СОДЕРЖАНИЕ

Моделирование и анализ информационных систем. Т.11, №2. 2004

Редактор, корректор А.А.Аладьева
Подписано в печать 20.10.2004. Формат 60x84 1/8. Печать офсетная.
Усл.печ.л. 4,65. Уч.-изд.л. 4,05. Тираж 100 экз. Зак. N 104/04

Отпечатано на ризографе. Ярославский государственный университет имени П.Г.Демидова, 150 000, Ярославль, ул.Советская, 14

Министерство образования и науки Российской Федерации
Ярославский государственный университет
имени П.Г.Демидова
Ярославское региональное отделение РАЕН

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ

Том 11 №2 2004

Основан в 1999 г.
Выходит 2 раза в год

Свидетельство о регистрации №019209 от 16.08.99
Государственного Комитета Российской Федерации по печати

Адрес редакции: 150000, Ярославль, ул.Советская, 14
E-mail: mais@uniyar.ac.ru

Научные статьи в журнал принимаются на кафедре ТИ. Статья должна содержать УДК, аннотацию и сопровождаться набором текста в редакторе LaTeX.

Модел. и анализ информ. систем. Т.11, №2 (2004) 3–7

УДК 541.1+612.82

Колебания в сети из пороговых нейронов

Ануфриенко С.Е.
Ярославский государственный университет
150 000, Ярославль, Советская, 14

получена 15 мая 2004

Аннотация

В работе рассматривается модель импульсного проведения возбуждения по гексагональной решетке, в узлах которой расположены пороговые нейроны. Описан возможный периодический режим, рассчитаны временные рассогласования между спайками нейронов.

Введение. Исследованию различных нейронных сетей, динамика которых описывается системой дифференциальных уравнений с запаздыванием, посвящено много работ, например [1] — [2]. Элементами сетей чаще служат нейроны-автогенераторы, которые могут самопроизвольно генерировать высокоамплитудные импульсы (спайки). Нейронная сеть, предложенная в настоящей статье, состоит из пороговых нейронов, которые способны генерировать спайк только в ответ на внешнее воздействие. Волны в таких сетях могут затухать. В статье предложен и исследован незатухающий периодический режим. Модель порогового нейрона предложена в статьях [4] — [5].

Колебания в системе из шести пороговых нейронов. Рассмотрим кольцо из шести одинаковых пороговых нейронов. Каждый нейрон связан с предыдущим и следующим за ним нейронами. Пусть $u_i(t) \geq 0$ — их мембранные потенциалы ($i = 1, \ldots, 6$). Динамика сети описывается следующей системой дифференциальных уравнений:

$$\dot{u}_i = \lambda[-1 - f_{Na}(u_i) + f_K(u_i(t-1))]u_i + \varepsilon + \lambda\epsilon^{-\lambda\sigma}(u_{i-1} + u_{i+1}), \qquad (1)$$

$$i = 1, \ldots, 6.$$

Здесь и далее считаем, что $u_0(t) \equiv u_6(t)$, $u_7(t) \equiv u_1(t)$.

Параметр $\lambda \gg 1$ отражает высокую скорость протекания электрических процессов, параметр $0 < \varepsilon \ll 1$ учитывает токи утечки, проходящие через мембраны нейронов. Положительные достаточно гладкие функции $f_{Na}(u)$ и $f_K(u)$ монотонно убывают к нулю при $u \to \infty$ быстрее, чем $1/u$. Они описывают состояние натриевых и калиевых каналов.

Параметры $\alpha = 1 + f_{Na}(0) - f_K(0) > 0$, $\alpha_1 = f_K(0) - 1 > 1$, $0 < \sigma < \alpha_1$, $\alpha_2 = f_{Na}(0) + 1$.

Число $f_K(0) - f_{Na}(1) - 1 > 0$ характеризует пороговое значение: начало спайка i-го нейрона условно свяжем с моментом времени t_s, таким что $u_i(t_s) = 1$, $u_i(t) < 1$ при $t_s - 1 < t < t_s$.

Зададим начальные условия:

$$u_1(s) = \varphi(s), \quad s \in [-1, 0].$$

Класс начальных функций для первого нейрона состоит из непрерывных на отрезке $s \in [-1, 0]$ функций $\varphi(s)$, удовлетворяющих условиям: $\varphi(0) = 1$ и $0 \leq \varphi(s) \leq max(e^{\lambda\alpha s/2}, 1/\lambda)$. Обозначим этот класс функций S. Из данного условия следует, что в момент времени $t = 0$ начинается спайк первого нейрона.
Для остальных нейронов будем считать, что

$$u_i(s) = u_* \approx \frac{\varepsilon}{\lambda\alpha} \quad \text{при} \quad s \in [-1, 0].$$

Проанализируем систему уравнений (1) при $\lambda \to \infty$. Начало и окончание спайка i-го нейрона условно свяжем с моментами времени, когда $u_i(t)$ пересекает единичное значение соответственно с положительной и отрицательной скоростью. Известно [3], что динамика мембранного потенциала изолированного нейрона описывается формулами:

$$u_1(t) = \begin{cases} e^{\lambda\alpha_1(t+o(1))} & \text{при} \quad t \in [\delta, 1-\delta], \\ e^{\lambda(\alpha_1-(t-1)+o(1))} & \text{при} \quad t \in [1+\delta, 1+\alpha_1-\delta], \\ \frac{\varepsilon+o(1)}{\lambda\alpha_2} & \text{при} \quad t \in [1+\alpha_1+\delta, 2+\alpha_1-\delta], \\ \frac{\varepsilon+o(1)}{\lambda\alpha} & \text{при} \quad t \geq 2+\alpha_1+\delta. \end{cases}$$

Здесь через $o(1)$ обозначены слагаемые, которые стремятся к нулю при $\lambda \to \infty$.

Рассмотрим помежуток времени, начиная с момента $t = \delta$, где $0 < \delta \ll 1$ — произвольно малое фиксированное число, до начала спайка второго нейрона. Будем считать, что по каким-либо причинам во время спайка первого нейрона воздействие на шестой нейрон не оказывается. Учитывая сказанное и то, что $u_1(t-1) = o(1)$, $u_1(t) \gg 1$, $u_i(t) = o(1)$ $i = 2, \ldots, 6$, перепишем систему (1):

$$\begin{aligned} \dot{u}_1 &= \lambda[\alpha_1 + o(1)]u_1, \\ \dot{u}_2 &= \lambda[-\alpha + o(1)]u_2 + \lambda e^{-\lambda\sigma}u_1, \\ \dot{u}_i &= \lambda[-\alpha + o(1)]u_i, \quad i = 3, \ldots, 6. \end{aligned}$$

Начальные условия имеют вид:

$$u_1(0) = 1, \quad u_i(0) = u_*, \quad i = 2, \ldots, 6.$$

Решение системы задается формулами:

$$\begin{aligned} u_1(t) &= e^{\lambda(\alpha_1 t + o(1))}, \\ u_2(t) &= \tfrac{\varepsilon + o(1)}{\lambda\alpha} + e^{\lambda(\alpha_1 t - \sigma + o(1))}, \\ u_i(t) &= u_*, \quad i = 3, \ldots, 6. \end{aligned}$$

Данные формулы верны до тех пор, пока $u_2(t) < 1$. Найдем τ из условия $u_2(\tau) = 1$:

$$\tau = \frac{\sigma}{\alpha_1} + o(1).$$

Заметим, что $\tau < 1$.

Рассмотрим систему (1) на промежутке времени $t \in [\tau + \delta, t_1]$, где $t_1 < min(1, t_2)$ (t_2 — момент начала спайка третьего нейрона). Имеем:

$$\begin{aligned} \dot{u}_1 &= \lambda[\alpha_1 + o(1)]u_1 + \lambda e^{-\lambda\sigma}u_2, \\ \dot{u}_2 &= \lambda[\alpha_1 + o(1)]u_2 + \lambda e^{-\lambda\sigma}u_1, \\ \dot{u}_3 &= \lambda[-\alpha + o(1)]u_3 + \lambda e^{-\lambda\sigma}u_2, \\ \dot{u}_i &= \lambda[-\alpha + o(1)]u_i, \quad i = 4, 5, 6. \end{aligned}$$

Начальные условия имеют вид:

$$u_1(\tau) = e^{\lambda\alpha_1\tau}, \quad u_2(\tau) = 1, \quad u_i(\tau) = u_*, \quad i = 3, \ldots, 6.$$

В силу непрерывности на некотором промежутке справедливо неравенство: $u_1(t) > u_2(t)$. В этом случае второй нейрон не может повлиять на первый, поэтому имеем:

$$u_1(t) = e^{\lambda(\alpha_1 t + o(1))}.$$

По теореме сравнения получаем:

$$u_2(t) \geq u_1(t - \tau).$$

Учитывая, что $\tau < \sigma$, убеждаемся в отсутствии воздействия на второй нейрон со стороны первого. Таким образом, получаем:

$$u_2(t) = e^{\lambda(\alpha_1(t-\tau) + o(1))}.$$

Следовательно, для u_3 получаем уравнение:

$$\dot{u}_3 = \lambda[-\alpha + o(1)]u_3 + \lambda e^{\lambda(\alpha_1(t-\tau) - \sigma + o(1))}.$$

Его решение имеет вид:

$$u_3(t) = \frac{\varepsilon + o(1)}{\lambda\alpha} + e^{\lambda(\alpha_1(t-\tau) - \sigma + o(1))}.$$

Найдем t_2 из условия $u_3(t_2) = 1$, получим: $t_2 = 2\tau$.

Позже мы покажем, что из условия существования волнового режима в нейронном кольце следует: $2\tau > 1$, поэтому приведенные формулы справедливы на промежутке $t \in [\tau + \delta, 1 - \delta]$.

Рассмотрим систему (1) на промежутке времени $t \in [1+\delta, 2\tau-\delta]$. Имеем:

$$\begin{aligned}
\dot{u}_1 &= \lambda[-1+o(1)]u_1 + \lambda e^{-\lambda\sigma}u_2,, \\
\dot{u}_2 &= \lambda[\alpha_1+o(1)]u_2 + \lambda e^{-\lambda\sigma}u_1, \\
\dot{u}_3 &= \lambda[-\alpha+o(1)]u_3 + \lambda e^{-\lambda\sigma}u_2, \\
\dot{u}_i &= \lambda[-\alpha+o(1)]u_i, \quad i=4,5,6.
\end{aligned}$$

Начальные условия имеют вид:

$$u_1(1) = e^{\lambda\alpha_1}, \quad u_2(1) = e^{\lambda\alpha_1(1-\tau)}, \quad u_3(1) = u_* + o(1), \quad u_i(1) = u_*, \quad i=4,5,6.$$

Рассуждения, аналогичные приведенным раньше, показывают, что первый и второй нейроны не оказывают влияния друг на друга, поэтому решение системы на данном промежутке задается формулами:

$$\begin{aligned}
u_1(t) &= e^{\lambda(\alpha_1-(t-1)+o(1))}, \\
u_2(t) &= e^{\lambda(\alpha_1(t-\tau)+o(1))}, \\
u_3(t) &= \tfrac{\varepsilon+o(1)}{\lambda\alpha} + e^{\lambda(\alpha_1(t-\tau)-\sigma+o(1))}, \\
u_i(t) &= u_*, \quad i=4,5,6.
\end{aligned}$$

Продолжая анализировать систему (1), можно показать, что во время спайка и на интервале времени длины единица после него нейрон невосприимчив к воздействию извне.

По кольцу будет распространяться волна нейронной активности, временные рассогласования между спайками соседних нейронов равны τ с точностью до слагаемых $o(1)$, время между последовательными спайками одного нейрона равно $6\tau + o(1)$.

Для существования данного режима нужно, чтобы во время начала спайка каждого нейрона следующий за ним нейрон вышел из рефрактерного состояния:

$$5\tau > \alpha_1 + 2.$$

Отсюда получаем:

$$\sigma > \frac{\alpha_1(\alpha_1+2)}{5}. \tag{2}$$

Неравенство (2) не противоречит условиям $\alpha_1 > 1$ и $\sigma < \alpha_1$.

Колебания в сети из пороговых нейронов на плоскости. Рассмотрим на плоскости гексагональную решетку. Ячейки решетки представляют собой правильные шестиугольники. В узлах решетки разместим нейроны. Каждый внутренний узел связан с тремя другими. Введем двухиндексную нумерацию. Через $Z_{i,j}$ обзначим множество узлов, связанных с узлом (i,j). Система уравнений, описывающая динамику сети, имеет вид:

$$\dot{u}_{i,j} = \lambda[-1 - f_{Na}(u_{i,j}) + f_K(u_{i,j}(t-1))]u_{i,j} + \varepsilon + \lambda e^{-\lambda\sigma} \sum_{(k,m)\in Z_{i,j}} u_{k,m}. \tag{3}$$

Опишем структуру возможного периодического режима. Выделим на решетке узел (i_0, j_0). Пусть в момент времени $t=0$ у нейрона, расположенного в этом узле, начинается спайк. Для этого зададим начальное условие:

$$u_{i_0,j_0}(s) = \varphi(s), \quad s\in[-1,0], \quad \varphi(s)\in S. \tag{4}$$

Для всех остальных нейронов считаем, что

$$u_{i,j}(s) = u_*, \quad s\in[-1,0]. \tag{5}$$

Будем считать, что по каким-либо причинам нейрон, расположенный в узле (i_0+1, j_0+1), окажется невосприимчивым к внешнему воздействию во время спайка нейрона (i_0, j_0).

По решетке начнет распространяться волна нейронной активности. Достигнув очередной ячейки, волна обойдет ее с двух сторон, захватывая по пути все новые и новые ячейки. На рисунке 1 показана схема прохождения волны по ячейке нейронной сети.

