2$), ($x = \{0, 1, 0, n/d, 0, 1, 1\}$, $p = 7$, $start = 2$). **Шаг 3.** Из очереди ($x = \{0, 0, 0, 0, 0, 1, 1\}$, $p = 7$, $start = 6$), $j = 1$. **Шаг 3.** Из очереди ($x = \{0, 1, 0, n/d, 1, 1, 0\}$, $p = 5$, $start = 2$), $j = 1$. **Шаг 4.** В правой части есть x_2. Базисное решение $x = \{0, 1, 0, 0, 1, 1, 0\}$. **Шаг 3.** Из очереди ($x = \{0, 1, 0, n/d, 0, 1, 1\}$, $p = 7$, $start = 2$), $j = 1$. **Шаг 4.** В правой части есть x_2. Базисное решение $x = \{0, 1, 0, 0, 0, 1, 1\}$.

3.2. Алгоритм решения произвольной системы одАНЛДУ

Алгоритм решения систем одАНЛДУ основан на преобразовании теоремы 1.6. Формальное описание алгоритма представлено в Алгоритме 7.

Рассмотрим произвольную систему одАНЛДУ

$$\mathrm{E}^m x = Ax, \tag{3.3}$$

где $\mathrm{E}^m = \mathrm{E}^m(I^n) \in \{0,1\}^{n\times m}$, $A \in \mathbb{Z}_+^{n\times m}$. Требуется найти базис Гильберта $\mathcal{H} \in \mathbb{Z}_+^{m\times q}$, где q — количество элементов базиса Гильберта. Систему (3.3) можно также записать в виде $\widetilde{A}x = \mathbb{O}$, где $\widetilde{A}^{(0)} = A - \mathrm{E}^m(I^n)$. Обозначим систему (3.3) в виде $S^{(0)}(I^n, A)$, соответствующий базис Гильберта — $\mathcal{H}^{(0)}$. Будем выполнять преобразование системы $S^{(0)}$ пошагово

$$S^{(0)} \to S^{(1)} \to \cdots \to S^{(r)}$$

Алгоритм 7 Алгоритм нахождения базиса Гильберта для систем одАНЛДУ

Вход: A — матрица коэффициентов правой части, $\mathrm{E}^m(I^n)$ — матрица коэффициентов левой части.

Выход: $\mathcal{H}$ — базис Гильберта.

1: $k = 0$, $n_0 = n$, $\widetilde{A}^{(0)} = A - \mathrm{E}^m(I^n)$;
2: **repeat**
3: вычисление коэффициентов $c_j = \sum\limits_{i=1}^{n_k} \widetilde{a}_{ij}$;
4: если $\forall c_j > 0$, то $\mathcal{H}^k = \emptyset$, переход на шаг 12;
5: если $\forall c_j \geqslant 0$, то $\forall c_p > 0 \Rightarrow x_p = 0$, решаем систему содАНЛДУ, переход на шаг 12;
6: составляем множество K по формуле (1.13);
7: строим уравнение (1.14);
8: строим $\widetilde{A}^{(k+1)} = \widetilde{A}^{(k)} \backslash$ {уравнение n_k и неизвестные x_k, $k \in K$};
9: k=k+1;
10: **until** $n_k > 1$
11: решаем оставшееся уравнение вида (1.15);
12: **repeat**
13: k=k-1;
14: $\forall h \in \mathcal{H}^{(k+1)}$ решаем уравнение (1.14);
15: **until** $k > 0$

до тех пор, пока не получим базис Гильберта $\mathcal{H}^{(r)}$.

Пусть $S^{(l)}$ — текущая система одАНЛДУ с басисом Гильберта $\mathcal{H}^{(l)}$. В соответствии с преобразованием теоремы 1.6 сложим все уравнения системы

$$\sum_{j=1}^{m_l} c_j x_j = 0, \qquad \text{где } c_j = \sum_i \widetilde{A}^{(l)}_{ij}. \tag{3.4}$$

Если все коэффициенты $c_j > 0$, то базис Гильберта $\mathcal{H}^{(l)} = \emptyset$ и преобразование завершается.

Если все коэффициенты $c_j \geqslant 0$ и существует хотя бы один коэффициент $c_j = 0$, то для всех коэффициентов $c_t > 0$ полагаем $x_t = 0$, и приводим систему к виду (1.16). Полученная система является системой содАНЛДУ. Алгоритм нахождения базиса Гильберта для системы содАНЛДУ описан в п. 3.1.

Если существует коэффициент $c_j = -1$, то для всех $c_t = 0$ в базис Гильберта $\mathcal{H}^{(l)}$ добавляем соответствующий единичный вектор e_t. Далее проводим преобразование системы $S^{(l)}$ в систему $S^{(l+1)}$ с соответствующим базисом Гильберта $\mathcal{H}^{(l+1)}$. В случае, когда система $S^{(l)}$ состоит из одного уравнения, решаем это уравнение и находим базис Гильберта $\mathcal{H}^{(l)}$.

Обратное преобразование для получения базиса Гильберта $\mathcal{H}^{(0)}$ выполняется следующим образом. Пусть задан базис Гильберта $\mathcal{H}^{(l+1)}$. Каждый элемент этого базиса подставляем в уравнение (1.14) и находим все решения данного уравнения. Полученные решения комбинируем с текущим элементом базиса Гильберта $\mathcal{H}^{(l+1)}$ и получаем элементы базиса Гильберта $\mathcal{H}^{(l)}$.

Пример 3.2. **Вход:** $A = \begin{pmatrix} 0 & 0 & 0 & 3 & 1 \\ 5 & 0 & 7 & 2 & 3 \\ 0 & 0 & 0 & 0 & 3 \end{pmatrix}$, $\mathrm{E}^5(I^3) = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$. **Шаг 1.** $k = 0$, $n_0 = 3$, $\widetilde{A}^{(0)} = \begin{pmatrix} -1 & 0 & -1 & 3 & 1 \\ 5 & -1 & 7 & 2 & 3 \\ 0 & 0 & 0 & -1 & 2 \end{pmatrix}$. **Шаг 3.** $c_1 = 4$, $c_2 = -1$, $c_3 = 6$, $c_4 = 4$, $c_5 = 6$.
Шаг 6. $K = \{2\}$. **Шаг 7.** $x_2 = 5x_1 + 7x_3 + 2x_4 + 3x_5 = T_1(x_1, x_3, x_4, x_5)$.
Шаг 8. $\widetilde{A}^{(1)} = \begin{pmatrix} -1 & -1 & 3 & 1 \\ 0 & 0 & -1 & 2 \end{pmatrix}$. **Шаг 9.** $k = 1$. **Шаг 10.** $n_1 = 2 \Rightarrow$ переход на Шаг 3. **Шаг 3.** $c_1 = -1$, $c_3 = -1$, $c_4 = 2$, $c_5 = 3$. **Шаг 6.** $K = \{1, 3\}$.
Шаг 7. $x_1 + x_3 = 3x_4 + x_5 = T_2(x_4, x_5)$. **Шаг 8.** $\widetilde{A}^{(2)} = \begin{pmatrix} -1 & 2 \end{pmatrix}$. **Шаг 9.** $k = 2$.
Шаг 10. $n_2 = 1 \Rightarrow$ переход на Шаг 11. **Шаг 11.** $\mathcal{H}^{(2)} = \left\{ \begin{pmatrix} 2 \\ 1 \end{pmatrix} \right\}$. **Шаг 13.** $k = 1$.

Шаг 14. $\mathcal{H}^{(1)} = \left\{ \begin{pmatrix} 7 \\ 0 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 6 \\ 1 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 5 \\ 2 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 4 \\ 3 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ 4 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 2 \\ 5 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 6 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 7 \\ 2 \\ 1 \end{pmatrix} \right\}$.
Шаг 15. $k > 0 \Rightarrow$ переход на Шаг 13. **Шаг 13.** $k = 0$.

Шаг 14. $\mathcal{H}^{(0)} = \left\{ \begin{pmatrix} 7 \\ 42 \\ 0 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 6 \\ 44 \\ 1 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 5 \\ 46 \\ 2 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 4 \\ 48 \\ 3 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ 50 \\ 4 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 2 \\ 52 \\ 5 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 54 \\ 6 \\ 2 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 56 \\ 7 \\ 2 \\ 1 \end{pmatrix} \right\}$.
Шаг 15. $k = 0 \Rightarrow$ конец.

Пример 3.3. **Вход:** $A = \begin{pmatrix} 0 & 3 & 1 & 5 & 0 \\ 0 & 1 & 1 & 2 & 3 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 2 & 3 & 0 \end{pmatrix}$, $\mathrm{E}^5(I^4) = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$. **Шаг 1.** $k = 0$, $n_0 = 4$, $\widetilde{A}^{(0)} = \begin{pmatrix} 0 & 3 & 1 & 5 & -1 \\ -1 & 1 & 1 & 2 & 3 \\ 0 & 0 & 2 & 1 & 0 \\ 0 & 0 & 1 & 2 & 0 \end{pmatrix}$. **Шаг 3.** $c_1 = -1$, $c_2 = 4$, $c_3 = 5$, $c_4 = 10$, $c_5 = 2$. **Шаг 6.** $K = \{1\}$. **Шаг 7.** $x_1 = x_2 + x_3 + 2x_4 + 3x_5 = T_1(x_2, x_3, x_4, x_5)$. **Шаг 8.** $\widetilde{A}^{(1)} = \begin{pmatrix} 0 & 1 & 5 & -1 \\ 0 & 2 & 1 & 0 \\ 0 & 1 & 2 & 0 \end{pmatrix}$. **Шаг 9.** $k = 1$. **Шаг 10.** $n_1 = 3 \Rightarrow$ переход на Шаг 3. **Шаг 3.** $c_2 = 0$, $c_3 = 4$, $c_4 = 8$, $c_5 = -1$. **Шаг 6.** $K = \{5\}$. **Шаг 7.** $x_5 = 3x_2 + x_3 + 5x_4 = T_2(x_2, x_3, x_4)$. **Шаг 8.** $\widetilde{A}^{(2)} = \begin{pmatrix} 0 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix}$. **Шаг 9.** $k = 2$. **Шаг 10.** $n_2 = 2 \Rightarrow$ переход на Шаг 3. **Шаг 3.** $c_2 = 0$, $c_3 = 3$, $c_4 = 3$. **Шаг 5.** $c_2 = 0 \Rightarrow x_3 = 0$, $x_4 = 0$, система содАНЛДУ: $x_2 = x_2$. Решение $\mathcal{H}^{(2)} = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \right\}$. Переход на Шаг 12. **Шаг 13.** $k = 1$.

Шаг 14. $\mathcal{H}^{(1)} = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 3 \end{pmatrix} \right\}$. **Шаг 15.** $k > 0 \Rightarrow$ переход на Шаг 13. **Шаг 13.** $k = 0$.

Шаг 14. $\mathcal{H}^{(0)} = \left\{ \begin{pmatrix} 10 \\ 1 \\ 0 \\ 0 \\ 3 \end{pmatrix} \right\}$. **Шаг 15.** $k = 0 \Rightarrow$ конец.

Выводы к главе

В этой главе была рассмотрена задача решения систем одАНЛДУ и содАНЛДУ. Задача решения является двойственной задаче генерации систем уравнений.

1. Предложен алгоритм нахождения базиса Гильберта для систем содАНЛДУ. Алгоритм основан на обходе графа системы содАНЛДУ и поиске простых циклов.

2. Предложен алгоритм нахождения базиса Гильберта для систем одАНЛДУ. Алгоритм основан на преобразовании системы одАНЛДУ (см. теорему 1.6 из главы 1).

4. Математическое обеспечение

В данной главе представлено разработанное на основе предложенных алгоритмов программное обеспечение для выполнения комплексного тестирования, экспериментального и сравнительного анализа реализаций алгоритмов решения систем одАНЛДУ. Кроме реализации собственно алгоритмов, это потребовало развития способов измерения потребляемых ресурсов, на основе технологий системного и межплатформенного программирования. Полученное ПО было использовано для решения основной из поставленных прикладных задач данного исследования — экспериментальное обоснование корректности и эффективности реализации синтаксического алгоритма syntactic solver решения систем одАНЛДУ и разработка на его основе системы Web-SynDic научно-образовательного и вычислительного сервиса. Основные результаты данной главы опубликованы в [8, 14].

В п. 4.1 ставится задача тестирования и экспериментального анализа алгоритмов решения систем одАНЛДУ. В п. 4.2 описаны практически приемлемые способы измерения вычислительных ресурсов алгоритмов решения. В п. 4.3 представлена программная система alg_analyser для выполнения тестирования, экспериментального анализа и сравнения алгоритмов решения систем одАНЛДУ. В п. 4.4 представлена система Web-SynDic, предназначенная для удаленной демонстрации эффективности и тестирования алгоритмов решения систем одАНЛДУ.

4.1. Задача тестирования и экспериментального анализа алгоритмов решения систем одАНЛДУ

4.1.1. Постановка задачи тестирования

Обоснование корректности программного обеспечения составляет важный этап технологии производства ПО [27]. Комплексное системное тестирование реализации алгоритма решения системы одАНЛДУ означает запуск реализации на большом числе тестовых систем, генерируемых автоматически. Причем, для обеспечения покрывающего тестирования множество тестовых систем разбивается на частные классы. Для каждого теста требуется проверка результата, полученного в процессе решения. Данная проверка может быть обеспечена путем сравнения полученного результата с результатом альтернативного решателя или с заранее известным результатом. Последний вариант предполагает генерацию тестовых систем одновременно с генерацией соответствующего эталонного решения.

Разработанные в главе 2 алгоритмы генерации систем одАНЛДУ выгодно использовать для решения такой задачи тестирования. Алгоритмы 1–4 позволяют генерировать тестовые системы для различных частных классов. Это разбиение управляется как собственно алгоритмом (каждый алгоритм генерирует свой класс систем, см. рис. 4 на с. 33), так и входыми параметрами (n, m, q и $||A||$). В случае аттестационного тестирования полезным оказывается Алгоритм 5, который позволяет выполнять тестирование на всем классе систем одАНЛДУ.

4.1.2. Постановка задачи экспериментального анализа

Экспериментальный анализ вычислительной сложности предназначен для исследования эффективности алгоритма решения. При этом оценивается объем потребляемых вычислительных ресурсов: время работы программы и используемый объем памяти. Такая оценка может быть использована для решения следующих задач.

1. Построение экспериментальной зависимости объема потребляемых ресурсов от размерности входных данных. В случае систем одАНЛДУ размерности входных данных могут определяться числом уравнений n, числом неизвестных m, абсолютными величинами коэффициентов $||A||$ и числом элементов q в искомом базисе Гильберта.

2. Построение карты эффективности исследуемого алгоритма. В случае систем одАНЛДУ множество всех возможных систем разбивается на классы со сходной эффективностью решения данным алгоритмом. Тем самым получаем детальное описание эффективности алгоритма, выявляя наихудшие и наилучшие случаи его работы.

3. Сравнительный анализ эффективности заданного алгоритма и его альтернатив. В рамках данной задачи может быть определено, на каких классах систем одАНЛДУ алгоритм решения превосходит или уступает по эффективности альтернативные решатели, а на каких классах он может быть сопоставим по эффективности.

4.2. Способы измерения вычислительных ресурсов

Далее описаны предлагаемые способы для оценивания объема потребляемых вычислительных ресурсов реализациями алгоритмов решения систем одАНЛДУ.

4.2.1. Оценивание затрачиваемого времени

Под затрачиваемым временем мы будем понимать *время работы приложения* — разница между временем окончания работы и временем запуска приложения, и *время работы процессора* — количество мигов (jiffies) которое было затрачено процессором на работу данного приложения. Время работы процессора является достаточно точным и независимым от внешних воздействий показателем, однако он не включает в себя такие моменты как время ожидания чтения данных с носителя и т.д. Эти моменты могут играть большую роль при решении трудоемких задач, поэтому целесообразно также анализировать и время работы приложения — величину менее устойчивой к внешним воздействиям, таким как общая загрузка системы и т.д.

1. Время работы приложения. Получение времени как разности между временем окончания и временем начала работы процесса. Такой метод дает хорошую оценку работы процесса, особенно если общее время работы мало (ср. со способом 3). В то же время, отметим следующие его недостатки:

— на значение полученной оценки могут сильно влиять параллельно работающие процессы. Они занимают часть ресурсов компьютера, увеличивая, тем самым, общее время работы исследуемой реализации алгоритма решения;

— полученная оценка общего времени работы включает в себя не только собственно время решения, но и время, требуемое для обслуживания процесса на уровне операционной системы (запуск/завершение процесса, открытие/закрытие файлов и т. п.).

2. Типичное время работы приложения. Получение времени как некоторой статистической оценки типичного значения разности между временем окончания и временем начала работы процесса. Такой оценкой, например, может выступать среднее или медиана. Для этого процесс запускается многократно для одной и той же системы одАНЛДУ. Данный способ позволяет частично избежать первого из недостатков предыдущего способа. Однако, второй недостаток остается неустраненным.

3. Время работы процессора. Получение времени по системным данным работы процесса. Данная информация находится в директории `\proc\<pid>`, где `<pid>` — идентификационный номер процесса[1]. Этот способ обеспечивает получение оценки с точностью

[1]Это верно для ОС семейства Unix. В других ОС эти данные можно получить либо через системные функции, либо с помощью эмуляторов ОС семейства Unix.

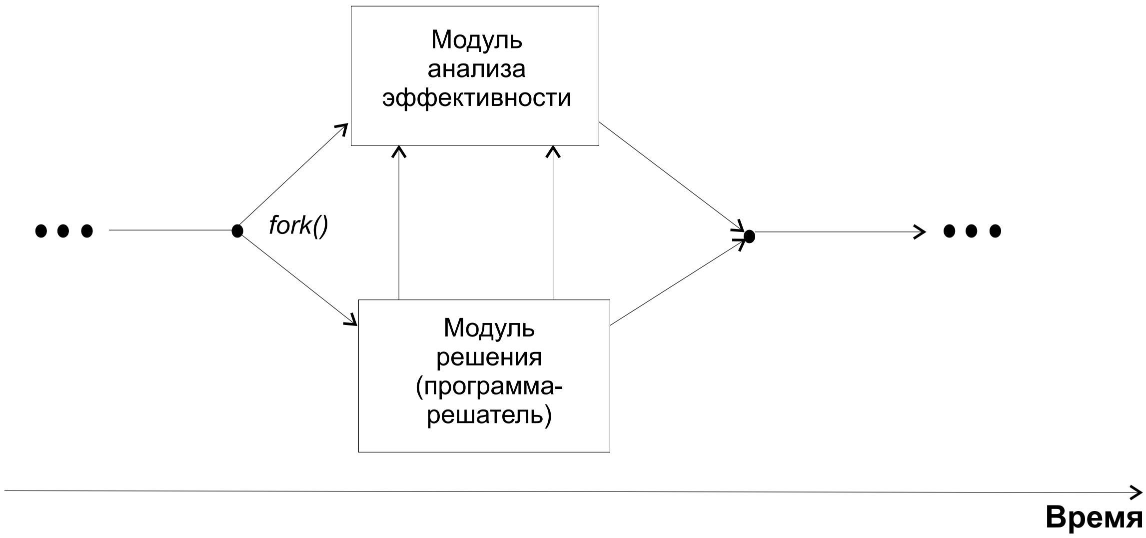


Рис. 6. Схема получения информации об используемой памяти

от одного мига (от 0.01 до 0.001 секунды в зависимости от версии ядра ОС), при этом не учитываются время чтения/записи, запуска и завершения работы программы-решателя, время ожидания нового кванта времени и т.д. Недостатком является то, что при малых вычислительных затратах время работы программы-решателя может быть меньше одного мига. В этом случае невозможно корректно оценить вычислительные затраты и следует использовать способы 1 и 2.

В дальнейшем мы будем использовать способ 1 для получения времени работы приложения и способ 3 для получения системного времени работы процессора, т.к. такой подход позволяет быстро получить оценки затрачиваемого времени. Интеграцию этих способов со способом 2 предполается выполнить в последующих исследованиях.

4.2.2. Оценивание используемого объема памяти

Объем занимаемой памяти в данный момент при работе программы можно определить по системным данным, которые находятся в директории `\proc\<pid>`, где `<pid>` — идентификационный номер процесса. Для получения более детальной и точной информации о работе программы требуется регулярный опрос системных данных. Этот опрос можно организовать с помощью специального процесса, работающего параллельно с программой решения системы.

Схема работы представлена на рис 6. В главном модуле посредством системного вызова *fork()* создается процесс-потомок и возвращается в родительский процесс идентификационный номер. Затем процесс-потомок заменяется на программу-решатель системы одАНЛДУ (системный вызов *execl()*), а родительский процесс осуществляет мониторинг и сбор информации о потреблении ресурсов процессом-потомком. Сбор информации останавливается при завершении работы программы-решателя.

Такой способ позволяет отслеживать изменения объема используемой памяти. В тоже время, недостатком данного способа является то, что часть ресурсов используется для постоянного мониторинга и это замедляет работу программы-решателя.

Для уменьшения количества ресурсов, затрачиваемых на мониторинг процесса можно воспользоваться дифференцированным методом опроса. В случае, если программа-решатель работает достаточно долго (несколько секунд, минут и т.д.) и объем используемой памяти меняется медленно, то достаточно лишь небольшое число раз проверять объем используемой памяти и получить оценку. В тоже время, если процесс решения занимает доли секунд, то требуется достаточно частый опрос для получения корректной оценки.