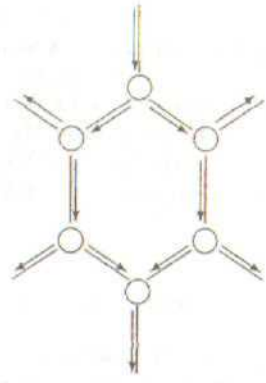

Рис. 1.

В момент времени $t = 5\tau + o(1)$ начнется спайк у нейрона $(i_0 + 1, j_0 + 1)$, который спустя время $\tau + o(1)$ вызовет новый спайк у нейрона (i_0, j_0). После этого по решетке будет распространяться вторичная волна, которая полностью повторит первичную (с точностью до сдвига по времени). Данный процесс будет повторяться с периодом $T = 6\tau + o(1)$. Каждая волна будет гаснуть, дойдя до края решетки. Направленность движения волны гарантирована наличием рефрактерного периода: каждый нейрон не реагирует на внешний сигнал во время спайка и некоторого промежутка времени после него. Описанный периодический режим называется ведущим центром.

Таким образом, справедливо следующее утверждение: рассмотрим клеточную нейронную сеть, динамика которой описывается системой (3). Пусть выполнено условие (2), начальные условия имеют вид (4) – (5), и по каким-либо причинам нейрон с номером (i_0+1, j_0+1) окажется невосприимчивым к внешнему воздействию во время спайка нейрона с номером (i_0, j_0), тогда все узлы решетки можно разбить на шесть попарно не пересекающихся множеств $G_l \quad (l = 1, \ldots, 6)$, при этом по сети будет распространяться волна нейронной активности, которая характеризуется следующими свойствами:

– нейроны одного множества генерируют спайки почти одновременно;

– вслед за нейронами множества G_l спайки генерируют нейроны из G_{l+1} (для множества G_6 следующим является множество G_1);

– временные рассогласования между спайками нейронов соседних множеств близки к τ;

– промежутки времени между последовательными спайками нейронов одного множества близки к 6τ.

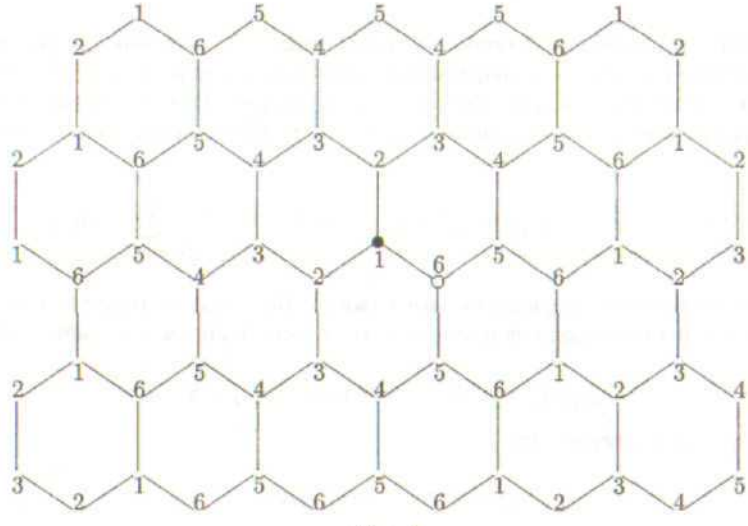

Рис. 2.

На рисунке 2 изображена схема нейронной сети, по которой распространяется волна описанной структуры. Цифра в каждом узле решетки соответствует номеру множества G_l, которому принадлежит данный

нейрон. Нейрон с номером (i_0, j_0) обозначен черным кружком, нейрон с номером (i_0+1, j_0+1) — светлым.

В статье описан лишь один из возможных периодических режимов. Многообразие аттракторов данной сети очень велико.

Список литературы

1. Кащенко С.А., Майоров В.В., Мышкин И.Ю. Исследование колебаний в кольцевых нейронных системах // Доклады РАН. 1993. Т. 333. №5. С. 594–597.

2. Кащенко С.А., Майоров В.В., Мячин М.Л. Колебания в системах уравнений с запаздыванием и разностной диффузией, моделирующих локальные нейронные сети // Доклады РАН. 1995. Т. 344. №3. С. 1274–1279.

3. Ануфриенко С.Е., Майоров В.В., Мышкин И.Ю., Громов С.А. Исследование системы уравнений с запаздыванием, моделирующей сальтаторное проведение возбуждения // Математическое моделирование. 2004. Т. 11. №1. С. 3–7.

4. Майоров В.В., Мышкин И.Ю. Математическое моделирование нейронной сети на основе уравнений с запаздыванием // Математическое моделирование. 1990. Т. 2. №11. С. 64–76.

5. Кащенко С.А., Майоров В.В. Исследование дифференциально-разностных уравнений, моделирующих импульсную активность нейрона // Математическое моделирование. 1993. Т. 5. №12. С. 13 - 25.

Модел. и анализ информ. систем. Т.11, №2 (2004) 8–17

УДК 681.3

Взаимодействующие раскрашивающие процессы

Кузьмин Е.В., Соколов В.А. [1]
Ярославский государственный университет
150 000, Ярославль, Советская, 14

получена 23 мая 2004

Аннотация

В данной работе представлен новый специальный фрагмент алгебры процессов, определяемой в стиле CSP Хоара и CCS Милнера, позволяющий строить формальные модели параллельных и распределённых систем, которые могут быть рассмотрены как независимые от данных помеченные системы переходов, а более конкретно, вполне структурированные системы переходов автоматного типа. Предложена к рассмотрению конкретная реализация данного класса систем переходов, новый формализм для моделирования распределённых систем, позволяющий отслеживать перемещение данных различного типа между компонентами системы, названный *взаимодействующие раскрашивающие процессы* (Communicating Colouring Processes – CCP). В рамках моделей CCP могут рассматриваться вопросы о доставке пакетов информации до адресата и недопущении передачи конфиденциальных данных в открытую небезопасную среду.

1. Введение

В настоящее время большое внимание уделяется разработке и исследованию формализмов для анализа поведенческих свойств и верификации распределённых систем, характеризуемых отсутствием централизованного управления работой системы.

Изучение природы вычислительных процессов, в том числе и параллельных, привело к появлению и широкому применению для этих целей алгебраических подходов [13, 8, 7, 14].

В алгебрах процессов, каковыми являются, например, CCS (Calculus of Communicating Systems – исчисление взаимодействующих систем) [14] и CSP (Communicating Sequential Processes – взаимодействующие последовательные процессы) [8], конструктивный подход играет крайне важную роль: процесс в них описывается указанием того, как он построен из меньших процессов.

В данной работе представлен фрагмент алгебры процессов, определяемой в стиле CSP Хоара и CCS Милнера, позволяющий строить формальные модели (параллельных и распределённых систем), которые могут быть рассмотрены как независимые от данных помеченные системы переходов, а более конкретно, вполне структурированные системы переходов с совместимостью по возрастанию и убыванию [6].

Более того, предлагается к рассмотрению конкретная реализация данного класса систем переходов, новый формализм для моделирования распределённых систем, позволяющий отслеживать перемещение данных различного типа между компонентами системы, названный *взаимодействующие раскрашивающие процессы* (Communicating Colouring Processes – CCP).

В рамках CCP возможно построение моделей распределённых систем, где поведение каждого компонента описывается последовательным процессом и между ними организовано взаимодействие, направленное на обмен и передачу пакетов информации. В данной модели мы игнорируем значения передаваемых данных и принимаем в расчёт только лишь тип (цвет) данных, определяемый, возможно, типом источника данных и уровнем секретности. Фокусируется внимание на возможности отслеживать перемещение данных различного типа между компонентами системы. В этом аспекте могут рассматриваться вопросы о доставке пакетов данных до адресата и о недопущении передачи конфиденциальных данных в открытую небезопасную среду.

Переход из одного допустимого состояния компонента распределённой системы в другое осуществляется посредством исполнения действия одного из следующих типов: отправка или приём сообщения в виде пакета данных из одного компонента в другой или же во внешнюю среду, формирование пакетов данных из других информационных пакетов (включая добавление и извлечение данных определённого типа).

Модель распределённой системы, реализованная в формализме CCP, представляет собой параллельную композицию взаимодействующих последовательных процессов, синхронизирующихся между собой

[1] Работа выполнена при поддержке РФФИ (грант 03-01-00804-а)

посредством взаимодействия через общие переменные (каналы). Каждое действие процесса – это либо передача, либо приём значения некоторого выражения над переменными. Результатом вычисления такого выражения является мультимножество над типами (цветами) данных. Совершение действия ССР приводит к изменению значения (раскраски) переменной. Поэтому работа (исполнение) процессов интерпретируется как раскрашивание множеств переменных. В терминах раскрашивания могут быть адекватно выражены, например, свойства систем, связанные с информационной безопасностью.

Конечные автоматы или конечные машины (Finite State Machines – FSM) широко используются для моделирования систем, где управляющий аспект является доминирующим. Конечные автоматы графически изображаются в виде помеченных ориентированных графов, где узлы представляют собой состояния, а дуги, помечающиеся метками действий, являются переходами по состояниям. Взаимодействующие автоматы [4] – это расширение конечных автоматов с помощью параллелизма и взаимодействия между параллельными компонентами. Параллелизм достигается организацией конечных автоматов в параллельную композицию, где все автоматы работают одновременно. Параллельные автоматы могут синхронизироваться посредством переходов, имеющих одинаковые метки. Но взаимодействующие автоматы используются только тогда, когда принимается в расчёт лишь факт взаимодействия компонентов, например, факт передачи пакета данных без описания его типа или содержимого.

С другой стороны, существует много формализмов, таких как машины Тьюринга и счётчиковые машины [15], сети Петри [16], системы с ненадёжными каналами (Lossy Channel Systems) [2], которые имеют семантическое расширение и производят манипуляции с данными. Модель в рамках этих формализмов состоит из системы переходов с конечным числом управляющих состояний (управляющая часть), работающая с потенциально бесконечным множеством D значений данных. Состояние такой системы может быть представлено в виде пары (q, d), где q – управляющее состояние, а $d \in D$. Переход в управляющей части имеет метку и операцию над множеством D. Переход возможен лишь в том случае, если будет истинен соответствующий этому переходу некоторый предикат над D (т.е. переход зависит как от управляющих состояний, так и от данных, с которыми производятся манипуляции).

Формализм ССР принимает во внимание факт передачи данных, а также позволяет отслеживать перемещение данных различного типа между компонентами системы. Но переход из одного состояния в другое не зависит от оперируемых данных, а определяется только управляющими состояниями. Таким образом, формализм ССР занимает промежуточное положение среди формализмов, указанных выше.

Статья организована следующим образом. Во втором разделе даются основные понятия и определения. Затем описывается фрагмент алгебры процессов, позволяющий строить формальные модели (параллельных и распределённых систем), которые рассматриваются как независимые от данных вполне структурированные системы переходов с совместимостью по возрастанию и убыванию. В четвёртом и пятом разделах приводится конкретная реализация взаимодействующих процессов независимых от данных – *взаимодействующие раскрашивающие процессы* (Communicating Colouring Processes – CCP). Доказывается, что любая модель ССР – это вполне структурированная система переходов с совместимостью по возрастанию и убыванию. В заключении рассматривается пример модели распределённой системы специального вида, выполненной с помощью формализма ССР.

2. Основные понятия и определения

Мультимножеством m над множеством X называется функция $m : X \to \mathbf{N}$, где $\mathbf{N}$ – множество неотрицательных целых чисел. Обозначим $\mathcal{M}(X)$ множество всех мультимножеств над X; $x \in m \iff m(x) \neq 0$. Размер мультимножества определяется как $|m| = \Sigma_{x \in X} m(x)$. Мультимножество конечно, если $m(x) = 0$ для всех $x \in X$ за исключением, быть может, конечного их числа. В дальнейшем будем рассматривать только конечные мультимножества.

Операции и отношения теории множеств естественным образом расширяются на конечные мультимножества. Пусть $m_1, m_2 \in \mathcal{M}(X)$, $x \in X$, $X' \subseteq X$ и $k \in \mathbf{N}$, тогда

$$(m_1 + m_2)(x) = m_1(x) + m_2(x), \quad (k \cdot m_1)(x) = k \cdot m_1(x),$$

$$(m_1 \ominus m_2)(x) = m_1(x) \ominus m_2(x) \stackrel{\text{def}}{=} max(0, m_1(x) - m_2(x)),$$

$$(m_1 \cup m_2)(x) = max(m_1(x), m_2(x)),$$

$$(m_1 \cap m_2)(x) = min(m_1(x), m_2(x),)$$

$$m_1 \subseteq m_2 \iff \forall x \in X : m_1(x) \leq m_2(x),$$

$$(m_1 \backslash X')(x) = \begin{cases} m_1(x), & \forall x \notin X' \\ 0, & \forall x \in X' \end{cases}, \qquad (m_1 / X')(x) = \begin{cases} 0, & \forall x \notin X' \\ m_1(x), & \forall x \in X' \end{cases}$$

Бинарное отношение $\leq$ называется отношением *частичного порядка*, если оно рефлексивно, транзитивно и антисимметрично. Если отношение только рефлексивно и транзитивно, то оно называется отношением *квазипорядка*.

Квазипорядок $\leq$ на множестве X называется *вполне упорядочиваемым* (well-quasi-ordering), если для любой бесконечной последовательности $x_0, x_1, x_2, \ldots$ элементов из X существуют индексы $i < j$ такие, что $x_i \leq x_j$.

Пусть $\leq$ – вполне упорядочиваемый квазипорядок на множестве X. *Идеалом или верхним конусом* (в X) называется подмножество $I \subseteq X$, такое что для $x \in I$, $y \in X$ и $x \leq y$, следует $y \in I$. Идеал может быть получен замыканием кверху некоторого множества. Каждый элемент $x \in X$ порождает верхний конус $\uparrow x \stackrel{\text{def}}{=} \{y \mid y \geq x\}$. *Базисом* верхнего конуса I называется множество $min(I)$ такое, что $I = \cup_{x \in min(I)} \uparrow x$. Известно, что если $\leq$ – вполне упорядочиваемый квазипорядок на множестве X, то всякий верхний конус I имеет конечный базис (док. см., напр., в [6]).