Для расчета времени ожидания между опросами наиболее эффективен экспоненциальный рост показателя. При этом требуется ограничение сверху на время ожидания для того, чтобы небыло слишком больших периодов ожидания.

Например, в системе alg_analyser реализован следующий алгоритм:

— начальное значение времени ожидания равно 0.5 мига;

— максимальное время ожидания - 1 секунда;

— в начале работы алгоритма время ожидания не увеличивается в течение 3-х раз;

— если память не меняется, то время ожидания увеличивается в 2 раза;

— если изменилась память, то время ожидания устанавливается в исходное значение в 0.5 мига и не увеличивается в течение последующих 3-х раз.

4.3. Программная система alg_analyser

Программная система alg_analyser предназначена для выполнения тестирования, экспериментального анализа и сравнения реализаций алгоритмов решения систем одАНЛДУ.

В силу специфики решаемых задач разработанное программное обеспечение (ПО) должно отвечать следующим ключевым требованиям:

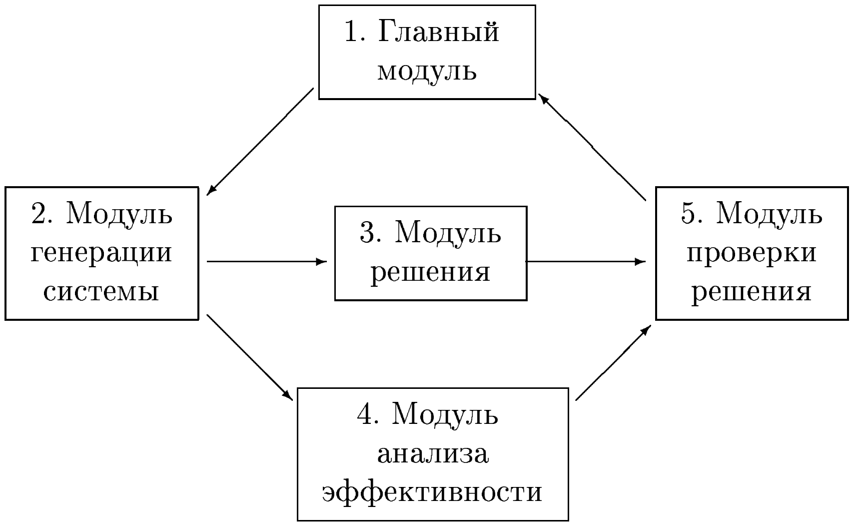


Рис. 7. Архитектура системы alg_analyser и основные информационные потоки

1. Модульность. Это необходимо для гибкого подключения новых алгоритмов генерации тестовых систем, исследуемых программ-решателей, используемых методов оценивания потребления ресурсов.

2. Автономность. Возможность работы ПО в автономном режиме на выделенном сервере в течении длительного времени. Это вызвано большим объемом тестов, необходимых для проведения комплексного и детального анализа.

3. Независимость от формата исследуемого алгоритма. Алгоритм может быть доступен без исходного кода, а лишь как непосредственно исполняемая программа. В любом случае разработанное ПО должно осуществлять измерение потребляемых ресурсов.

4.3.1. Модульная структура ПО

Общая схема работы разработанного ПО представлена на рис. 7. В системе alg_analyser можно выделить следующие модули.

Главный модуль. Главный модуль координирует работу всех остальных модулей. В его задачу входит выбор и последовательный запуск этих модулей, распределение и передача информации модулям. При возникновении критической ситуации (ошибки) он выбирает один из двух вариантов действий:

– зафиксировать возникновение фатальной ошибки путем записи в log-файл и вывода на экран соответствующих сообщений, а затем аварийно завершить работу;

– вывести сообщение о возникновении критической ситуации, записать вызвавшие ошибку данные на диск и продолжить работу (восстановление после ошибки).

При этом приоритет отдается второму направлению действия, т. к. в этом случае впоследствии можно установить факт существования ошибки и вызвавшие ее причины.

Кроме этого главный модуль ведет сбор и вывод первоначальной статистической информации о текущем тесте (размерности системы одАНЛДУ, характеристики решения и результатов тестирования), подсчет пройденных тестов и задает исходные параметры для модуля генерации системы одАНЛДУ.

Модуль генерации системы одАНЛДУ. Модуль генерации по заданным исходным параметрам генерирует систему одАНЛДУ и соответствующий ей базис Гильберта. В текущей версии в состав модуля входят три алгоритма генерации (см. пп. 2.2 - 2.4). Исходными параметрами являются: 1) номер алгоритма, 2) число уравнений n, 3) число неизвестных m. Остальные данные задаются через файл конфигурации программы. Результатом работы модуля является система одАНЛДУ и базис Гильберта к ней.

Модуль решения. Модуль решения запускает программу-решатель для решения системы одАНЛДУ. Так как программа является внешней, то она запускается в виде отдельного процесса и работа с ней идет на уровне входных-выходных файлов. Модуль решения накладывает ограничения на процесс решения. Поэтому, работа программы-решателя может быть завершена без получения решения, например, из-за нехватки памяти, по истечении отведенного на решение времени, из-за численного переполнения при вычислениях и т. п.

Модуль проверки решения. Модуль проверки решения проверяет, удалось ли исследуемой программе-решателю получить решение и сравнивает его с эталонным. Последнее получается одновременно с генерацией тестовой системы. В системе alg_analyser выделяется 3 варианта результатов сравнения: решения совпадают, решения не совпадают и программа-решатель не смогла найти решение. Эти варианты достаточны для проведения тестирования и сравнительного анализа.

Модуль анализа эффективности. Данный модуль запускается одновременно с запуском модуля решения и отслеживает эффективность работы программы-решателя: время и занимаемая память. По завершении работы модуля решения вся накопленная информация передается главному модулю.

На рис. 7 показана последовательность работы модулей ПО, при этом, как было сказано выше, в случае возникновения критической ситуации управление передается главному модулю, независимо от того, какой модуль в данный момент работает, и, в случае успешной обработки критической ситуации, последовательность работы начинается с главного модуля.

4.3.2. Сводные характеристики реализации

В таблице 1 представлены сводные характеристики модулей реализаций ПО. Для реализации используются языки программирования C++ и Perl в среде ОС Linux и Windows. Последний вариант реализован на основе технологии Cygwin [19]. Объем каждого модуля указан в строках кода без учета комментариев.

Таблица 1. Характеристики реализованного программного обеспечения

Название модуля	Язык программирования	Количество строк
Главный модуль	C++	739
Модуль генерации:		
Алгоритм 1	C++	55
Алгоритм 2	C++	171
Алгоритм 3	C++	321
Модуль анализа эффективности	C++	145
Модуль проверки решения	Perl	63
ИТОГО:	C++/Perl/Всего	1431/63/1494

4.3.3. Эксперименты

Для экспериментального исследования был предоставлен синтаксический алгоритм нахождения базиса Гильберта систем одАНЛДУ (syntactic solver), разработанный и реализованный Д.Ж.Корзуном [5, 6]. В качестве альтернативного алгоритма был выбран алгоритм нахождения базиса Гильберта систем одНЛДУ (slopes), разработанный и реализованный A. P. Tomás и M. Filgueiras [28].

Экспериментальная часть исследований состояла из двух этапов:

1) проверка качества реализации алгоритма на наличие ошибок,

2) проверка эффективности алгоритма syntactic solver и сравнение его с альтернативным алгоритмом slopes.

На первом этапе было произведено массовое тестирование алгоритма на различных классах тестовых систем для выявления возможных нерешаемых систем. Результаты тестирования представлены в таблице 2.

Под размером матрицы и размером переменных понимается ограничение на максимум из количества столбцов и строк и максимум из коэффициентов матрицы A. Короткие серии экспериментов связаны с обнаруженными в ходе тестирования недочетами в реализации alg_analyser.

Таблица 2. Журнал проведенных тестов

Номер алгоритма	Дата запуска	Дата остановки	Кол-во тестов	Пропущено	Ошибочно тестов	Размер матрицы	Размер переменных
1	26.02.02	28.02.02	410786	0	0	100	100000
1	28.02.02	28.02.02	3	0	0	500	100000
2	28.02.02	07.03.02	204308	0	0	1000	100000
1	28.02.02	01.03.02	10	0	0	300	500
1	01.03.02	07.03.02	2837	0	0	1000	10000
1	18.03.02	08.07.02	88640	0	0	1000	10000
1	18.03.02	08.07.02	937234	0	0	1000	10000
Итого	—	—	1643818	0	0	—	—

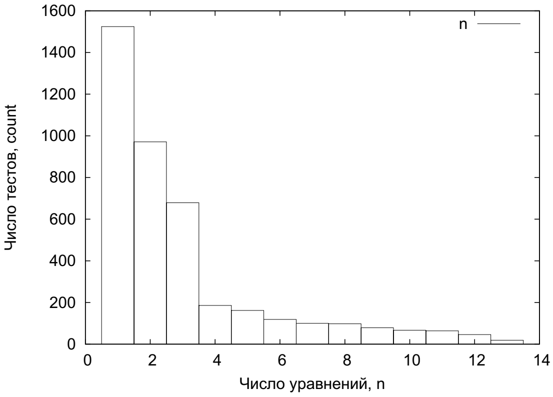
Было сгенерировано и решено более 1.5 миллиона систем. Ошибок в реализации алгоритма syntactic solver выявлено не было.

На втором этапе была произведена генерация 10 тысяч различных тестовых систем и сравнение на их основе алгоритмов решения syntactic solver и slopes. Так как португальский алгоритм накладывал существенные ограничения на размерность решаемых систем одНЛДУ, то генерация и проверка была произведена на максимально возможной размерности ($m \leqslant 20$, $||A|| \leqslant 100$).

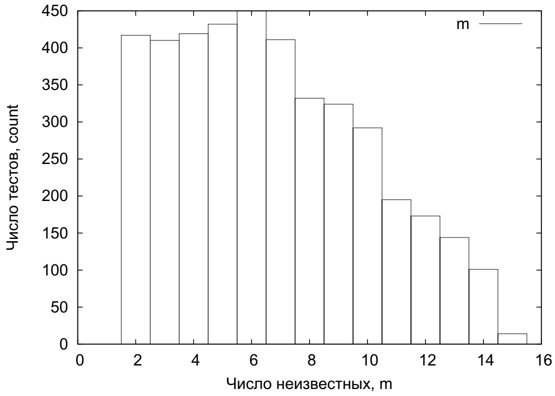
Несмотря на то, что возможности разработанного ПО позволяют измерять используемый объем памяти, для систем столь малых размерностей подобное измерение становилось малоинформативным — размерности систем отличаются друг от друга незначительно и средства операционной системы не позволяют их дифференцировать. В силу этого, в сравнительном анализе мы использовали только измерение затраченного на решение времени.