Система помеченных переходов есть четвёрка $LTS = (S, T, \rightarrow, s_0)$, где S есть множество состояний с элементами $s_0, s_1, s_2, \ldots$, T – некоторый алфавит пометок (множество имён действий), $\rightarrow \subseteq (S \times T \times S)$ – отношение перехода между состояниями, $s_0 \in S$ – начальное состояние системы.

Переход (s, t, s') обычно обозначается как $s \stackrel{t}{\rightarrow} s'$, что означает, что действие с именем t переводит состояние s в состояние s'. Через $Succ(s)$ обозначается множество последующих состояний для s, через $Pred(s)$ – множество его предыдущих состояний. Система LTS будет конечно ветвящейся, если для любого s множество $Succ(s)$ конечно. В данной работе рассматриваются системы с конечным ветвлением.

Последовательное исполнение для LTS есть конечная или бесконечная цепочка переходов $s_0 \stackrel{t_1}{\rightarrow} s_1 \stackrel{t_2}{\rightarrow} s_2 \rightarrow \ldots$, где s_0 – начальное состояние системы.

Системы помеченных переходов – одна из наиболее распространённых моделей для описания поведения систем. Для решения задач анализа семантических свойств систем переходов полезной оказывается теория вполне структурированных систем переходов [1, 6].

Вполне структурированной системой переходов с совместимостью по возрастанию (соот. по убыванию) WTS называется система переходов $LTS = (S, T, \rightarrow, s_0)$, дополненная отношением квазипорядка $\leq \subseteq S \times S$, удовлетворяющая следующим условиям: отношение $\leq$ является вполне упорядочиваемым квазипорядком; квазипорядок $\leq$ совместим по возрастанию с отношением переходов $\rightarrow$, а именно, для любых состояний $s_1 \leq s_2$ и перехода $s_1 \stackrel{t}{\rightarrow} s_1'$ существует переход $s_2 \stackrel{t}{\rightarrow} s_2'$, такой что $s_1' \precsim s_2'$ (соот. квазипорядок $\leq$ совместим по убыванию с отношением переходов $\rightarrow$, а именно, для любых состояний $s_1 \leq s_2$ и перехода $s_2 \stackrel{t}{\rightarrow} s_2'$ существует переход $s_1 \stackrel{t}{\rightarrow} s_1'$, такой что $s_1' \leq s_2'$).

Система переходов имеет эффективный предбазис, если для каждого состояния $s \in S$ и пометки $t \in T$ вычислимо множество $min(Pred_t(\uparrow s))$.

Система переходов является эффективной по пересечению, если для любых состояний s, s' вычислимо множество $min(\uparrow s \cap \uparrow s')$.

3. Независимые от данных взаимодействующие процессы

В данном разделе описывается фрагмент алгебры процессов, определяемой в стиле CSP Хоара и CCS Милнера, позволяющий строить формальные модели (параллельных и распределённых систем), которые могут быть рассмотрены как независимые от данных помеченные системы переходов, а более конкретно, вполне структурированные системы переходов с совместимостью по возрастанию и убыванию.

Предполагается наличие следующих множеств.

- $Pr = \{X, Y, Z, X_1, \ldots\}$ – множество имён процессов;
- $Var = \{x, y, z, x_1, \ldots\}$ – конечное множество переменных;
- $Chl = \{v, v_1, v_2, \ldots\}$ – конечное множество каналов;
- $Exp = \{e, e_1, e_2, \ldots\}$ – конечное множество выражений над множеством переменных Var;

- $Act_{Var} = \{x!e, \ldots, x?y, \ldots\}$ – конечное множество простых действий процессов;
- $Act_{Chl} = \{(v!e, v?x), (v_1!e, v_1?x), \ldots)\}$ – множество взаимодействий между процессами (передача данных через каналы $v, v_1, \ldots$);
- $D = \{d, d_1, d_2, \ldots\}$ – бесконечное множество допустимых значений переменных.

Приводимое ниже определение описывает абстрактный синтаксис выражений (для) процессов. Выражения процессов $E, E', \ldots$ задаются следующей грамматикой:

$$E, E' ::= 0 \mid X \in Pr \mid v?x.E \mid v!e.E \mid E + E' \mid E \parallel E'$$

Дадим теперь некоторое интуитивное представление смысла приведённых выше конструкций.

- 0 представляет процесс, который не является активным, т.е. никакое действие не может быть выполнено.
- Процесс $v?x.E$ совершает действие $v?x$, а затем продолжается как процесс E. Действие $v?x$ означает копирование (присвоение) в переменную x значения из v, где в данном случае v может быть как обычной переменной, так и синхронизирующим каналом передачи данных, т.е. $v \in Var \cup Chl$.
- Аналогично, после совершения действия $v!e$ процесс $v!e.E$ становится процессом E. Действие $v!e$ представляет собой копирование в v значения выражения e, где в данном случае $v \in Var \cup Chl$. Вместо краткой записи выражения e будет также использоваться и полная форма вида $e(x_1, \ldots, x_n)$.
- Для двух данных процессов E и E' оператор суммы даёт процесс $E + E'$, который ведёт себя либо как E, либо как E'.
- Для двух данных процессов E и E' параллельная композиция даёт процесс $E \parallel E'$, способный выполнять действия E и E' параллельно, т.е. либо произвольно чередуя их, либо синхронизируя работу (в случае передачи данных по каналу).

Пассивный процесс 0 может рассматриваться как пустая сумма или же как пустая параллельная композиция.

Процессом является выражение процесса, переменные которого определены конечным набором рекурсивных уравнений (процессов).

Декларацией процесса называется совокупность Δ рекурсивных уравнений процессов

$$\Delta = \{X_i \stackrel{\text{def}}{=} E_i \mid 1 \le i \le n\},$$

где n – натуральное число, переменные X_i все различны, и где выражения E_i используют только переменные, принадлежащие множеству $Pr(\Delta) \stackrel{\text{def}}{=} \{X_1, X_2, \ldots, X_n\}$. Более того, если выражение E_i строится с помощью оператора параллельной композиции $||$, запретим использование в этом выражении переменной X_i.

Переменная X_1 называется *головной переменной* и служит в качестве начального состояния процесса, если нет других уточнений.

Как это принято в алгебре процессов, определим поведение процессов посредством *структурной операционной семантики*, данной в качестве системы переходов.

Каждая декларация процессов Δ задаёт систему помеченных переходов: управляющими состояниями в ней являются выражения процессов (в данной семантике их конечное число), построенные на множестве $Pr(\Delta)$, множество меток задаётся как $Act = Act_{Var} \cup Act_{Chl}$, и отношение переходов определяется с помощью следующих правил вывода.

$$\alpha.E \xrightarrow{\alpha} E$$

$$\frac{E \xrightarrow{\alpha} E'}{X \xrightarrow{\alpha} E'} \; X \stackrel{\text{def}}{=} E \in \Delta$$

$$\frac{E_j \xrightarrow{\alpha} E'}{\sum\{E_i : i \in I\} \xrightarrow{\alpha} E'} \; j \in I$$

$$\frac{E \xrightarrow{\alpha} E'}{E \parallel F \xrightarrow{\alpha} E' \parallel F} \; \alpha \in Act_{Var}$$

$$\frac{E \xrightarrow{v!e} E' \quad F \xrightarrow{v?x} F'}{E \parallel F \xrightarrow{(v!e,v?x)} E' \parallel F'} \; v \in Chl, x \in Var, e \in Exp$$

Последнее правило означает одновременное взаимодействие между процессами, связанное с передачей значения выражения e в переменную x через канал v, т.е. после срабатывания перехода с пометкой $\alpha = (v!e, v?x)$ переменная x становится равной значению выражения e. Если есть процесс вида $v!e.E$ (или соот. $v?x.E$), где $v \in Chl$, $e \in Expr$, он может перейти в процесс E тогда и только тогда, когда в параллельной композиции будет процесс $v?x.F$ (соот. $v!e.F$), который при одновременном срабатывании перейдёт в F.

Необходимо отметить, что в случае обычных переменных $x, y \in Var$ процессы $x!y.E$ и $y?x.E$ будут по сути совершать одно и то же действие, а именно, присвоение переменной x значения переменной y.

Рассмотрим систему помеченных переходов, порождаемую декларацией процессов Δ, как вполне структурированную систему переходов с совместимостью по возрастанию и убыванию $\mathcal{S} = (S, T, \rightarrow, s_0)$, где $T = Act$. Для этого необходимо на Δ наложить дополнительные ограничения. Положим, что на множестве D задан вполне упорядочиваемый квазипорядок $\leq$. Далее, будем рассматривать только те выражения $e(x_1, \ldots, x_n) \in Exp$, которые представляют собой монотонную функцию по своим переменным относительно вполне упорядочиваемого квазипорядка $\leq$, т.е. для любых двух векторов значений переменных $(d_1, \ldots, d_n) \leq (d'_1, \ldots, d'_n)$ должно выполняться $e(d_1, \ldots, d_n) \leq e(d'_1, \ldots, d'_n)$.

Тогда состояние s системы $\mathcal{S}$ будет представлять собой пару $(q, \bar{d})$, где $q \in Q$ – это выражение процессов, а $\bar{d} = (d_1, \ldots, d_k)$ – вектор значений всех переменных (при условии, что переменные некоторым образом упорядочены), $k = |Var|$. Начальное состояние s_0 есть $(q_0, \bar{d}_0)$, где q_0 соответствует выражению процесса X_1, а $\bar{d}_0$ – вектор начальных значений переменных. Из определения декларации процессов Δ следует, что множество Q конечно. Но множество S системы $\mathcal{S}$ бесконечное, т.к. векторов значений переменных может быть бесконечно много. Более того, на множестве S можно задать вполне упорядочиваемый квазипорядок $\leq_S$ так, что $s{=}(q, \bar{d}) \leq_S s'{=}(q', \bar{d}') \iff q = q'$ и $\bar{d} \leq \bar{d}'$.

Покажем, что отношение $\leq_S$ совместимо по возрастанию и по убыванию с отношением переходов $\rightarrow$.

Из монотонности функций $e \in Exp$ и независимости отношения переходов $\rightarrow$ от данных (недетерминизм переходов) следует, что для любых двух состояний

$$(q_1, d_1, \ldots, d_k) \leq_S (q_1, d'_1, \ldots, d'_k)$$

и перехода

$$(q_1, d_1, \ldots, d_k) \xrightarrow{\alpha} (q_2, d_1, \ldots, d''_i, \ldots, d_k),$$

где $d''_i = e(d_1, \ldots, d_k)$, выражение e (как и индекс i) определяется переходом $\alpha \in Act$, существует переход

$$(q_1, d'_1, \ldots, d'_k) \xrightarrow{\alpha} (q_2, d'_1, \ldots, d'''_i, \ldots, d'_k),$$

где $d'''_i = e(d'_1, \ldots, d'_k)$, причём

$$(q_2, d_1, \ldots, d''_i, \ldots, d_k) \leq_S (q_2, d'_1, \ldots, d'''_i, \ldots, d'_k),$$

т.к. $\forall j\, (1 \leq j \leq k) : d_j \leq d'_j$, и $d''_i{=}e(d_1, \ldots, d_k) \leq d'''_i{=}e(d'_1, \ldots, d'_k)$.

Аналогично, по тем же соображениям, для любых двух состояний

$$(q_1, d_1, \ldots, d_k) \leq_S (q_1, d'_1, \ldots, d'_k)$$

и перехода

$$(q_1, d'_1, \ldots, d'_k) \xrightarrow{\alpha} (q_2, d'_1, \ldots, d'''_i, \ldots, d'_k),$$

где $d'''_i = e(d'_1, \ldots, d'_k)$, $\alpha \in Act$, существует переход

$$(q_1, d_1, \ldots, d_k) \xrightarrow{\alpha} (q_2, d_1, \ldots, d''_i, \ldots, d_k),$$

где $d''_i = e(d_1, \ldots, d_k)$, причём

$$(q_2, d_1, \ldots, d''_i, \ldots, d_k) \leq_S (q_2, d'_1, \ldots, d'''_i, \ldots, d'_k),$$

т.к. $\forall j\, (1 \leq j \leq k) : d_j \leq d'_j$, и $d''_i \leq d'''_i$.

4. Взаимодействующие раскрашивающие процессы

Рассмотрим конкретную реализацию взаимодействующих процессов независимых от данных – *взаимодействующие раскрашивающие процессы* (Communicating Colouring Processes – CCP).

Для этого определим новое множество Σ типов (или цветов) данных. Опишем тип переменных из Var как мультимножество над Σ. Без нарушения условия монотонности зададим допустимые для данного формализма выражения $e(x_1, \ldots, x_n)$. Итак, выражение $e(x_1, \ldots, x_n)$ в рамках CCP может быть одного из следующих видов:

- $e_+(x_1, x_2) = x_1 + x_2$;
- $e_\cup(x_1, x_2) = x_1 \cup x_2$;
- $e_\ominus(x_1) = x_1 \ominus m$, где $m \in \mathcal{M}(\Sigma)$;
- $e_\cap(x_1) = x_1 \cap m$, где $m \in \mathcal{M}(\Sigma)$;
- $e_\setminus(x_1) = x_1 \setminus C$, где $C \subseteq \Sigma$;
- $e_/(x_1) = x_1 / C$, где $C \subseteq \Sigma$;

Выражения вида $e_+(y, z) = y + z$ или $e_\cup(y, z) = y \cup z$ моделируют формирование (построение) нового пакета данных из данных других пакетов, находящихся в переменных y и z, который, например, с помощью действия $x!e$ помещается в переменную x.

Выражения $e_\setminus(x) = x \setminus C$, $e_/(x) = x/C$ или $e_\ominus(x) = x \ominus m$ в сочетании с действием $x!e(x)$ могут интерпретироваться как извлечение данных из пакета переменной x.

Выражения $v!e(x_1, x_2)$ и $v?x$ означают отправку и приём пакета данных через канал v соответственно.

Оператор $\cap$ может быть использован для формирования пакета данных определённого формата. Например, последовательность действий $z!(x \cap m)$, $x!(x \ominus m)$, $v!z$ означает передачу пакета данных формата m через канал v; оператор $\cap$ ограничивает размер передаваемых за один раз данных.

Таким образом, *состояние* CCP – это набор $(q, m_1, \ldots, m_n)$, где q – локальное состояние (определяемое декларацией процессов Δ), представляющее собой выражение процессов, а $m_1, \ldots, m_n$ – мультимножества, являющиеся содержимым соответствующих переменных.

Раскраска (coloring) CCP – это отображение $M : Var \to \mathcal{M}(\Sigma)$. Определим на множестве переменных Var некоторый порядок, т.е. $Var = (x_1, \ldots, x_n)$, тогда раскраска M – это вектор из мультимножеств $M = (m_1, \ldots, m_n)$, где $\forall i \ (1 \le i \le n)$ $m_i = M(x_i)$, $n = |Var|$. Для некоторого подмножества $Var' = \{x_{i_1}, \ldots, x_{i_k}\}$, $Var' \subseteq Var$, обозначим $M(Var')$ множество раскрасок (разметок) $\{M(x_{i_1}), \ldots, M(x_{i_k})\}$. Тогда состояние CCP A может быть представлено как (q, M).

Исполнение CCP – это последовательность состояний, начинающаяся в начальном состоянии (q_0, M_0), где q_0 представляет собой начальное выражение процесса X_1, и строящаяся в соответствии с правилами переходов.

Каждое срабатывание перехода $q \xrightarrow{t} q'$ переводит CCP из состояния (q, M) в некоторое последующее состояние (q', M'), где M' вычисляется в соответствии с выражением, ассоциированным с меткой действия перехода t.

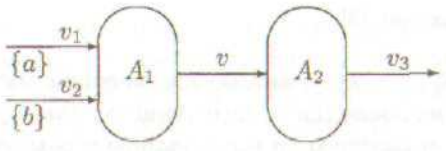

Рис. 1. Схема взаимодействия компонентов A_1 и A_2

Рассмотрим CCP A, представленный на рис. 2 в виде параллельно работающих автоматов (*взаимодействующих раскрашивающих автоматов*), где $\Sigma = \{a, b\}$, $Chl = \{v\}$, $Var = \{v_1, v_2, v_3\} \cup \{x, y, z\}$, v_1, v_2, v_3 представляют собой каналы, через которые происходит взаимодействие с внешней средой, а через канал v происходит взаимодействие между процессами A_1 и A_2, $Act = \{v_1?y, v_2?y, x!(x + y), (v!x, v?z), v_3!(z \setminus \{a\})\}$. Компонент A_1 принимает и накапливает данные двух типов a и b из каналов v_1 и v_2; затем A_1 отправляет данные в фильтр A_2, который не пропускает данные типа a во внешнюю среду (рис. 1).

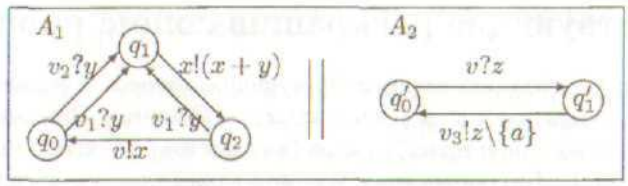

Рис. 2. Пример взаимодействующих раскрашивающих процессов $A = A_1 \| A_2$

$$A = A_1 \parallel A_2$$

$$A_1 = v_1?y\,.\,A_1' + v_2?y\,.\,A_1', \quad A_1' = x!(x+y)\,.\,(v!x\,.\,A_1 + v_1?y\,.\,A_1'),$$

$$A_2 = v?z\,.\,v_3!(z\backslash\{a\})\,.\,A_2$$

Последовательность переходов $(q_0, q_0') \xrightarrow{v_2?y} (q_1, q_0') \xrightarrow{x!(x+y)} (q_2, q_0') \xrightarrow{v_1?y} (q_1, q_0') \xrightarrow{x!(x+y)} (q_2, q_0') \xrightarrow{(v!x, v?z)} (q_0, q_1') \xrightarrow{v_3!z\backslash\{a\}} (q_0, q_0')$ из начального состояния (q_0, q_0') с начальной раскраской $(\{a\}, \{b\}, \emptyset, \emptyset, \emptyset, \emptyset)$ производит следующую последовательность раскрасок:

$$(\{a\},\{b\},\emptyset,\emptyset,\emptyset,\emptyset) \to (\{a\},\{b\},\emptyset,\emptyset,\{b\},\emptyset) \to (\{a\},\{b\},\emptyset,\{b\},\{b\},\emptyset) \to (\{a\},\{b\},\emptyset,\{b\},\{a\},\emptyset) \to$$

$$(\{a\},\{b\},\emptyset,\{a,b\},\{a\},\emptyset) \to (\{a\},\{b\},\emptyset,\{a,b\},\{a\},\{a,b\}) \to (\{a\},\{b\},\{b\},\{a,b\},\{a\},\{a,b\}).$$

5. CCP – вполне структурированная система переходов

В зависимости от используемых операторов взаимодействующие раскрашивающие процессы можно рассматривать как системы переходов с конечным или бесконечным числом состояний. Во втором случае CCP является вполне структурированной системой переходов, что влечёт разрешимость ряда классических проблем для этого формализма. Вполне структурированные системы переходов – это весьма широкий класс систем переходов с бесконечным числом состояний, для которых разрешимость многих свойств следует из существования совместимого с отношением переходов вполне упорядочиваемого квазипорядка на множестве состояний.

Рассмотрим CCP как систему помеченных переходов $\mathcal{S} = (S, T, \to, s_0)$. Определим естественным образом частичный порядок $\preceq$ на множестве состояний S системы $\mathcal{S}$.

Определение 1 *Пусть (q, M), (q', M') – состояния CCP. Положим $(q, M) \preceq (q', M')$ тогда и только тогда, когда $q = q'$ и $M \preceq M'$, где $M \preceq M' \iff \forall x \in Var : M(x) \subseteq M'(x)$.*

Утверждение 1 *Отношение порядка $\preceq$ на множестве состояний S является вполне упорядочиваемым квазипорядком.*

Доказательство. По лемме Диксона [5]. □

Утверждение 2 *Пусть $\mathcal{S} = (S, T, \to, s_0)$ – взаимодействующие раскрашивающие процессы, $\preceq$ – вполне упорядочиваемый квазипорядок на множестве состояний S. Тогда $(\mathcal{S}, \preceq)$ есть вполне структурированная система переходов с совместимостью по возрастанию и убыванию.*

Доказательство. Пусть (q, M), (q', M') – состояния CCP. По определению 1 $(q, M) \preceq (q', M')$ тогда и только тогда, когда $q = q'$ and $M \preceq M'$. Отсюда, для всех состояний $(q, M) \preceq (q', M')$: $(q, M) \xrightarrow{t} (q_2, M_2) \iff (q', M') \xrightarrow{t} (q_2, M_2')$.

Легко проверить, что для всех операторов, которые используются в выражениях $e(x_1, x_2) \in Exp$, выполняются следующие условия:
$\forall (m_1, m_2) \preceq (m_1', m_2') : op(m_1, m_2) \preceq op(m_1', m_2'),\ op \in \{\cup, +\}$;
$\forall m_1 \preceq m_1'$ и $m_2 : op(m_1, m_2) \preceq op(m_1', m_2),\ op \in \{\cap, \ominus\}$;
$\forall m_1 \preceq m_1'$ и $C \subseteq \Sigma : op(m_1, C) \preceq op(m_1', C),\ op \in \{\backslash, /\}$, m_i, m_i' – мультимножества.

Следовательно, для всех состояний $(q, M) \preceq (q', M')$ и перехода $q \xrightarrow{t} q_2$ имеем $(q_2, M_2) \preceq (q_2, M_2')$, где M_2 и M_2' вычисляются в соответствии с выражением $e(x_1, x_2)$, которое используется в действии t, из M и M' соответственно, т.е. выполняется условие монотонности выражений $e(x_1, x_2)$ использующихся в ССР. □

Утверждение 3 *Пусть система переходов $(\mathcal{S}, \preceq)$ – это ССР с вполне упорядочиваемым квазипорядком на множестве состояний. Тогда для каждого состояния s и действия t множество $min(Pred_t(\uparrow s))$ вычислимо.*

Доказательство. Построим множество ${}^\circ t$ минимальных состояний, при которых переход с пометкой t может сработать. Для любого локального состояния q, из которого существует переход t, состояние $(q, \emptyset, \ldots, \emptyset)$ принадлежит множеству ${}^\circ t$, т.е. ${}^\circ t = \{(q, \emptyset, \ldots, \emptyset) \mid \exists q' : q \xrightarrow{t} q'\}$.

Обозначим через t° множество состояний, получаемых в результате срабатывания перехода t на состояниях из множества ${}^\circ t$.

Рассмотрим множество $\uparrow s \cap \uparrow t^\circ$, состоящее из таких состояний s', что s' может быть получена в результате срабатывания перехода t и $s \leq s'$. Это множество является направленным вверх конусом и в силу вполне упорядочиваемости отношения $\leq$ имеет конечный базис, который обозначим $lub(s, t^\circ)$.

Конечный базис $lub(s, t^\circ)$ состоит из одного состояния и может быть эффективно построен следующим образом. Рассмотрим состояния $(q, m_1, \ldots, m_n) \in t^\circ$, $s = (q', m_1', \ldots, m_n')$. Для того, чтобы пересечение конусов $\uparrow s \cap \uparrow t^\circ$ не было пустым, необходимо выполнение условия $q = q'$. Тогда $lub(s, t^\circ) = (q, m_1' \cup m_1, \ldots, m_n' \cup m_n)$.

Пусть теперь некоторое состояние $s' \in Pred_t(\uparrow s)$. Тогда возможно срабатывание перехода t, которое переводит s' в состояние $s'' \in \uparrow s$. Отсюда следует, что $s'' \in \uparrow t^\circ$. С другой стороны, для любого состояния s'' такого, что $s'' \in \uparrow s$ и $s'' \in \uparrow t^\circ$ можно построить $Pred_t(s'')$, т.к. каждое действие t изменяет значение только одной переменной с помощью простого выражения $e \in Exp$. Получаем $Pred_t(\uparrow s) = Pred_t(\uparrow lub(s, t^\circ)) = \uparrow Pred_t(lub(s, t^\circ))$.

Эта формула задаёт эффективное построение конечного базиса множества $Pred_t(\uparrow s)$ по конечному базису $lub(s, t^\circ)$. □

Мы получили разрешимость следующих проблем для ССР [1, 6]:

- *проблема покрытия*, которая состоит в том, чтобы для состояний (q, M) и (q', M') определить, возможно ли, начиная исполнение из (q, M), покрыть состояние (q', M'), т.е. может ли быть достигнуто состояние $(q'', M'') \succeq (q', M')$;

- *проблема субпокрытия*, состоящая в том, чтобы для состояний (q, M) и (q', M') определить, возможно ли, начиная исполнение из (q, M), попасть в состояние, которое покрывается состоянием (q', M'), т.е. может ли быть достигнуто состояние $(q'', M'') \preceq (q', M')$;

- *проблема достижимости*, т.е. определить, достижимо ли из состояния s данное множество Q управляющих (локальных) состояний;

- *проблема неизбежности*, т.е. по данному множеству Q локальных состояний и начальному состоянию s_0 определить, верно ли, что всякое исполнение приводит к некоторому локальному состоянию из Q (частным случаем является *проблема останова*: определить, верно ли, что всякое исполнение завершается, т.е. приводит к некоторому финальному состоянию).

Следовательно, например, имеем разрешимость проблемы передачи данных определённого цвета во внешнюю среду через некоторый канал.

Большое количество формализмов, таких как сети Петри [16], Lossy Channel Systems [2], Basic Parallel Process, Real-Time Automata [3] и мн. др., могут быть рассмотрены как вполне структурированные системы переходов или вполне структурированные системы переходов с совместимостью по убыванию. Формализм ССР – это вполне структурированная система переходов, обладающая свойствами совместимости и по возрастанию, и по убыванию. Это влечёт разрешимость некоторых новых проблем для ССР. Например, задачи проверки модели для различных подмножеств темпоральных логик при интерпретации элементарных высказываний верхними и нижними конусами состояний.

6. Пример

Пример (рис. 3) демонстрирует, помимо возможности моделирования взаимодействия между сотрудниками, секретарём и начальником, возможность отслеживания перемещения данных различных типов.