Выборка составлена из серий экспериментов. Разбиение на серии было вызвано необходимостью использования различных алгоритмов генерации и причинами технического характера, вызывавшими прерывание генерации. Часть сгенерированных систем, которые не смогли быть решены slopes в течении 100 секунд, были удалены из выборки (≈ 500 систем).

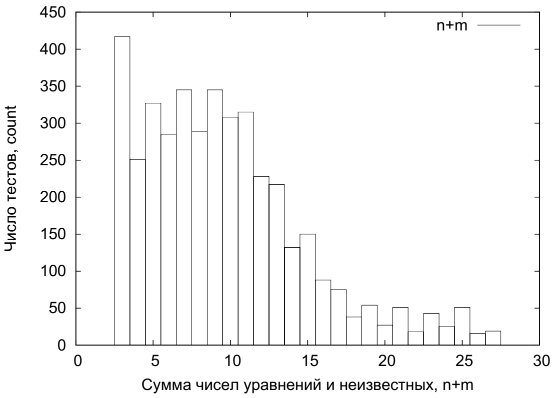
Описание общих характеристик полученной выборки представлены на рис. 8(a),



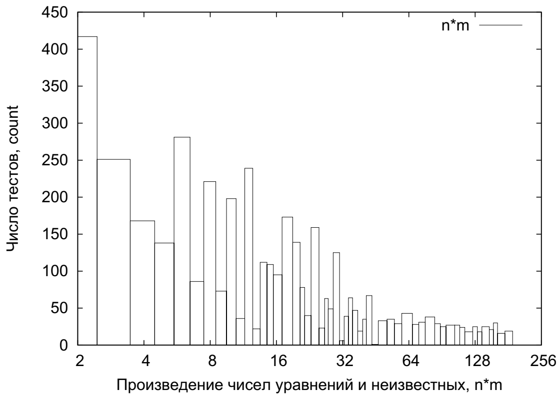
(a) по числу уравнений n



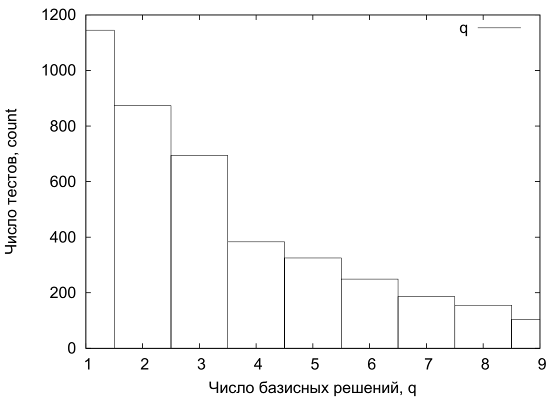
(b) по числу неизвестных m



(c) по сумме чисел уравнений и неизвестных



(d) по произведению чисел уравнений и неизвестных

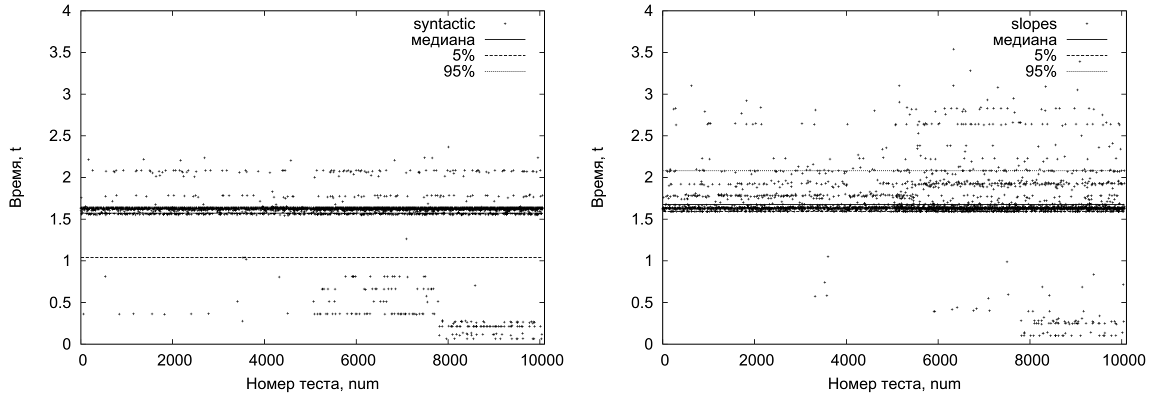


(e) по числу векторов базиса Гильберта

Рис. 8. Распределение систем одАНЛДУ

8(b), 8(e), 8(c) и 8(d).

На рис. 8(a) показано распределение выборки по числу уравнений n. С ростом n количество соответствующих систем убывает. Это вызвано тем, что n ограничено сверху m, которое выбирается первым (равномерно из $\mathbb{N}_{\bar{m}}$), затем определяется значение n



(a) Алгоритм syntactic (b) Алгоритм slopes

Рис. 9. Распределение времени решения по номерам сгенерированных тестовых систем одАНЛДУ

(равномерно из $\mathbb{N}_m$).

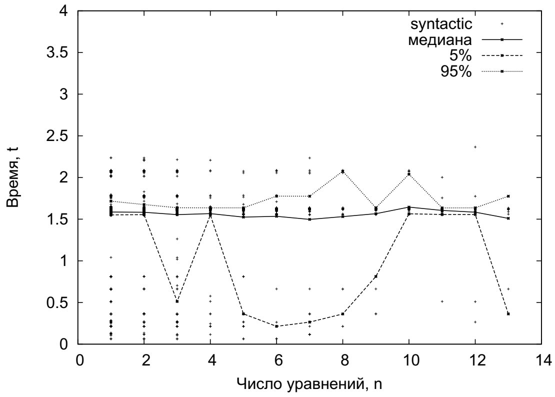
Поведение числа генерируемых систем в зависимости от m (см. рис. 8(b)) объясняется выбором верхней границы $1 \leqslant \bar{m} \leqslant 20$ для m в каждой серии экспериментов. Кроме того, убывание числа систем с ростом m вызывается и тем, что нерешенные системы (алгоритм slopes) исключались из выборки.

На рис. 8(e) показано распределение систем по числу векторов базиса Гильберта. Такое распределение определяется используемыми алгоритмами генерации, для которых характерно построение систем с небольшим числом базисных решений. В настоящих экспериментах нами рассматривался случай с ограничением на размер базиса Гильберта $q \leqslant 9$.

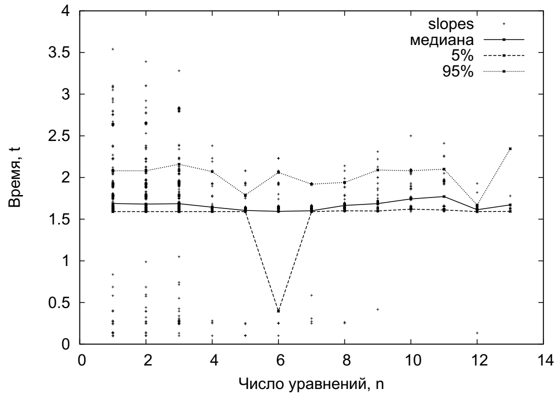
Другие особенности сгенерированной выборки представлены на рис.8(c), где показано распределение систем по сумме чисел уравнений и переменных в системе, а также и на рис. 8(d), где в логарифмической шкале показано распределение систем по произведению $n \times m$ числа уравнений и переменных в системе.

На основе полученной выборки строились экспериментальные зависимости потребления времени от параметров, характеризующих сгенерированные системы. Для каждого графика результатов решения вычислялись медиана и процентили $Q_{5\%}$ и $Q_{95\%}$. Так как среднее в экспериментах практически не отличалось от медианы, то первое на графиках не представлено.

На рис. 9(a) и 9(b) показано распределение времени решения по номерам генерируемых систем. Выборка выглядит достаточно однородной, т.к. нет явных трендов



(a) Алгоритм syntactic



(b) Алгоритм slopes

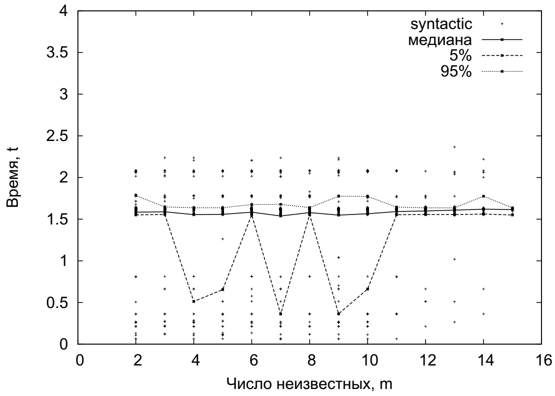
Рис. 10. Распределение времени решения по числу уравнений n в системе одАНЛДУ

в поведении алгоритмов решения с увеличением порядкового номера решаемой системы. Распределение времени решения по уровням вызывается дискретным характером систем одАНЛДУ и, в первую очередь, связано с размерностью базиса Гильберта генерируемых систем: для сходных размерностей, системы одАНЛДУ, имеющие одинаковую размерность базиса Гильберта решаются примерно за одинаковое время. Это важное практическое свойство алгоритма решения особенно типично для syntactic solver. Для slopes оно менее характерно, т.к. существенное влияние на его время решения оказывает размерность коэффициентов.

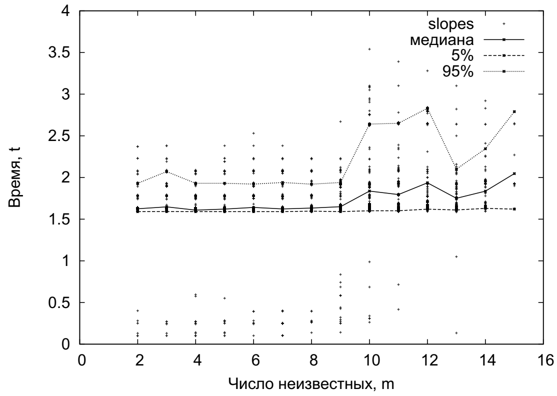
Для используемой размерности $n \times m$ эффективность обоих алгоритмов сравнима (если не учитывать системы, не решенные slopes). Однако, syntactic solver более стабилен с точки зрения верхней границы на время решения: последнее не превосходит 2.5 секунды для всех систем, а аналогичный показатель slopes — около 4 секунд.

На рис. 10(a) и 10(b) показано распределение времени решения в зависимости от числа уравнений в системе. Реализация syntactic solver опять показывает тенденцию к более компактной верхней границе, нежели реализация slopes. Кроме того, поведение $Q_{5\%}$ показывает, что многие системы решаются syntactic solver существенно быстрее, нежели в типичном случае.

На рис. 11(a) и 11(b) показано распределение времени решения по числу неизвестных в системе. Для реализации slopes заметен характерный рост времени при увеличении размерности, начиная с $m = 9$. В то же время, реализация syntactic solver для таких размерностей продолжает сохранять эффективность.