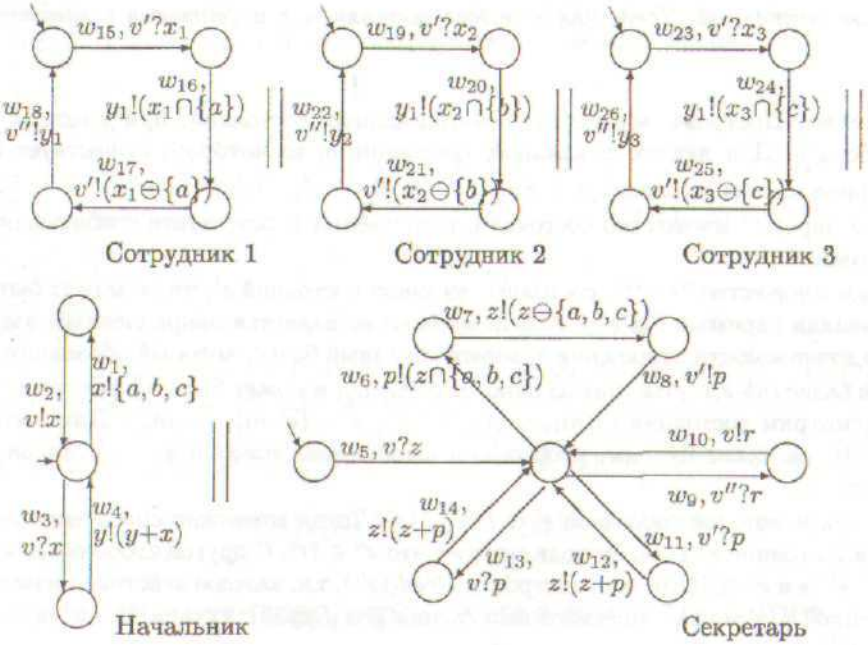

Рис. 3. Пример взаимодействующих раскрашивающих процессов (в виде взаимодействующих раскрашивающих автоматов)

Действия начальника:
w_1 – сформировать пакет заданий для сотрудников (a, b, c – типы заданий);
w_2 – отдать задания секретарю;
w_3, w_4 – получить решённое задание;

Действия секретаря:
w_5 – получить от начальника пакет заданий;
w_6, w_7, w_8 – отдать сотруднику пакет, состоящий не более чем из трёх различных по типу заданий;
w_9 – получить от сотрудника решение;
w_{10} – отдать решение начальнику;
w_{11}, w_{12} – получить от сотрудника оставшиеся задания;
w_{13}, w_{14} – получить от начальника новые задания;

Действия первого сотрудника:
w_{15} – получить пакет заданий;
w_{16} – взять из пакета своё задание (типа a);
w_{17} – отдать пакет с оставшимися заданиями (либо другому сотруднику, либо секретарю);
w_{18} – отдать секретарю решённое задание;

Каналы:
v – используется для обмена данными между начальником и секретарём;
v' – используется для обмена заданиями между сотрудниками и секретарём;
v'' – используется для передачи решённых задач от сотрудников к секретарю;

Список литературы

1. Abdulla P.A., Čerāns K., Jonsson B., Yih-Kuen T. General decidability theorems for infinite-state systems // Proc. 11th IEEE Symp. Logic in Computer Science (LICS'96), 1996. P. 313-321.
2. Abdulla P., Jonsson B. Verifying Programs with Unreliable Channels // Proc. LICS'93, 1993. P. 160-170.
3. Alur R., Courcoubetis C., Dill D. Model-checking for real-time systems // Proc. 5^{th} IEEE Int. Symp. on Logic in Computer Science, Philadelphia, 1990. P. 414-425.
4. Alur R., Kannan S., Yannakakis M.: Communicating hierarchical automata // ICALP'99, Springer LNCS 1644, 1999. P. 169-178.
5. Dickson L. E. Finiteness of the odd perfect and primitive abundant numbers with r distinct prime factors // Amer. Journal Math. 1913. 35. P. 413-422.
6. Finkel A., Schnoebelen Ph. Well-structured transition systems every-where! // Theoretical Computer Science, 256(1-2), 2001. P. 63-92.
7. Hennessy M. Algebraic Theory of Processes. MIT Press, 1988.
8. Hoare C.A.R. Communicating sequential processes. Prentice-Hall, 1985.
9. Hopcroft J.E., Ullman J.D. Introduction to Automata Theory, Languages and Computation. Addison Wesley, 1979.
10. Jensen K. Coloured Petri Nets // Vol.1. Eatcs Monographs on TCS, Springer-Verlag, 1994.
11. Kouzmin E., Sokolov V. Communicating Colouring Automata // Proc. Int. Workshop on Program Understanding (sat. of PSI'03), 2003. P. 40-46.
12. Mayr R. Lossy counter machines. Tech. Report TUM-I9827, Institut für Informatik, TUM, Munich, Germany, October 1998.
13. Milner R. A Calculus of Communicating Systems // LNCS 92, Springer-Verlag, 1980.
14. Milner R. Communication and Concurrency. Prentice Hall Int., 1989.
15. Minsky M. Computation: Finite and Infinite Machines. Prentice-Hall, 1967.
16. Peterson J. L. Petri Net Theory and the Modeling of Systems. Prentice-Hall Int., 1981.

Модел. и анализ информ. систем. Т.11, №2 (2004) 18–21

УДК 517.5

Приближение кусочно-гладких функций в пространствах L_p $(0 < p < 1)$

Морозов А.Н.
Ярославский государственный университет
150000, Ярославль, Советская, 14

получена 3 сентября 2004

Аннотация

В статье развиты результаты о наилучших кусочно-полиномиальных приближениях в метриках пространств L_p. Показано, что в случае $0 < p < 1$, хорошее качество приближения функций уже не полностью зависит от их гладкости.

1. Основные обозначения

Пусть $L_p[I]$ обозначает пространство действительных функций, интегрируемых в степени p по Лебегу на замкнутом слева полуинтервале I (равносильно интервале или отрезке), с $0 < p < \infty$. При всех таких p для удобства полагаем

$$\|f\|_{L_p[I]} = \Big(\int\limits_I |f(t)|^p dt\Big)^{\frac{1}{p}}.$$

Когда неясность возникнуть не может, сокращаем обозначение до $\|f\|_p$.

$$L_\infty[I] = \Big\{ f \ : ess\sup_{x\in I} |f(x)| < \infty \Big\}, \quad \|f\|_{L_\infty[I]} = ess\sup_{x\in I} |f(x)|.$$

Также используются пространства ($k \in N$, $p \geq 1$)

$$W_p^k[I] = \Big\{ f : f^{(k-1)} \text{ абсолютно непрерывна на отрезке } I, f^{(k)} \in L_p[I] \Big\}.$$

Определим для $f \in L_p[I]$

$$E_k(f;I)_p = inf\{ \|f - \pi_k\|_{L_p[I]} : \ \pi_k \in P_k \}$$

- наилучшее приближение алгебраическими многочленами степени не выше $k-1$ (порядка k) в $L_p[I]$.
Пусть $a = t_o < t_1 < \cdots < t_m = b$, $m \in N$. Набор τ полуинтервалов $\{[t_{j-1}, t_j)\}_{j=1}^m$ назовем разбиением полуинтервала $I = [a, b)$.

$$\lambda(\tau) = \max_{1\leq j\leq m} |t_j - t_{j-1}| \ \text{ - мелкость разбиения } \tau.$$

Через u_m будем обозначать разбиение полуинтервала I на m равных по длине полуинтервалов.
Положим

$$U_m^k(f;I)_p = \Big(\sum_{J\in u_m} E_k(f;J)_p^p\Big)^{\frac{1}{p}}$$

- наилучшее приближение кусочно-полиномиальными функциями степени не выше $k-1$, подчиненными равномерному разбиению u_m полуинтервала I , т.е. на каждом полуинтервале J из u_m приближающая функция - многочлен порядка k.
Пусть $|I|$ далее обозначает длину полуинтервала I.
В статьях [1]-[2], кроме прочего, показано, что для $f \in W_p^k[I]$ выполняется:

$$\lim_{m\to\infty} m^k\, U_m^k(f,I)_p = c_{k,p}\, |I|^k\, \|f^{(k)}\|_{L_p[I]},$$

где $c_{k,p} = E_k(\frac{t^k}{k!}, [0,1])_p$.
В статье [3] - что если $f \in W_1^k[I]$, то при всех $0 < p < 1$

$$\lim_{m\to\infty} m^k\, U_m^k(f,I)_p = c_{k,p}\,|I|^k\,\|f^{(k)}\|_{L_p[I]}.$$

В данной статье доказывается, что такое соотношение выполняется для функций из более широкого пространства. Вместо обычной k-той производной для них появляется некоторая "обобщенная".

2. Качество приближения кусочно дифференцируемых функций

При $0 < p < 1$ с порядком $O(m^k)$ могут приближаться фушкции, существенно отличающиеся от функций из W_1^k, причем, при $p < \frac{1}{k}$ - и разрывные функции.

Скажем, что функция f кусочно принадлежит пространству $W_1^k[a,b]$, если при некотором наборе точек $a = t_o < t_1 < \cdots < t_n = b$, $n \in N$ эта фушкция принадлежит каждому пространству $W_1^k[t_{i-1}, t_i]$ для $1 \le i \le n$; точнее, попадает в каждое из таких пространств после изменения ее значений не более, чем в двух крайних точках отрезка $[t_{i-1}, t_i]$.
Для упрощения формул будем считать, что $W_\infty^0 = L_\infty$.

Теорема 1. Пусть $k \in N$, $0 < p < 1$, а функция f, кусочно принадлежащая $W_1^k[I]$, находится в пространстве $W_\infty^l[I]$. Тогда при $\frac{1}{p} + l > k$

$$\lim_{m\to\infty} m^k\, U_m^k(f,I)_p = c_{k,p}\,|I|^k\,\|D^k f\|_{L_p[I]},$$

здесь через $D^k f$ обозначена такая функция, что если $f \in W_1^k[J]$, $J \subset I$, то $D^k f_{|J} = f^{(k)}{}_{|J}$.

Доказательство.
Заметим, что

$$m^k\, U_m^k(f,I)_p = m^k \left(\sum_{J\in u_m} (E_k(f,J)_p)^p\right)^{\frac{1}{p}} = \left|I\right|^k \left(\sum_{J\in u_m} \left(\frac{E_k(f,J)_p}{|J|^k}\right)^p\right)^{\frac{1}{p}},$$

где J - полуинтервал из равномерного разбиения полуинтервала I на m одинаковых полуинтервалов.
Докажем сначала справедливость требуемой формулы для $(l-1)$ раз непрерывно дифференцируемого сплайна s порядка $k+1$(степени k). Т.е., что

$$\lim_{m\to\infty} m^k\, U_m^k(s,I)_p = c_{k,p}\,|I|^k\,\|D^k s\|_{L_p[I]},$$

где функция $D^k s$ совпадает с $s^{(k)}$ там, где последняя определена.
По лемме из [3], при всех $0 < p < \infty$ вышолняется:

$$E_k\Big(\frac{t^k}{k!}; J\Big)_p = c_{k,p}\,|J|^{k+\frac{1}{p}}.$$

Поэтому, для многочлена π_{k+1} порядка $k+1$ (степени k) получим:
$E_k\Big(\pi_{k+1}; J\Big)_p = E_k\Big(a_{k+1}t^k; J\Big)_p = |a_{k+1}|\;k!\;E_k\Big(\frac{t^k}{k!}; J\Big)_p = |a_{k+1}|\;k!\;c_{k,p}\,|J|^{k+\frac{1}{p}} = c_{k,p}\,|J|^{k+\frac{1}{p}}\,|\pi_{k+1}^{(k)}|,$
где a_{k+1} - коэффициент при степени k в многочлене π_{k+1} .

Рассмотрим для наглядности сплайн s_n обсуждаемого вида, подчиненный некоторому равномерному разбиению u_n (рассуждение полностью сохраняется и в общем случае). Для подпоследовательности равномерных разбиений (u_m), являющихся измельчениями разбиения u_n (т.е., $m = \theta \cdot n$, $\theta \in N$) получаем:

$$m^k\, U_m^k(s_n,I)_p = |I|^k \left(\sum_{J\in u_m} \left(\frac{E_k(s_n,J)_p}{|J|^k}\right)^p\right)^{\frac{1}{p}} = |I|^k \left(\sum_{J\in u_m} \left(c_{k,p}\left|s_n^{(k)}(\xi_J)\right|\left|J\right|^{\frac{1}{p}}\right)^p\right)^{\frac{1}{p}} =$$

$$= c_{k,p}\,|I|^k \left(\sum_{J\in u_m} \|s_n^{(k)}\|_{L_p[J]}^p\right)^{\frac{1}{p}} = c_{k,p}\,|I|^k\,\|D^k s\|_{L_p[I]},$$

здесь ξ_J - некоторая точка из полуинтервала J.

Для произвольного равномерного разбиения u_m, $m \geq n$, имеется не более $n-1$ полуинтервалов $J_m \in u_m$, содержащих по узлу сплайна s_n. На каждом из этих полуинтервалов выполняется:

$$E_k(s_n, J_m)_p \leq |J_m|^{\frac{1}{p}} E_k(s_n, J_m)_\infty \leq \gamma\, |J_m|^{\frac{1}{p}+l} \|s_u^{(l)}\|_\infty$$

с постоянной γ, зависящей только от параметра k. Значит, если $\frac{1}{p} + l > k$, то

$$\frac{E_k(s_n, J_m)_p}{|J_m|^k} \to 0 \text{ при } m \to \infty;$$

поэтому вклад этих интервалов в общую сумму, соответствующую $m^k\ U_m^k(s_n, I)_p$, стремится к нулю с возрастанием m.

По такой же схеме утверждение доказывается для функций из формулировки теоремы.

Пусть функция f, кусочно принадлежащая $W_1^k[I]$, $I = [a, b]$, находится в пространстве $W_\infty^l[I]$. $a = t_o < t_1 < \cdots < t_n = b$, $n \in N$, - точки на I такие, что $f \in W_1^k[t_{i-1}, t_i]$ для всех $1 \leq i \leq n$.

Для функции $f \in W_1^k[J]$ и произвольного числа $\varepsilon > 0$ существует число $\delta > 0$ такое, что для каждого разбиения τ отрезка J с $\lambda(\tau) < \delta$ найдется $(k-1)$ раз непрерывно дифференцируемая сплайн-функция s порядка $k+1$, подчиненная разбиению τ , для которой выполняется: $\|(f-s)^{(k)}\|_{L_1[J]} < \varepsilon$. Это следует, например, из плотности множества k раз непрерывно дифференцируемых функций в W_1^k и равномерной непрерывности непрерывной функции (т.е. сколь угодно точного приближения непрерывной функции $h^{(k)}$ кусочно-постоянными функциями $s^{(k)}$, подчиненными достаточно мелким разбиениям). Очевидно, можно считать, что значения функции и сплайна, а также значения их первых l (по условию, $l \leq k-1$) производных совпадают на концах отрезка J.

На каждом отрезке $[t_{i-1}, t_i]$ подберем к функции f соответствующую сплайн-функцию s_i, и все их объединим в один сплайн s. Таким образом, для любого числа $\varepsilon > 0$ найдется сплайн s, удовлетворяющий условию: $\|D^k(f-s)\|_{L_1[I]} < \varepsilon$.

Для произвольного равномерного разбиения u_m, $m \geq n$, имеется не более $n-1$ полуинтервалов $J' \in u_m$, содержащих по точке из множества $\{t_i\}_{i=1}^n$.

Запишем:

$$m^k\ U_m^k(f, I)_p = \left|I\right|^k \left(\sum_{J \in u_m} \left(\frac{E_k(f, J)_p}{|J|^k} \right)^p + \sum_{J' \in u_m} \left(\frac{E_k(f, J')_p}{|J'|^k} \right)^p \right)^{\frac{1}{p}}.$$

Изучим поведение каждой из сумм.

Если $g \in W_1^k[J]$, применяя разложение Тейлора с остаточным членом в интегральной форме, легко получить оценку:

$$E_k(g, J)_p \leq \frac{1}{(k-1)!} \left|J\right|^{k-1+\frac{1}{p}} \left\|g^{(k)}\right\|_{L_1[J]}.$$

Выходит, что для данной функции f и приближающего сплайна s кусочно-полиномиальные приближения в L_p $(0 < p < 1)$ очень близки по величине (т.к. $f - s \in W_1^k[J]$ для каждого полуинтервала J из первой суммы):

$$\left| \sum_{J \in u_m} \left(\frac{E_k(f, J)_p}{|J|^k} \right)^p - \sum_{J \in u_m} \left(\frac{E_k(s, J)_p}{|J|^k} \right)^p \right| \leq$$

$$\leq \sum_{J \in u_m} \left(\frac{E_k(f-s, J)_p}{|J|^k} \right)^p \leq \left(\frac{1}{(k-1)!} \right)^p \sum_{J \in u_m} \left\|(f-s)^{(k)}\right\|_{L_1[J]}^p |J|^{1-p}$$

Применив неравенство Гельдера с показателями $\frac{1}{p}$ и $\frac{1}{1-p}$, получаем:

$$\sum_{J \in u_m} \left\|(f-s)^{(k)}\right\|_{L_1[J]}^p |J|^{1-p} \leq \left(\sum_{J \in u_m} \left\|(f-s)^{(k)}\right\|_{L_1[J]} \right)^p \left(\sum_{J \in u_m} |J| \right)^{1-p} \leq \left\|D^k(f-s)\right\|_{L_1[I]} \left|I\right|^{1-p}.$$

Поскольку величину $\left\|D^k(f-s)\right\|_{L_1[I]}$ можно неограниченно уменьшить, то поведение главной части выражения $m^k\ U_m^k(f, I)_p$ при $m \to \infty$ соответствует поведению такой суммы для сплайна (см.ниже).

Поведение второй суммы также соответствует поведению аналогичных сумм для сплайнов:

$$\sum_{J' \in u_m} \left(\frac{E_k(f, J')_p}{|J'|^k} \right)^p \le \gamma^p \sum_{J' \in u_m} \left(\frac{|J'|^{\frac{1}{p}+l}}{|J'|^k} \left\| f^{(l)} \right\|_\infty \right)^p,$$

т.е., если $\frac{1}{p} + l > k$,

$$\sum_{J' \in u_m} \left(\frac{E_k(f, J')_p}{|J'|^k} \right)^p \to 0 \text{ при } m \to \infty.$$

Получаем:

$$\left| \left(m^k \, U_m^k(f, I)_p \right)^p - \left(c_{k,p} \, |I|^k \, \|D^k f\|_{L_p[I]} \right)^p \right| \le \left| \left(m^k \, U_m^k(f, I)_p \right)^p - \left(m^k \, U_m^k(s, I)_p \right)^p \right| +$$

$$+ \left| \left(m^k \, U_m^k(s, I)_p \right)^p - \left(c_{k,p} \, |I|^k \, \|D^k s\|_{L_p[I]} \right)^p \right| + \left| \left(c_{k,p} \, |I|^k \, \|D^k s\|_{L_p[I]} \right)^p - \left(c_{k,p} \, |I|^k \, \|D^k f\|_{L_p[I]} \right)^p \right|.$$

Первое и третье слагаемые в правой части неравенства могут быть одновременно сделаны сколь угодно малыми за счет выбора сплайна s, второе - неограниченно приближается к нулю с увеличением m.

Теорема доказана.

Список литературы

1. Морозов А.Н. Аналог теоремы Бернштейна в пространстве L_1//Матем.заметки. 1995. Т.57. N5. С.699-703.

2. Морозов А.Н. Об одном описании пространств дифференцируемых функций//Матем.заметки. 2001. Т.70. N5. С.758-768.

3. Морозов А.Н. Асимптотическое поведение наилучших кусочно-полиномиальных приближений в пространствах L_p $(0 < p < 1)$ //Модел. и анализ информ. систем. 2004. Т.11. N1. С.24-27.

Модел. и анализ информ. систем. Т.11, №2 (2004) 22–31

УДК 681.3

Улучшенная ролевая модель управления доступом к объектам

Майоров А.В.
Ярославский государственный университет
150 000, Ярославль, Советская, 14

получена 23 мая 2004

Аннотация

В статье предлагается новая модель управления доступом к объектам приложения. Говорится о том, какие преимущества она имеет по сравнению с классическими моделями, и показывается, как эти модели можно реализовать в ее рамках.

1. Введение

В статье освещаются общие принципы построения систем безопасности и дается описание базовых моделей, обычно использующихся для этой цели (раздел 3). Детально рассматриваются ограничения ролевой модели, которая, из-за сравнительной простоты администрирования, является предпочтительной к использованию в прикладных программах (раздел 4).

Подробно описываются возможности и составные части разработанной нами модели и показывается, что каждая из них лишена вышеупомянутых ограничений (раздел 5). Доказывается, что в рамках предлагаемой модели можно реализовать базовые ролевую и дискреционную модели, а также обсуждаются возможные пути соединения ее с мандатной моделью (раздел 6).

В заключении подчеркивается основное отличительное свойство нашей модели и кратко говорится о преимуществах, которые оно несет.

2. Постановка задачи и результат

Перед нами стояла задача разработать модель управления доступом к объектам приложения, позволяющую реализовывать любую разумную политику безопасности приложения, оставаясь при этом максимально удобной в администрировании.

В результате получена модель, которая, с одной стороны, совместима с известными базовыми моделями, а с другой стороны, вводит дополнительную степень свободы - привязку пользовательских полномочий к иерархии объектов системы. Подобная привязка позволяет ограничить область действия выданных пользователю полномочий и, как следствие, упростить схемы доступа к объектам.

3. Базовые модели

Общим подходом для всех моделей управления доступом является разделение множества сущностей, составляющих систему, на множества объектов и субъектов. При этом определения понятий "объект" и "субъект" могут существенно различаться. Мы будем подразумевать, что объекты являются некоторыми контейнерами с информацией, а субъекты - пользователи, которые выполняют различные операции над этими объектами.

Безопасность обработки информации обеспечивается путем решения задачи управления доступом субъектов к объектам в соответствии с заданным набором правил и ограничений, которые образуют политику безопасности.

Можно выделить три основные модели управления доступом к объектам: мандатную, дискреционную и ролевую.

1. Мандатная модель

Классической мандатной моделью считается модель Белла-ЛаПадулы [1]. Она базируется на правилах секретного документооборота, использующегося в правительственных учреждениях. В этой модели каждому объекту и субъекту (пользователю) системы назначается свой уровень допуска. Все возможные уровни допуска системы четко определены и упорядочены по возрастанию секретности. Действуют два основных правила:

1. Пользователь может читать только объекты с уровнем допуска не выше его собственного.

2. Пользователь может изменять только те объекты, уровень допуска которых не ниже его собственного.

Цель первого правила очевидна каждому, второе может вызвать недоумение. Смысл же его в том, чтобы воспрепятствовать пользователю с высоким уровнем доступа, даже случайно, раскрыть какие-то известные ему тайны.

Одной из проблем этой модели считается беспрепятственность обмена информацией между пользователями одного уровня, так как эти пользователи могут выполнять в организации разные функции, и то, что имеет право делать пользователь А, может быть запрещено для Б. Поэтому в практике мандатную модель обычно используют совместно с какой-нибудь другой [2] [3].

Из этих двух правил можно вынести несколько интересных наблюдений, указывающих на проблемы, которые могут проявиться в процессе адаптации модели к реальному приложению:

• Пользователи "снизу" могут попытаться передать информацию наверх, выложив ее на своем уровне. При этом они никогда не узнают, читал ли ее кто-либо "сверху" или нет, так как документ будет защищен от редактирования вышестоящими лицами.

• Еще пользователи могут попробовать "закинуть" данные на уровень выше. В этом случае верха будут иметь возможность вставить в полученный документ свои комментарии, но отправитель об этом также не узнает. Вообще, о существовании верхних уровней он может узнать только из документации к системе.

• У пользователей с высоким уровнем допуска нет никаких возможностей коммуникации с нижними уровнями. Возможно, наверху сидят очень умные люди, советы которых были бы просто бесценны, но мы об этом никогда не узнаем.

2. Дискреционная модель

В дискреционной модели безопасности управление доступом осуществляется путем явной выдачи полномочий на проведение действий с каждым из объектов системы. Например, в модели Харрисона-Руззо-Ульмана [4] для этого служит матрица доступа, в которой определены права доступа субъектов системы к объектам. Строки матрицы соответствуют субъектам, а столбцы - объектам. Каждая ячейка матрицы содержит набор прав, которые соответствующий субъект имеет по отношению к соответствующему объекту.

Как правило, создатель объекта обладает на него полными правами и может делегировать часть прав другим субъектам.

Дискреционный подход позволяет создать гораздо более гибкую схему безопасности, чем мандатный, но при этом он и гораздо более сложен в администрировании. С программной точки зрения его реализация очень проста, но при достаточно большом количестве объектов и субъектов система становится практически неуправляемой.

Для решения этой проблемы применяется, например, группировка пользователей. В этом случае права раздаются группам пользователей, а не каждому пользователю в отдельности. Для того чтобы пользователь получил соответствующие разрешения, нужно просто добавить его в одну или несколько групп.

Также можно использовать типизацию объектов. Каждому объекту назначается тип, а для каждого типа определяется свой набор прав (схема доступа). В этом случае столбцы матрицы доступа соответствуют не объектам, а типам объектов. Комбинирование этого подхода с группировкой пользователей позволяют существенно уменьшить матрицу доступа, а значит, и упростить ее администрирование.

В сущности, набор прав - это не что иное, как список известных системе операций, снабженных разрешением или запретом на выполнение данной операции. В крупном приложении количество известных операций может быть весьма большим. При этом большая часть операций имеет смысл только для определенных типов объектов, а многие типовые процессы, осуществляемые пользователем в приложении, включают в себя выполнение нескольких элементарных операций над различными объектами. Поэтому, даже с уменьшенной матрицей доступа, продумать политику безопасности приложения, т.е. грамотно разделить полномочия между различными пользователями системы, достаточно сложно.

3. Ролевая модель

В ролевой модели [1] операции, которые необходимо выполнять в рамках какой-либо служебной обязанности пользователя системы, группируются в набор, называемый "ролью". Например, операции по регистрации документов могут быть сгруппированы в роль "регистратор".

Для того чтобы множества операций, связанных с различными ролями, не пересекались, вводится иерархическая зависимость между ролями. К примеру, роль "секретарь" может включать в себя роль "регистратор" и, плюс к тому, еще несколько дополнительных операций.

Каждый пользователь системы играет в ней одну или несколько ролей. Выполнение пользователем определенного действия разрешено, если в наборе его ролей есть нужная, и запрещено, если есть нежелательная.

В этой модели у объектов нет определенных хозяев. Вся информация расценивается как принадлежащая организации, владеющей системой. Соответственно, и роли пользователя внутри системы - это роли, которые он играет в данной организации. Как следствие, пользователю невозможно делегировать права на какой-то определенный объект. Либо у него есть доступ ко всем подобным объектам системы, либо нет.

Таким образом, преимуществом ролевой модели перед дискреционной является простота администрирования: назначение пользователей на роли и создание новых ролей не составляют никаких трудностей. В то же время она не позволяет управлять разными частями системы по отдельности и тем более - делегировать какому-либо пользователю такие полномочия.

4. Ограничения ролевой модели

Выше мы говорили о том, что ролевая модель контроля доступа является компромиссным решением, обеспечивающим неплохие возможности в задании политики безопасности при достаточной простоте администрирования. Это позволяет рассматривать ролевую модель как наиболее подходящую для применения в прикладных программах. В то же время существующие ограничения в ряде случаев сильно затрудняют ее использование. Рассмотрим эти ограничения более подробно.

1. Глобальность ролей

Первое ограничение состоит в том, что пользователь принимает свои роли по отношению ко всей системе сразу. Соответственно, для системы нет разницы в правах между двумя пользователями, находящимися на одинаковой должности, даже если они занимают эти должности в разных отделах. Например, любой пользователь в роли "начальник отдела" имеет право управлять любым отделом своей организации, а это, конечно, неправильно.

Решением могло бы стать введение отдельных ролей "начальник отдела А", "начальник отдела Б" и т.п., что позволило бы нам решить проблему, не выходя за рамки ролевой модели. К сожалению, подобный вариант привносит гораздо больше проблем, чем решает.