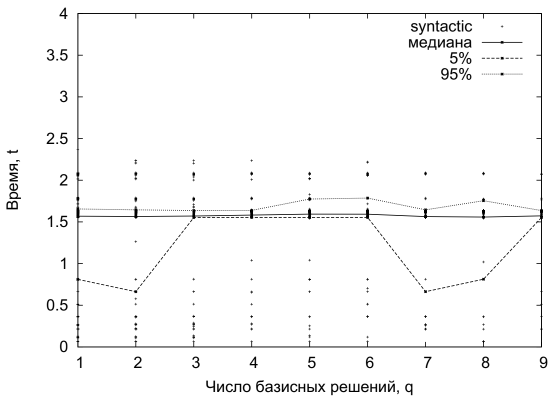


(a) Алгоритм syntactic

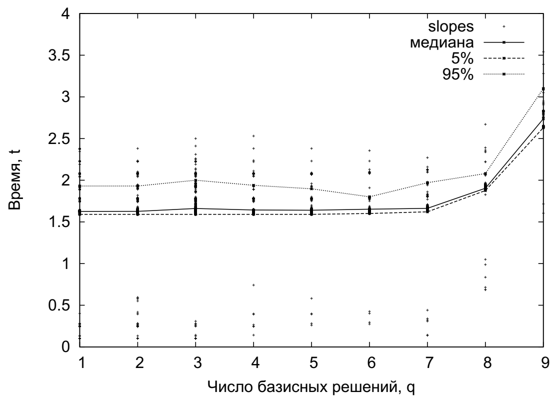


(b) Алгоритм slopes

Рис. 11. Распределение времени решения по числу неизвестных m в системе одАНЛДУ



(a) Алгоритм syntactic

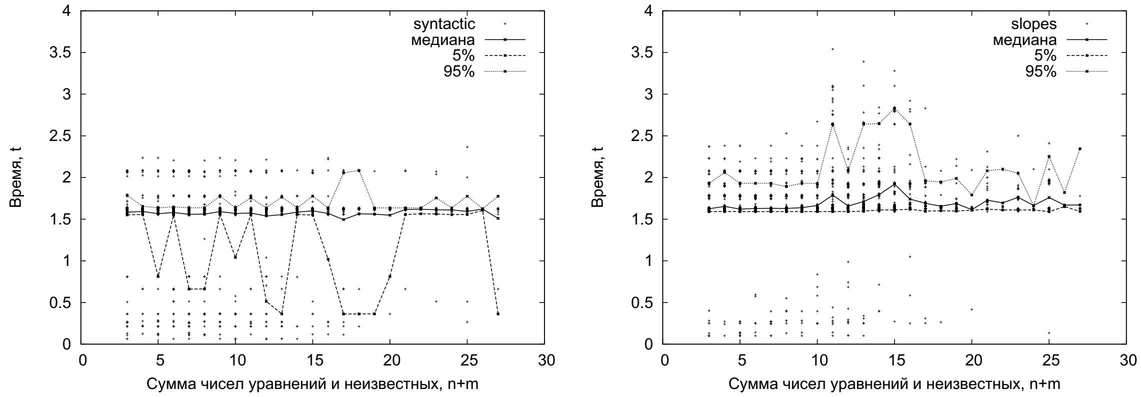


(b) Алгоритм slopes

Рис. 12. Распределение времени решения по числу векторов базиса Гильберта

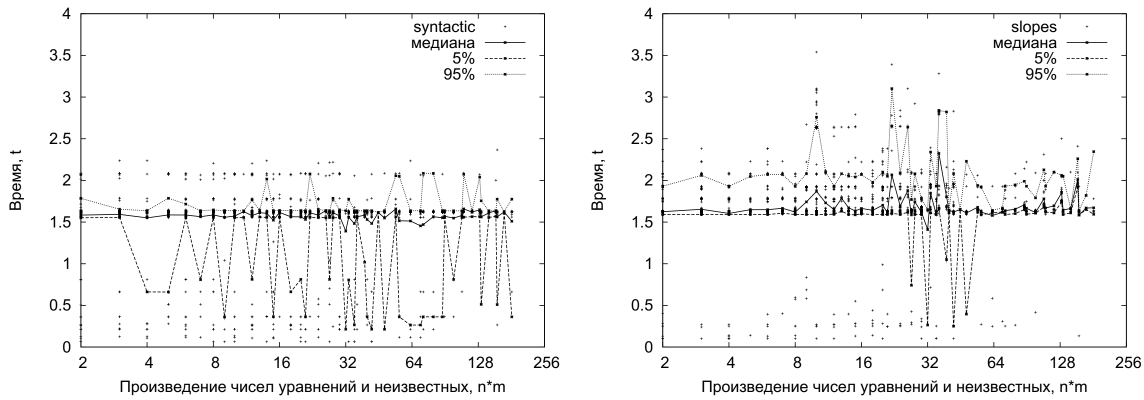
На рис.12(a),12(b) показано распределение времени решения в зависимости от размера базиса Гильберта. Для реализации slopes наблюдается существенный рост времени решения, начиная с $q = 7$. В то же время, реализация syntactic solver менее чувствительна к такому росту размерности. Это подтверждает теоретические оценки сложности этих алгоритмов. Для slopes она имеет экспоненциальный рост из-за переборного характера алгоритма. В случае syntactic solver, оценка наихудшего случая — полиномиальна [5].

На рис. 13(a)–13(b) и рис. 14(a)–14(b) представлены зависимости времени решения от интегральных параметров размерности решаемых систем: $n+m$ и nm соответственно. В последнем случае используется логарифмическая шкале для оси X. Полученные экспериментальные зависимости опять подтверждают более компактную верхнюю границу



(a) Алгоритм syntactic (b) Алгоритм slopes

Рис. 13. Распределение времени решения по сумме чисел уравнений и неизвестных



(a) Алгоритм syntactic (b) Алгоритм slopes

Рис. 14. Распределение времени решения по произведению чисел уравнений и неизвестных

для реализации syntactic solver по сравнению с реализацией slopes, равно как и более вариативную нижнюю границу.

Суммируем основные заключения проведенного сравнительного анализа.

— Сравнимость syntactic solver и slopes на малых размерностях. Однако, медиана показывает следующую небольшую разницу в пользу синтаксического алгоритма: 1.57 сек. у синтаксического против 1.60 сек. у алгоритма slopes. Кроме того, синтаксический алгоритм более устойчив к изменению размерности системы и базиса Гильберта по сравнению с алгоритмом slopes.

— Существенный рост времени решения при небольшом возрастании размерности характерен лишь для slopes. В силу этого не имеет большого смысла проводить сравнительный анализ на больших размерностях.

— Более компактная верхняя граница и более вариабельная нижняя граница для времени решения — у syntactic solver. Эти свойства являются практически важными. Перое обеспечивает более лучшее гарантированное время решения, а второе позволяет в ряде случаев решать системы более быстро, нежели в типичном случае.

4.4. Система Web-SynDic

Система Web-SynDic — это Web-ориентированная система для удаленной демонстрации и тестирования алгоритмов решения неотрицательных линейных диофантовых уравнений. Данная система разрабатывается в рамках студенческих программных проектов на кафедре Информатики и математического обеспечения Петрозаводского государственного университета. В настоящее время автор данной магистерской диссертации является менеджером этой разработки.

Система Web-SynDic является системой научного и прикладного сервиса Интернет и предназначена для удаленного решения следующих задач, связанных с алгоритмами решения систем одАНЛДУ, в первую очередь, синтаксических.

1) Демонстрация эффективности работы алгоритмов решения.

2) Распределенное тестирование реализаций алгоритмов решения.

3) Экспериментальный и сравнительный анализ эффективности алгоритмов решения.

4) Вычислительный сервис решения систем одАНЛДУ, в том числе и систем большой размерности.

Система Web-SynDic поддерживает следующие возможности:

— Генерация и решение одной системы одАНЛДУ.

— Генерация и решение множества систем одАНЛДУ.

— Регистрация пользователей и сохранение настроек для зарегистрированных пользователей.

— Конфигурация алгоритмов генерации и решения.

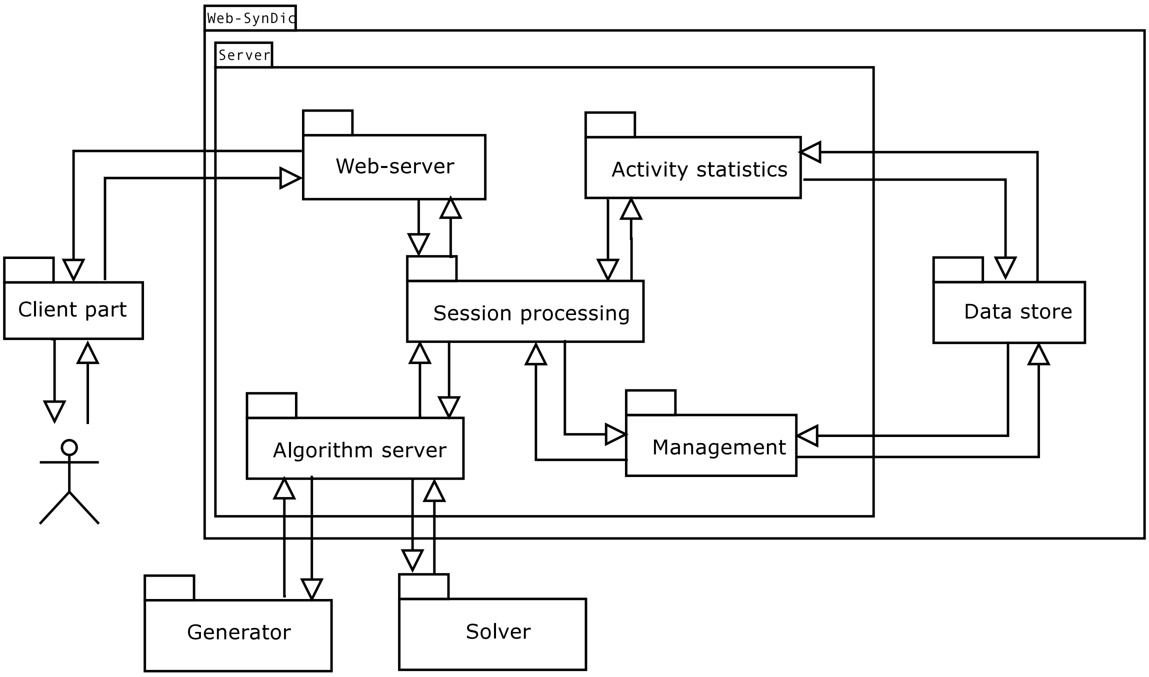


Рис. 15. Архитектура системы Web-SynDic

— Сбор и представление статистической информации о работе системы.

Общая архитектура системы Web-SynDic представлена на рис. 15. Алгоритмы генерации и решения систем НЛДУ являются внешними объектами по отношению к системе Web-SynDic. Тем самым, обеспечивается недоступность самих реализаций алгоритмов пользователям.

Подсистема "Algorithm server" обеспечивает взаимодействие алгоритмов генерации и решения с системой Web-SynDic, являясь, тем самым, ядром алгоритмической части системы в целом. Реализация подсистемы основана на предложенных нами алгоритмах генерации (см. гл. 2) и способах измерения вычислительных ресурсов для алгоритмов решения (см. пп. 4.2 и 4.3). Общая архитектура подсистемы представлена на рис. 16. Система Web-SynDic реализована на языке Java.

В подсистему "Algorithm server" входят два объекта "Generator Spooler" и "Solver Spooler", которые обеспечивают последовательные процессы генерации и решения соответственно. Эти объекты реализованы в виде отдельных нитей и позволяют производить процессы генерации и решения систем НЛДУ независимо от процесса работы системы Web-SynDic.

Для взаимодействия между алгоритмами и системой Web-SynDic реализованы два класса "Generator" и "Solver". Эти обеспечивает гибкость подключения новых алгоритмов генерации и решения. Для анализа и проверки корректности решений используется

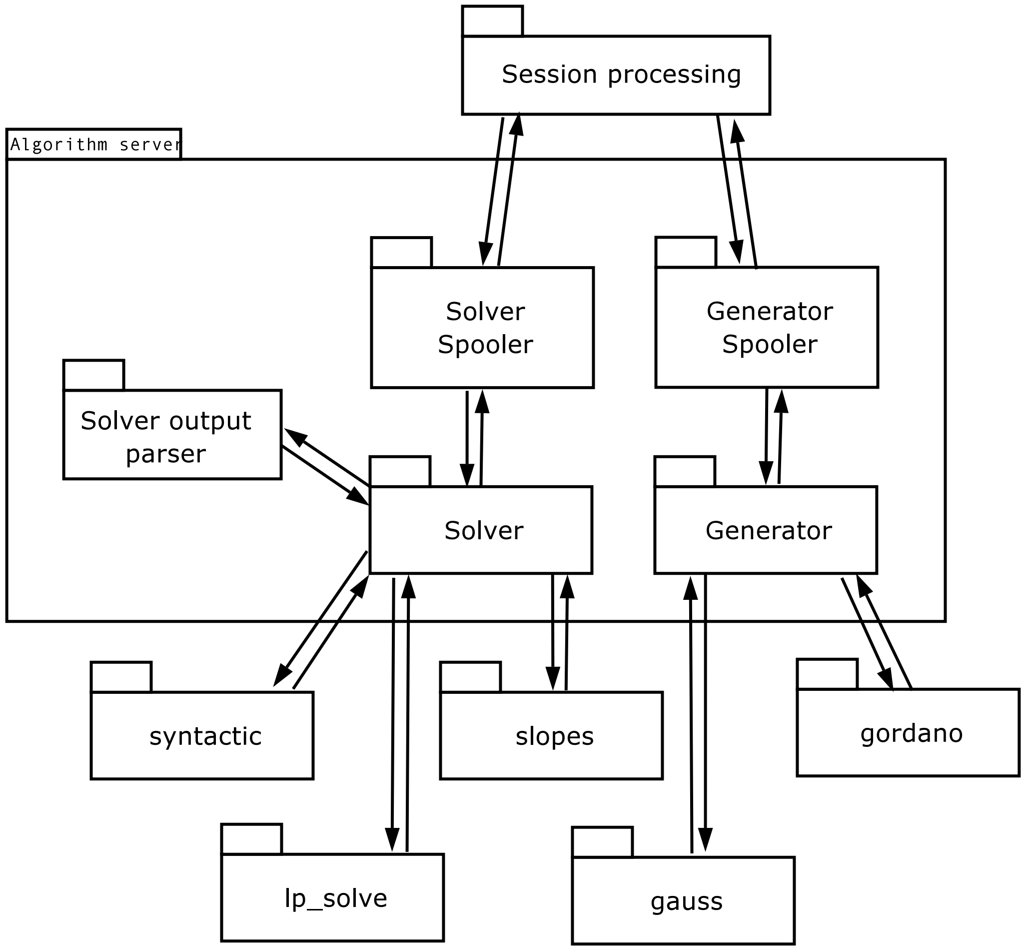


Рис. 16. Архитектура подсистемы “Algorithm server”

“Solver output parser”.

Общая схема работы подсистемы “Algorithm server” представлена на рис 17. Задачи, полученные от пользователя, поступают в очередь для генерации и решения соответственно. “Generator spooler” (“Solver spooler”) считывает задачу из очереди и отправляет на генерацию (решение) соответствующему для требуемого алгоритма расширению класса “Generator” (“Solver”). Последний вызывает внешний алгоритм, который генерирует (решает) систему одАНЛДУ.

Одним из основных требований к работе с внешними алгоритмами является их автономная работа и анализ ошибочных ситуаций. Пользователю информация об ошибке должна возвращаться в удобочитаемой форме. Наиболее типичными ошибочными ситуациями являются превышение объема разрешенной к использованию памяти и превышение времени работы. Для этих двух случаев предусмотрены отдельные сообщения об ошибке, а остальные сгруппированы в две категории — сообщение об ошибке алгоритма и сообщение о неправильном результате работы алгоритма. В случае, если потребуется вы-

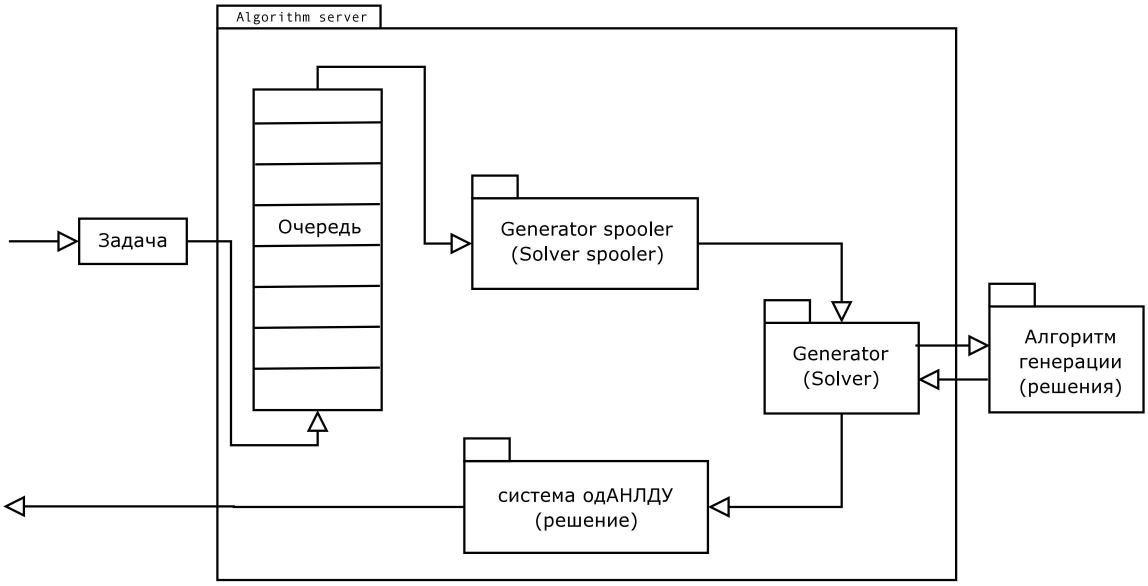


Рис. 17. Схема работы подсистемы “Algorithm server”

делить дополнительную категорию ошибочных ситуаций, достаточно внести изменения в соответствующие расширения классов “Generator” и “Solver”.

В таблице 3 приведены некоторые метрики реализации подсистемы “Algorithm server” и системы Web-SynDic, такие как количество строк кода (LOC), количество пустых строк кода (BLOC), количество строк с комментариями (CLOC), количество строк без комментариев (NCSL), процент строк без комментариев к общему количеству строк кода (%(NCSL)). Остальные метрики реализации и тестирования можно найти в [30].

Таблица 3. Характеристики реализации системы Web-SynDic

Метрика	подсистема “Algorithm server”	система Web-SynDic
LOC	3805	11907
BLOC	491	1356
CLOC	1275	2757
NCSL	2207	5356
%(NCSL)	58	45

На рис. 18 представлено фрагмент формы для работы с одиночной системой одАНЛДУ. Меню в левой части окна позволяет выбрать соответствующее действие в системе Web-SynDic: работа с одиночной системой одАНЛДУ, работа с множеством систем одАНЛДУ, настройка параметров и ограничений для алгоритмов генерации и решения и т.п. На форме работы с одиночной системой одАНЛДУ можно ввести систему вручную либо воспользоваться одним из алгоритмов генерации; решить систему с помощью синтаксического и выбранного алгоритмов решения или записать эту систему одАНЛДУ в