Более правильным будет разбить все множество объектов системы на несколько подмножеств (доменов) и дать пользователям возможность играть разные роли в разных доменах системы. В нашем примере систему можно разбить на отделы, так что каждый из начальников отделов будет играть роль "начальник" только в домене, соответствующем его отделу. Такой подход применяется, например, в библиотеке Microsoft Authorization Manager [6].

2. Отсутствие владельца объекта

Второе препятствие перед использованием ролевой модели в ряде систем - это отсутствие в ней понятия владельца объекта. Другими словами, пользователь, создавший объект, не имеет на него никаких исключительных прав. Это вполне приемлемо для систем, поддерживающих, например, процесс купли-продажи, но перестает годиться, как только документы начинают содержать какие-то авторские материалы.

Зачастую для решения этой проблемы к объектам системы добавляют свойство "владелец", являющееся внешним по отношению к модели безопасности. В том случае, если пользователь является владельцем объекта, над которым он хочет совершить действие, проверка производится по специальным отдельным правилам, а не по тем общим, которые предусмотрены ролевой моделью.

Другой подход - ввести в ролевую модель элементы дискреционной и явным образом дать пользователю нужные права на созданный им объект.

Заметим, что оба эти варианта решают задачу, используя внешние по отношению к модели средства, и, соответственно, не снимают ограничений самой модели.

3. Операции принадлежат ролям

Казалось бы, группировка операций системы в роли, в рамках которых они выполняются, упрощает администрирование, но это снова верно не для всех типов систем. Предположим, что в нашей системе есть десять различных типов объектов, для каждого из которых определена операция "удалить". Тогда, если мы добавляем эту операцию в какую-либо из ролей, то любой пользователь, играющий эту роль, получает право удалять объекты любого типа. Очевидно, что это далеко не всегда является желательным поведением.

Можно попытаться решить эту проблему, введя десять различных операций, предназначенных для удаления объекта каждого из типов. Это будет выглядеть примерно так: "удалить статью", "удалить папку" и т.д. Такое решение оказывается не очень удачным, если типов не десять, а, например, сто.

Проблема еще более усугубляется, если в системе возможны различные схемы доступа к разным объектам одного типа. Например, объект типа "статья" может быть или открытым для публики, или совершенно секретным. Создавать действия "удалить публичную статью", "удалить секретную статью", "удалить публичную папку", "удалить секретную папку" и т.п. кажется совершенно неразумным.

В принципе, можно еще более увеличить количество операций, пытаясь использовать подобный подход в решении "проблемы владельца": добавить операцию "удалить свою статью", и использовать ее для проверки прав на удаление статей, принадлежащих данному пользователю.

На практике, при использовании ролевой модели в сложных системах, разработчики обычно не пытаются декларативно задавать схему доступа к объектам системы. Вместо этого процедура проверки встраивается в нужное место программы, при этом проверяются как сведения, предоставляемые модулем ролевой безопасности (т.е. роли, в которых выступает пользователь), так и любые другие сведения об объекте (владелец объекта, уровень секретности и т.п.).

Подобный подход затрудняет изменение схемы доступа, так как для этого нужно исправлять код процедуры проверки и перекомпилировать приложение. Для упрощения этой процедуры некоторые библиотеки ролевой безопасности предлагают встроенные средства написания сценариев проверки. Эти сценарии являются фактически теми же процедурами, но написанными на другом языке программирования, и расположенными не в самой программе, а среди настроек библиотеки. В Microsoft Authorization Manager программирование сценариев ведется на VBScript и теоретически может производиться администратором системы, так как не требует перекомпилирования всей программы.

5. Модель Force-Field

Force-Field - это разработанная нами модель управления доступом, которая позволяет создавать простые в администрировании политики безопасности, является существенно более мощной, чем ролевая модель, и при этом лишена ее основных недостатков.

1. Дерево объектов

В разделе 4.1 обсуждалась проблема диапазона действия ролей в ролевой модели, описанной в [1], и вариант ее решения с использованием доменов. Недостаток этого решения в том, что домены нужно создавать вручную, и они явным образом не связаны ни с какими объектами системы. Фактически приложение само должно решить, к какому домену относится текущая проверка, и передать эту информацию библиотеке. К тому же отсутствует иерархия доменов, а значит, наборы ролей пользователя в разных доменах приложения совершенно независимы. Это создает трудности, если пользователь должен играть определенную роль во всей системе сразу - его придется назначить на эту роль в каждом домене в отдельности.

В модели Force-Field все объекты системы объединяются в единое дерево. У каждого объекта, кроме единственного корневого, есть один родительский объект, и любое количество дочерних. Роль может быть назначена пользователю в контексте любого объекта. При этом пользователь начинает играть назначенную роль во всей ветви дерева, которая образована этим объектом.

Отметим, что терминология, использующаяся в работе, подразумевает, что дерево объектов "растет" вниз. Самым верхним объектом является корень, а перемещение от него к ветвям - это движение вниз (или вглубь) иерархии.

Таким образом, любой объект приложения может образовать домен (см. раздел 4.1), в который будут входить он сам и все его дочерние объекты. Любой из его дочерних объектов может также образовать домен, являющийся подчиненным по отношению к домену родительского объекта. Список ролей, которые пользователь играет в определенном домене, состоит из ролей, назначенных ему в данном домене, плюс роли из домена более высокого уровня, и из домена еще более высокого уровня, и так далее, до корневого

домена приложения. Роли, назначенные пользователю в корневом домене, имеют глобальный характер, т.е. действительны в контексте каждого объекта приложения.

Например, корневой объект "предприятие" имеет десять подчиненных объектов типа "отдел", под каждым из которых, в свою очередь, располагаются документы, относящиеся к данному отделу. Пользователь, которому назначена роль "начальник" в контексте конкретного отдела (или отделов), имеет полный доступ ко всем документам своего отдела, но не имеет доступа к документам других отделов, так как там он не играет соответствующей роли. Пользователь же, являющийся "начальником" корневого объекта системы, имеет полный доступ ко всем документам предприятия, что полностью соответствует соображениям элементарной логики.

2. Роль "владелец"

Вернемся еще раз к проблеме отсутствия владельцев у объектов в базовой ролевой модели (см. раздел 4.2). В качестве решения подойдет любой механизм, позволяющий выделить хозяина объекта и дать ему особые права на этот объект. В нашей модели для этого вводится роль "владелец", назначающаяся пользователям в контексте тех объектов, которыми они владеют.

Эта роль по своему поведению слегка отличается от обычных ролей. Во-первых, только один пользователь может играть роль "владелец" в контексте какого-то определенного объекта. Во-вторых, объект не должен наследовать роль "владелец" от родителя, если в его собственном контексте такая роль кому-либо назначена.

Первый принцип очевиден, второй поясним на примере. Пусть у нас есть иерархия объектов "дом" - "квартира". "Дом" является родительским объектом для нескольких "квартир". Если некий пользователь является владельцем "дома", то он также является и владельцем всех "квартир", у которых нет своих собственных владельцев. "Квартиры" же с явно указанным владельцем ему не принадлежат.

Развивая эту идею, заметим, что роль может быть ограничена не только одним актером, но и большим их количеством. Таким образом, если роль ограничена n пользователями, то в контексте любого объекта системы не больше n пользователей могут играть эту роль. При этом, очевидно, что в системе в целом у этой роли может быть больше n назначений.

В базовой модели подобные ограничения называются кардинальностью роли и определены как максимальное количество пользователей, которые могут играть эту роль в рамках всей системы.

3. Класс доступа

В разделе 4.3 было показано, что, хотя распределение операций по ролям кажется логичным, оно весьма затрудняет разработку схемы безопасности. Использование же сценариев для проверки прав доступа усложняет администрирование системы.

Для создания гибкой схемы безопасности без ручного программирования сценариев проверки, в Force-Field введено понятие "класс доступа". Класс доступа содержит набор правил, задающих права выполнения определенных операций для определенных ролей. Например, правило может быть таким: "роли Администратор выполнять операцию Удаление Разрешено". Порядок следования правил важен, так как при проверке поиск делается сверху вниз и продолжается до тех пор, пока не будет найдено правило, подходящее проверяемой ситуации. Правило из нашего примера подойдет, если будет запрошено разрешение на удаление объекта, а пользователь в контексте этого объекта будет администратором.

Каждому объекту системы ставится в соответствие ровно один класс доступа, а любой класс доступа может быть назначен произвольному количеству объектов. Это позволяет иметь в системе несколько разных схем доступа к объектам, не заставляя нас связывать эти схемы с типами или какими-то другими признаками объектов. Отметим, также, что назначения классов никак не связаны с иерархией объектов: дочерний объект может иметь любой класс доступа, независимо от того, какой класс назначен родительскому объекту.

Класс может базироваться на другом классе, так что, если подходящего правила в классе нет, будет просмотрен базовый класс, потом его базовый класс и так далее. Если правило так и не будет найдено, то операция считается запрещенной. Этот механизм также служит для упрощения администрирования системы.

Перечислим некоторые возможные варианты распределения классов по объектам:

1. У всех объектов одного типа один и тот же класс доступа. Следует применять в системах, где различные типы отличаются друг от друга по правилам контроля доступа, но все объекты одного типа ведут себя одинаково.

2. Есть несколько классов доступа, которые могут быть назначены любому объекту, независимо от его типа. Например, классы, определяющие уровень секретности информации (публичная, для служебного пользования, совершенно секретная).

3. Есть всего один класс доступа, который назначается всем объектам.

4. Иерархии

Помимо единой иерархии объектов, в нашей модели предусмотрена также иерархия ролей и иерархия операций.

Как и в базовой модели, роль может включать в себя любое количество других ролей (циклы запрещены). В этом случае, если пользователь играет в каком-то контексте некоторую роль, то он играет и все подчиненные ей роли. Корнем иерархии ролей является роль "любая роль".

Иерархия операций позволяет упростить администрирование, раздавая права не на каждую операцию в отдельности, а на целые группы операций. Например, имея операции "создать объект типа А", "... Б", "... В" и т.д., мы могли бы добавить объединяющую их операцию "создать объект". Пользователь, которому разрешена эта операция, будет иметь право создавать объекты любого типа. Как и с ролями, у этой иерархии есть корень - операция "любая операция".

5. Наследование

Мы уже говорили о двух механизмах наследования в нашей модели - это, во-первых, наследование списка ролей пользователя при движении вглубь по иерархии объектов, и, во-вторых, наследование правил из базовых классов. Не хватает еще одного - наследования правил доступа от вышестоящих объектов. Подобный механизм часто встречается в файловых системах (дискреционная модель): файлы, лежащие в папке, могут не иметь своих собственных правил доступа, а наследовать эти правила у папки. При переносе этих файлов в другую папку, права пользователя на доступ к этим файлам могут поменяться.

В Force-Field это реализуется следующим образом: в любом правиле класса кроме резолюций "разрешено" и "запрещено" можно использовать вариант "как у родителя". В этом случае, для выдачи окончательного ответа будет проверено, можно ли данному пользователю выполнить запрашиваемое действие по отношению к родительскому объекту. Если можно, то и на первоначальный запрос ответ будет положительным, нет - отрицательным. Естественно, этот процесс может быть рекурсивным: если родительский объект также не имеет своего мнения, то будет проверен его родительский ответ и так далее. Если по достижении корня иерархии объектов явного разрешения или запрета так и не будет найдено, то действие считается запрещенным.

Чтобы продублировать описанное выше поведение файловой системы, достаточно создать класс, с которым объект наследует от родителя права на выполнение всех операций. В то же время наша модель позволяет и такой вариант, когда права на часть операций наследуются, а для другой части задаются явным образом.

6. Гибридность модели Force-Field

Докажем, что предлагаемая нами модель позволяет реализовать в своих рамках функциональность базовых ролевой (раздел 3.3) и дискреционной (раздел 3.2) моделей.

1. Реализация ролевой модели

Предположим, что в нашем приложении четыре объекта двух различных типов: A_1, A_2, B_1, B_2. Существуют две операции, которые можно выполнить над объектом типа A, и одна для объектов типа B: op_{A1}, op_{A2}, op_{B1}. Также в системе зарегистрировано два пользователя: U_1 и U_2.

Базовая ролевая модель предписывает нам ввести в систему роли и распределить операции между ролями. Введем следующие роли. Роль r_1 включает в себя операции op_{A2} и op_{B1}, а роль r_2 — операцию op_{A1}. Назначим на роли пользователей: U_1 играет в системе роль r_2, а U_2 — обе роли. Напомним, что пользователь, назначенный на определенную роль, имеет право выполнять операции, входящие в эту роль, по отношению к любому объекту системы. Эта схема проиллюстрирована на рис. 1.

Чтобы реализовать ее в нашей модели, необходимо сначала свести все объекты приложения в единую иерархию. Для этого достаточно добавить фиктивный корневой объект — *root* — и сделать объекты его прямыми потомками. Для того чтобы все роли были глобальными (как того требует базовая модель)

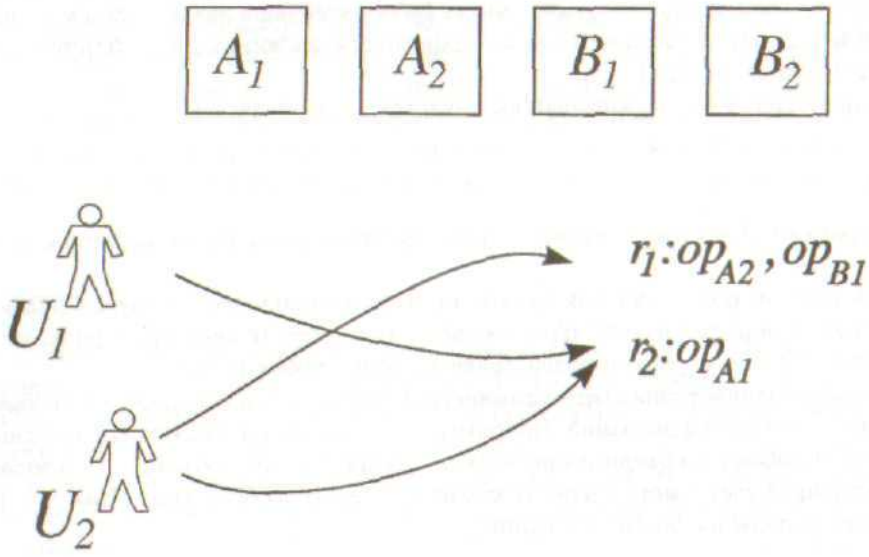

Рис. 1.