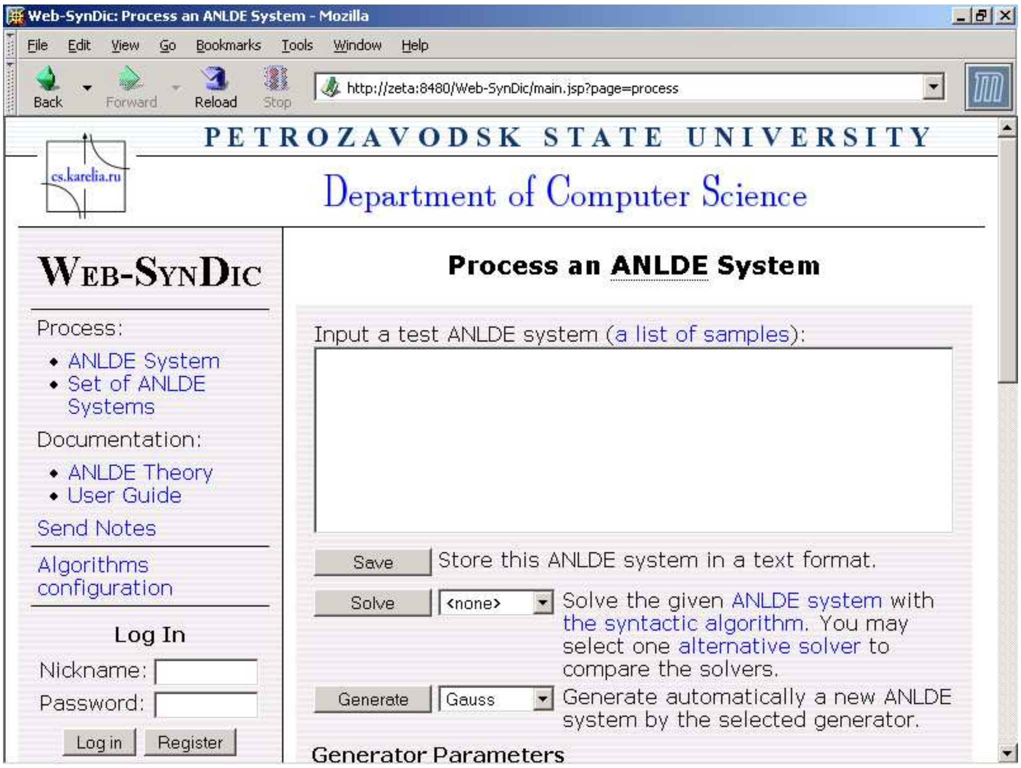


Рис. 18. Форма работы с одиночной системой одАНЛДУ

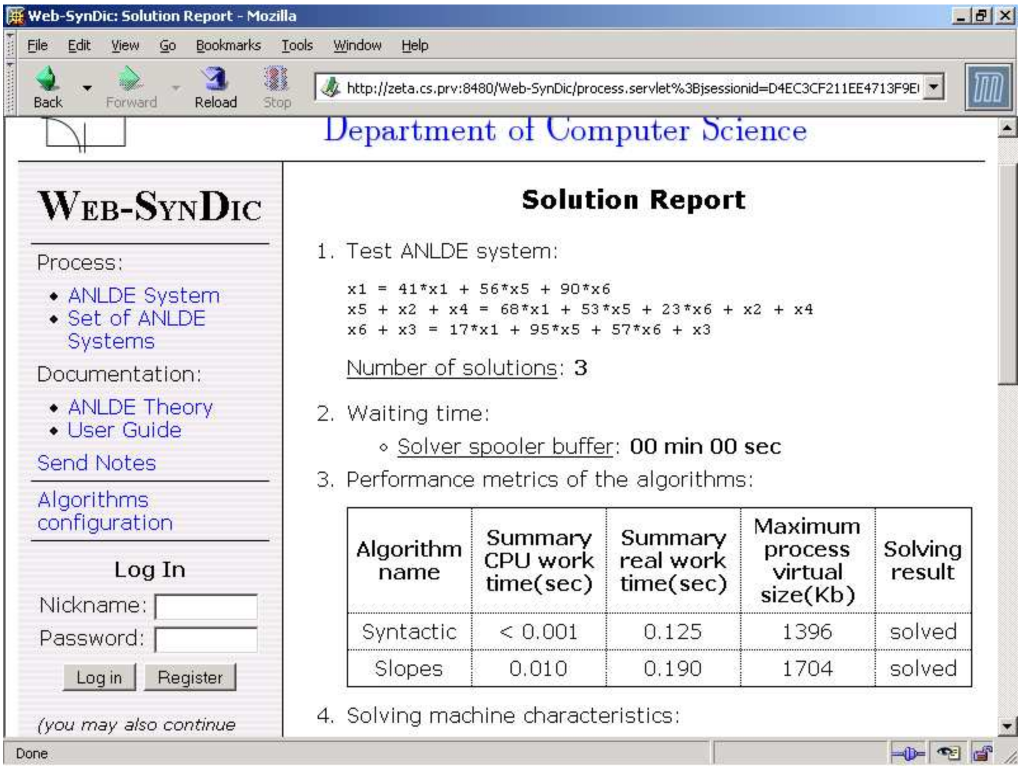


Рис. 19. Отчет о решении системы одАНЛДУ

текстовом формате.

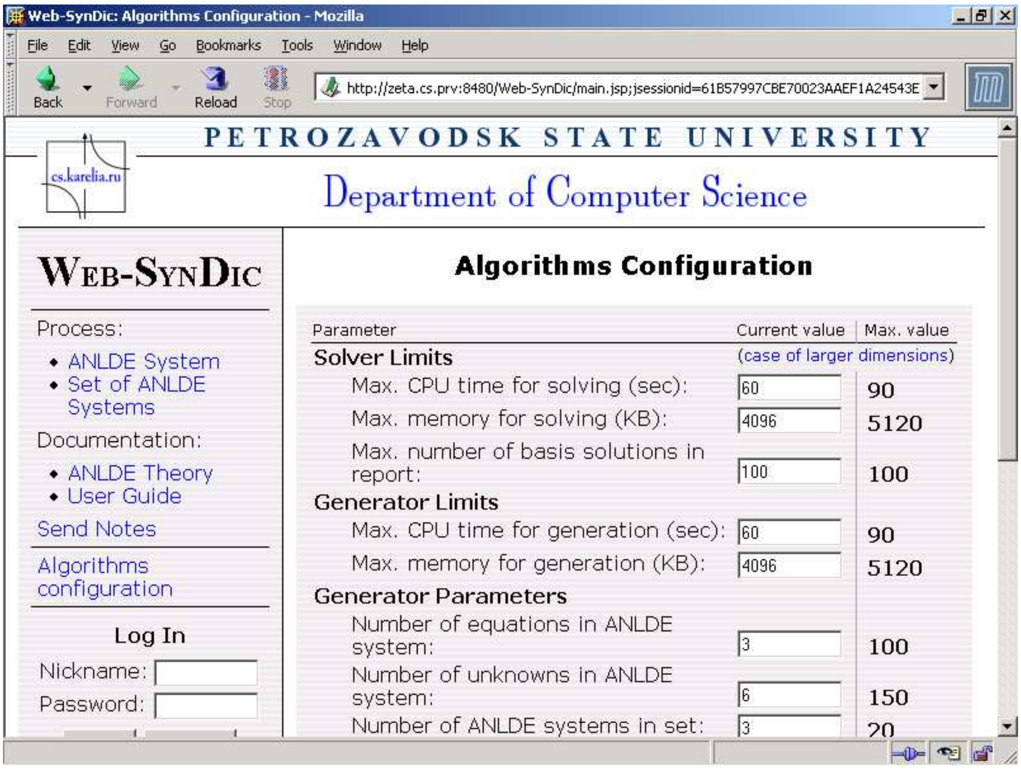


Рис. 20. Форма настройки параметров и ограничений алгоритмов генерации и решения систем одАНЛДУ

На рис. 19 представлен фрагмент отчета о решении одиночной системы одАНЛДУ. Отчет содержит информацию о системе одАНЛДУ, время ожидание в буфере задач, метрики решения для выбранных решателей и, разумеется, сами решения для каждого из выбранных решателей.

На рис. 20 представлен фрагмент формы настройки параметров и ограничений для алгоритмов генерации и решения систем одАНЛДУ. Каждый параметр или ограничение имеет верхнюю и нижнюю[2] границы. Эти границы устанавливаются администратором системы и действуют только на настройки обычных пользователей.

Выводы к главе

В этой главе была представлена технология для проведения тестирования, экспериментального анализа и сравнения алгоритмов решения систем одАНЛДУ.

1. Выполнена постановка задач тестирования, экспериментального и сравнительного анализа алгоритмов решения систем одАНЛДУ, а также определены основные тре-

[2]По умолчанию нижняя граница равна единице. Исключением является параметр количества решений в отчете — для него нижняя граница равна нулю.

бования к этим задачам.

2. Предложены способы измерения вычислительных ресурсов, таких как затрачиваемое время и используемая память. Данные способы существенным образом базируются на стандартизованных средствах системного и межплатформенного программирования. Данные способы и предложенные в главе 2 алгоритмы генерации составляют технологию автоматического тестирования, экспериментального анализа и сравнения алгоритмов решения систем одАНЛДУ. Эта технология была реализована в виде мобильной программной системы alg_analyser.

3. На основе системы alg_analyser было выполнено масштабное системное тестирование и экспериментальный анализ реализации syntactic solver синтаксического алгоритма решения систем одАНЛДУ, а также сравнительный анализ данной реализации с реализацией альтернативного алгоритма решения slopes. Результаты подтверждают высокое качество реализации syntactic solver и эффективность синтаксического алгоритма.

4. Разработанное ПО включено в состав системы Web-SynDic, составляя ядро ее алгоритмической части. Это позволяет развивать предложенную технологию в направлении поддержки научного и прикладного вычислительного сервиса в Интернет.

Заключение

В качестве основных теоретических результатов данного исследования следует выделить следующие.

1. Выполнено развитие методов теории систем одАНЛДУ. Сформулирована теорема о преобразовании произвольной системы одАНЛДУ к двум частным классам таких систем. Один из таких классов, а именно, системы содАНЛДУ, определяется и исследуется нами впервые.

2. Разработаны четыре алгоритма генерации частных классов систем одАНЛДУ, включая алгоритм генерации систем содАНЛДУ, и один алгоритм генерации произвольной системы одАНЛДУ. Это позволяет более адекватно учитывать специфику приложений задач генерации — тестирование и экспериментальный анализ алгоритмов решения систем одАНЛДУ.

3. Исследовано свойство двойственности задач генерации и решения систем одАНЛДУ, на основании чего разработаны двойственные алгоритмы решения систем содАНЛДУ и одАНЛДУ.

4. Разработана технология автоматизации тестирования, экспериментального анализа и сравнения алгоритмов решения систем одАНЛДУ. Это позволит выполнять комплексную оценку качества и эффективности таких алгоритмов.

Из практических результатов выделим следующие.

1. Реализованы три первых из предложенных алгоритмов генерации систем одАНЛДУ. Таким образом, начата разработка библиотеки алгоритмов генерации, которая необходима для адекватного решения задач тестирования и экспериментального анализа алгоритмов решения систем одАНЛДУ.

2. Предложенная технология автоматизации тестирования, экспериментального анализа и сравнения алгоритмов решения систем одАНЛДУ реализована в виде мобильной программной системы alg_analyser (ОС Linux и Windows).

3. Выполнено масштабное системное тестирование и экспериментальный анализ реализации syntactic solver синтаксического алгоритма решения систем одАНЛДУ. Выполнен сравнительный анализ этой реализации и реализации альтернативного алгорима slopes. Результаты позволяют гарантировать высокое качество реализации syntactic solver и экспериментально подтвержают теоретически обоснованную эффективность синтаксического алгоритма.

4. Система alg_analyser была использована при разработке и реализации системы удаленной демонстрации и тестирования алгоритмов решения систем одАНЛДУ Web-SynDic.

Полученные результаты требуют дальнейшего развития. Предлагается продолжить эти исследования. Наиболее интересными направлениями здесь представляются следующие.

— Разработка новых алгоритмов генерации систем одАНЛДУ. Это приведет к увеличению разнообразия классов генерируемых систем, что требуется в задачах тестирования и экспериментального анализа. Привлечение вероятностных методов анализа задачи генерации с целью более детальной характеризации множества систем, генерируемых заданным алгоритмом.

— Развитие других методов тестирования, отличных от основанных на сравнении базисов Гильберта. Например, проверка только на частные решения. Это позволит выполнять экспериментальный анализ и сравнение более широкого круга алгоритмов решения.

— Развитие методов экспериментального и сравнительного анализа реализаций алгоритмов решения систем одАНЛДУ, в частности, основанных на статистической обработке данных измерений. Это позволит дать более качественные оценки эффективности. Продолжение экспериментального исследования синтаксического алгоритма на различных диапазонах размерностей систем одАНЛДУ с целью получить детальную карту его эффективности, а также привлечение новых альтернативных ему алгоритмов решения для проведения комплексного сравнительного анализа.

— Продолжение развития системы Web-SynDic с целью получения профессионального инструмента для выполнения научного и прикладного вычислительного сервиса теории систем одАНЛДУ.

Приложение

Связь систем одАНЛДУ с контекстно-свободными грамматиками

Произвольная система одАНДДУ (1.1) ассоциирована с КС-грамматикой следующим образом [5]. Грамматика состоит из m правил $(r_i)_{i=1}^m$ и n нетерминалов $(\mathcal{A}_k)_{k=1}^n$. Нетерминальный алфавит пуст. Для каждого нетерминала $\mathcal{A}$ введем обозначение $|\alpha|_{\mathcal{A}}$ для числа вхождений $\mathcal{A}$ в строку α. Отображение Парикха Ψ_n от $\{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n\}^*$ в $\mathbb{Z}_+^n$ определено как $\Psi_n(\alpha) = (|\alpha|_{\mathcal{A}_1}, |\alpha|_{\mathcal{A}_2}, \ldots, |\alpha|_{\mathcal{A}_n})$. Правило грамматики $r_i = (\mathcal{A}_k \to \alpha_i)$ для некоторого $k \in \{1, 2, \ldots, n\}$ и нетерминальной цепочки α_i. Это правило определяет, что $\mathrm{E}_{ki} = 1$, $\mathrm{E}_{ji} = 0$ $\forall\, j \neq k$ и i-й столбец матрицы A есть $\Psi_n(\alpha_i)$.

Таким образом, неизвестные системы одАНЛДУ ассоциированы с правилами грамматики а уравнения — с нетерминалами. Значение x_i определяет количество применений правила r_i в некотором выводе. Существует соответствие между множеством всех решений системы одАНЛДУ и множеством всех циклических выводов $A_k \Rightarrow^* A_k$, $k = 1, 2, \ldots, n$ в грамматике. Любое решение связано с соответствующим выводом (выводами) и наоборот (см. [5] для деталей и доказательств, некоторые примеры также могут быть найдены в [6]). Оригинальная идея восходит к M. Filgueiras и A. P. Tomás [23].

Пример 4.1. Для примера 1.3 соответствующая КС-грамматика

$$\begin{aligned}
r_1 : &\quad A \to B^5 \\
r_2 : &\quad B \to \varepsilon \\
r_3 : &\quad A \to B^7 \\
r_4 : &\quad C \to A^2B^2 \\
r_5 : &\quad C \to AB^3C^3
\end{aligned}$$

Например, базисное решение $h^{(1)} = (5, 32, 0, 2, 1)^{\mathsf{T}}$ соответствует в циклическому выводу $C \stackrel{r_5}{\Longrightarrow} AB^3C^3 \stackrel{2\times r_4}{\Longrightarrow} A^5B^7C \stackrel{5\times r_1}{\Longrightarrow} B^{32}C \stackrel{32\times r_2}{\Longrightarrow} C$.

Библиографический список использованной литературы

1. Богоявленский Ю. А., Дуплихин М. Ю. *Некоторые направления современных исследований алгоритмов линейного прогрммирования* // Труды Петрозаводского государственного университета. Сер. “Прикладная математика и информатика”. Вып. 7. — Петрозаводск: Изд-во ПетрГУ, 1998. — С. 209–222.
2. Богоявленский Ю. А., Корзун Д. Ж. *Общий вид решения системы линейных диофантовых уравнений, ассоциированной с контекстно-свободной грамматикой* // Труды Петрозаводского государственного университета. Сер. “Прикладная математика и информатика”. Вып. 6. — Петрозаводск: Изд-во ПетрГУ, 1997. — С. 79–94.
3. Корзун Д. Ж. *Об одной взаимосвязи формальных грамматик и систем линейных диофантовых уравнений* // Вестник молодых ученых, 2000. № 3. — СПб: Изд-во СПбГТУ, 2000. — С. 50–56.
4. Корзун Д. Ж. *Решение одного класса линейных диофантовых уравнений в неотрицательных целых числах методами теории формальных языков* // Труды Петрозаводского государственного университета. Сер. “Прикладная математика и информатика”. Вып. 7. — Петрозаводск: Изд-во ПетрГУ, 1999. — С. 93–116.
5. Корзун Д. Ж. *Синтаксические алгоритмы решения неотрицательных линейных диофантовых уравнений и их приложение к моделированию структуры нагрузки канала Интернет.* Дисс. на соиск. канд. физ.-мат. наук. Петрозаводск, ПетрГУ, 2002. 185 с.
6. Корзун Д. Ж. *Grammar-Based Algorithms for Solving Certain Classes of Nonnegative Linear Diophantine Systems* // Труды международного семинара Finnish Data Processing Week at the University of Petrozavodsk (FDPW’2000): Advances in Methods of Modern Information Technology. Vol. 3. — Петрозаводск: Изд-во ПетрГУ, 2001. — С. 52–67.
7. Корзун Д. Ж. *Syntactic Methods in Solving Linear Diophantine Equations* // Труды международного семинара Finnish Data Processing Week at the University of Petrozavodsk (FDPW’2004): Advances in Methods of Modern Information Technology. Vol. 6. — Петрозаводск: Изд-во ПетрГУ, 2005. — С. 151–156.

8. Корзун Д. Ж., Богоявленский Ю. А., Кулаков К. А., Сало А. Ю., Крышень М. Ю., Ананьин А. В. *Система Web-SynDic для демонстрации и тестирования синтаксических алгоритмов решения линейных диофантовых уравнений в неотрицательных целых числах* // Материалы Всероссийской конференции "Научный сервис в сети интернет ’2004". Москва, 2004. — С. 8–10.
9. Кострикин А. И., Манин Ю. И. *Линейная алгебра и геометрия.* — М.: Наука, 1986.
10. Кулаков К. А. *Однородные ассоциированные с формальными грамматиками системы неотрицательных линейных диофантовых уравнений* // Материалы 57-й научной студенческой конференции ПетрГУ. Секция «Информатика». Принято к печати. — Петрозаводск: Изд-во ПетрГУ, 2005.
11. Кулаков К. А. *Технология автоматизации тестирования алгоритмов решения неотрицательных линейных диофантовых уравнений* // Материалы межвузовского конкурса-конференции "Технологии Microsoft в теории и практике программирования". — СПб: Издательство СПбГПУ, 2004. — С. 142–143.
12. Кулаков К. А. *Технология тестирования, экспериментального анализа и сравнения алгоритмов решения неотрицательных линейных диофантовых систем* // Материалы 54-й научной студенческой конференции ПетрГУ. Секция «Информатика». — Петрозаводск: Изд-во ПетрГУ, 2002. — С. 173–174.
13. Кулаков К. А. Корзун Д. Ж. *Generating Homogenious Systems of Equations for Testing and Experimental Analysis of Linear Diophantine Solvers* // Труды международного семинара Finnish Data Processing Week at the University of Petrozavodsk (FDPW’2003): Advances in Methods of Modern Information Technology. Vol. 5. — Петрозаводск: Изд-во ПетрГУ, 2005. — С. 259–278.
14. Кулаков К. А., Сало А. Ю., Ананьин А. В., Крышень М. Ю. *Web-SynDic — система демонстрации и тестирования синтаксических алгоритмов решения неотрицательных линейных диофантовых уравнений* // Материалы межвузовского конкурса-конференции "Технологии Microsoft в теории и практике программирования". — СПб: Издательство СПбГПУ, 2004. — С. 43–44.
15. Нефедов В. Н., Осипова В. А. *Курс дискретной математики.* — М.: Изд-во МАИ, 1992. — 264 с.
16. Смирнова Е. С. *Разработка и реализация алгоритма обработки нечетких данных в многоуровневых логиках.* Дисс. на соиск. канд. физ.-мат. наук. Санкт-Петербург, Санкт-Петербургский государственный университет, 2002. 135 с.

17. Схрейвер А. *Теория линейного и целочисленного программирования* : Пер. с англ. Т.1. — М.: Мир, 1991. — 360 с.; Т.2. — М.: Мир, 1991. — 342 с.
18. Contejean E., Devine H. *An efficient incremental algorithm for solving systems of linear diophantine equations* // Information and Computation. — 1994. Vol. 113. No. 1. — pp. 143–172.
19. Cygwin. *Project home page.* `http://www.cygwin.com`
20. Domenjoud E., Tomás A. P. *From Elliot-MacMahon to an algorithm for general constraints on naturals* // In U. Montanari, F. Rossi (eds.), Principles and Practice of Constraint Programming (CP'95). Springer–Verlang, 1995. LNCS 976. — PP. 18–35.
21. Elliot E. B. *On linear homogeneous diophantine equations* // Quartely Journal of Pure and Applied Maths. 1903. Vol. 136.
22. Filgueiras M., Tomás A. P. *A fast method for finding the basis of nonnegative solutions to a linear Diophantine equation* // Journal of Symbolic Computation. — 1995. Vol. 19. — PP. 507–526.
23. Filgueiras M., Tomás A. P. *Solving Linear Constraints on Finite Domains through Parsing* // In P. Barahona, L. Moniz Pereira, A. Porto (eds.), Proceedings of the 5th Portuguese Conference on Artificial Intelligence, Springer-Verlag, 1991. LNAI 541. — PP. 1–16.
24. Giles F., Pulleyblank F. *Total dual integrality and integer polyedra.* Linear algebra and its applications. No. 25, 1979. pp. 191–196.
25. Huet G. *An algorithm to generate the basis of solutions to homogenous linear diophantine equations* // Information Processing Letters. — 1978. Vol. 3. No. 7. — PP. 144–147.
26. Pottier L. *Minimal solutions of linear diophantine systems: bounds and algorithms* // Proceedings of the 4th International Conference on Rewriting Techniques and Applications (RTA'91), Como (Italy). — 1991. — PP. 162–173
27. Sommerville I. *Software Engineering.* 6th ed. Addison-Wesley, 2000.
28. Tomás A. P., Filgueiras M. *Algorithms for solving Diophantine Equations in Naturals.* DCC-FC & LIACC, Universidade do Porto, 2001.
`http://www.ncc.up.pt/ apt/dioph/`
29. Tomás A. P., Filgueiras M. *An algorithm for solving systems of linear Diophantine equations in naturals* // In E. Costa, A. Cardoso (eds.), Progress in Artificial Intelligence (EPIA'97). Springer-Verlag, 1997. LNAI 1323. — PP. 73–84.
30. Web-SynDic system. *Project documentation.* `http://websyndic.cs.karelia.ru.`