мы должны будем назначать пользователей на них только в контексте корневого объекта, или, другими словами, в корневом домене d_{root}. Мы вводим роли r_1 и r_2 и назначаем на них пользователей.

После введения в систему операций остается только одна нерешенная проблема: необходимо, чтобы роль определялась операциями, выполняемыми в ее рамках. Действительно, роль в модели Force-Field, в общем случае, не соответствует этому требованию. Фактически она ничем не отличается от группы пользователей.

Решение заключается во вводе в систему единого для всех объектов класса доступа c_0, в котором операции, составляющие определенную роль, для этой роли явным образом разрешены. Очевидно, что это и есть нужный нам способ записи соответствия между ролями и операциями.

Принципиальная схема реализации приведена на рис. 2.

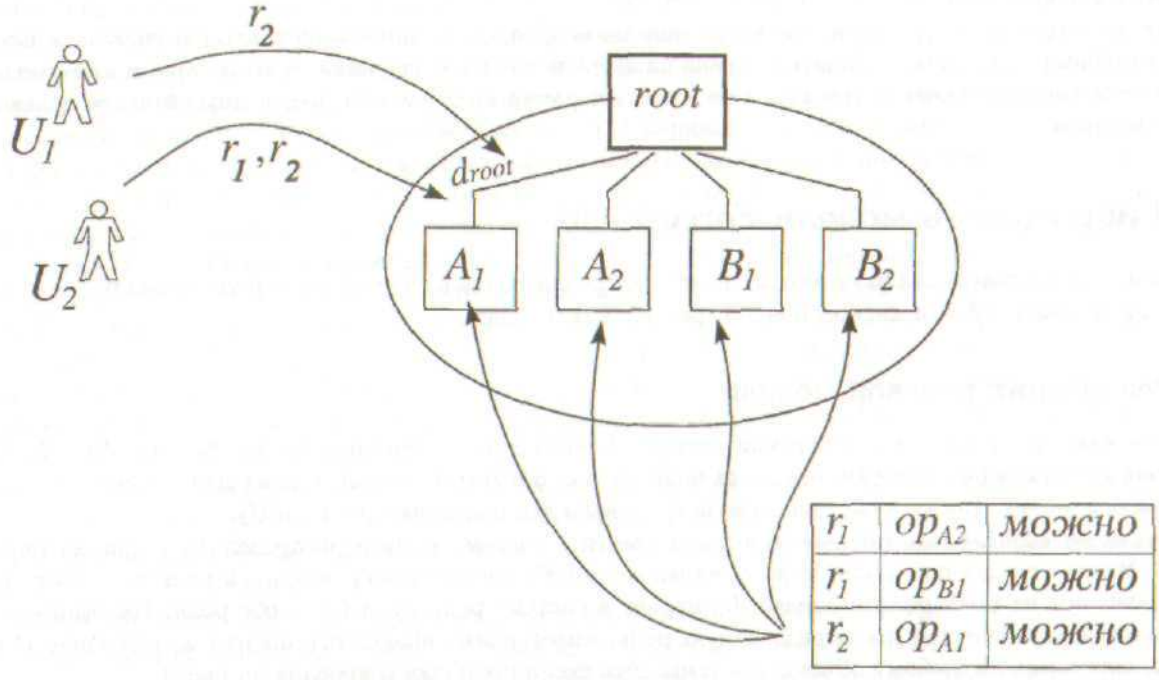

Рис. 2.

2. Реализация дискреционной модели

Для приложения, описанного в предыдущей задаче, система контроля доступа по модели Харрисона-Руззо-Ульмана будет подобна схеме на рис. 3. В ней столбцы соответствуют объектам, строки - пользователям. В ячейках прописаны индивидуальные права пользователя на соответствующий объект. Отметим, что, хотя в этом и нет большой необходимости, в таблице явным образом запрещены операции, не имеющие смысла для соответствующих объектов.

	A_1	A_2	B_1	B_2
U_1	$op_{A1} = да$ $op_{A2} = нет$ $op_{B1} = нет$	$op_{A1} = да$ $op_{A2} = нет$ $op_{B1} = нет$	$op_{A1} = нет$ $op_{A2} = нет$ $op_{B1} = нет$	$op_{A1} = нет$ $op_{A2} = нет$ $op_{B1} = нет$
U_2	$op_{A1} = да$ $op_{A2} = да$ $op_{B1} = нет$	$op_{A1} = да$ $op_{A2} = да$ $op_{B1} = нет$	$op_{A1} = нет$ $op_{A2} = нет$ $op_{B1} = да$	$op_{A1} = нет$ $op_{A2} = нет$ $op_{B1} = да$

Рис. 3.

Для реализации этой модели в Force-Field нужно создать четыре отдельных класса доступа (c_{A1}, c_{A2}, c_{B1} и c_{B2}) и назначить их соответствующим их объектам. В составляющих классы правилах разрешения будут даваться не ролям, а непосредственно пользователям. Необходимо подчеркнуть, что возможность указывать в правиле не роль, а пользователя, следует использовать только в крайних случаях, так как это может привести к созданию плохо управляемой политики безопасности.

В классы не включены правила, запрещающие не имеющие смысла операции, т.к. все явным образом не разрешенное автоматически считается запрещенным.

Результирующая схема приведена на рис. 4.

Обратим внимание, что на схеме не показан корневой домен и назначения пользователей на роли. Это связано с тем, что модель, которую мы рассматривали, является слишком простой, и роли в ней не используются. Как мы уже обсуждали в разделе 3.2, в таком виде модель не годится для большинства реальных применений, поэтому расширим ее, добавив типы объектов и группы пользователей.

Хотя в нашем приложении изначально есть два типа объектов, они до сих пор находились вне разрабатываемой системы безопасности. Введем типы в систему, связав с ними классы доступа. Таким образом, количество классов безопасности уменьшается до двух: класс безопасности для объектов типа A (c_A) и класс для объектов типа B (c_B).

Для группировки пользователей мы можем использовать роли, назначаемые пользователям в корневом домене. Поэтому введем две роли: g_1 и g_2 . Фактически это то же самое, что роли r_1 и r_2 из предыдущих моделей.

Изменим правила в классах доступа, с тем чтобы они выдавали разрешения группам пользователей (т.е. ролям), а не каждому пользователю в отдельности.

На рис. 5 приведена схема реализации дискреционной модели, оптимизированной за счет типизации объектов и введения групп пользователей. В нашей модели можно реализовать и другие способы оптимизации. Например, наследование прав по аналогии с файловой системой легко реализуется за счет специального класса доступа на дочернем объекте. Он должен определять собственную схему доступа объекта и дополнять ее правилом "любая роль - любая операция - как у родителя".

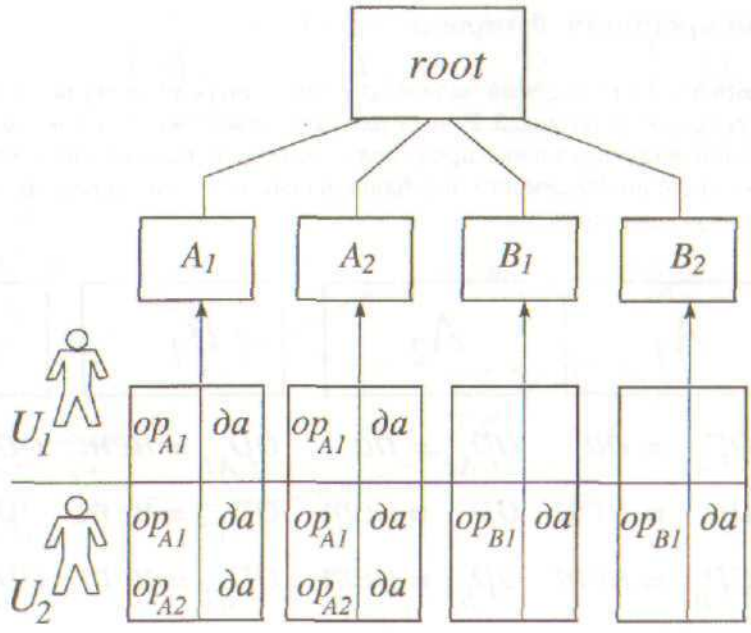

Рис. 4.

3. Введение элементов мандатной модели

Самой первой из рассмотренных нами классических моделей была мандатная модель (см. раздел 3.1). В дальнейшем мы не уделяли ей достаточного внимания, так как она является весьма экзотичной, и крайне редко применяется в реальных приложениях. В связи с этим мы не ставили перед собой задачу охвата и этой модели, но наметили пути, по которым это может быть сделано.

Во-первых, классам доступа нужно назначить уровни секретности. Так как каждый объект приложения проассоциирован с определенным классом доступа, это автоматически назначит уровни секретности и всем объектам. При этом если уровень секретности классу все же не назначен, то можно считать, что класс и его объекты находятся на самом низком уровне.

Во-вторых, мы должны назначить уровни допуска пользователям системы. В соответствии с моделью, пользователь будет иметь право читать документы с уровнем секретности не выше его собственного, и изменять документы с уровнем не ниже.

Это подводит нас к третьему пункту. Необходимо разделить все операции системы на две группы: группу чтения и группу записи. Тогда, при проверке на допустимость операции, мы будем точно знать, какое правило применить. Дополнительно мы могли бы добавить и еще одну группу, принадлежность к которой означала бы, что операцию не нужно проверять по мандатной модели. Это позволит сделать, например, операцию уведомления о прочтении документа (ранее мы уже обсуждали, какие сложности возникают с этим в строгой мандатной модели).

Наконец, модифицируется процедура проверки прав. Если у объекта и пользователя разные уровни, то мы проводим проверки по стандартным принципам мандатной модели. Если уровень одинаковый или операция входит в "свободную" группу, то применяем обычные правила нашей модели.

Заметим, что нужно будет еще тщательно обдумать желаемое поведение системы в ситуации, когда по мандатной модели операция разрешена, а по модели Force-Field - запрещена. Ответ на этот вопрос определит направленность системы безопасности. Нужно решить, что приоритетней: полная свобода пользователям с высоким уровнем или ограничение пользователей с низким.

7. Заключение

Мы показали, что предлагаемая модель контроля доступа объединяет в себе базовые модели. Что более важно, она расширяет их возможностью назначения пользователей на роли в контексте любого объекта системы. Поэтому множество ролей, которые пользователь играет в некий момент времени, не является одним и тем же для всех объектов приложения, а пополняется новыми ролями по мере спуска вглубь по объектной иерархии.

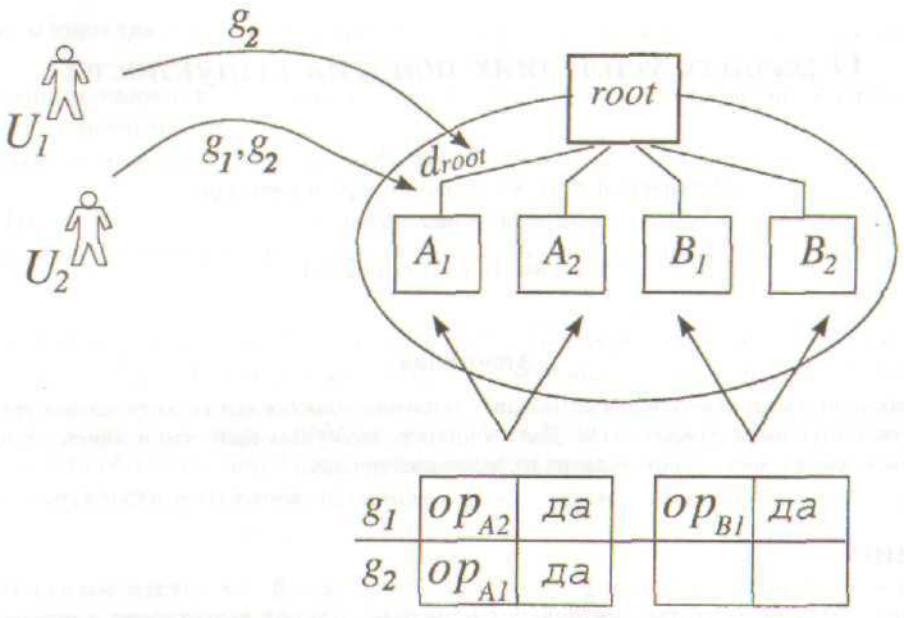

Рис. 5.

За счет этого появляется возможность максимально естественным образом ограничить область действия выданных пользователю полномочий определенной частью приложения. Это позволяет уменьшить необходимое количество ролей и упростить схемы доступа к объектам.

Таким образом, наша модель позволяет создавать легкие в администрировании политики безопасности, обладая, в то же время, необходимыми возможностями по ограничению несанкционированного доступа к объектам приложения.

Список литературы

1. Leonard J. LaPadula and D. Elliott Bell. Secure Computer Systems: A Mathematical Model // MITRE Corporation Technical Report 2547. Volume II. 31 May 1973.

2. Зегжда Д.П. Общая схема мандатных моделей безопасности и ее применение для доказательства безопасности систем обработки информации // Проблемы информационной безопасности. Компьютерные системы. СПбГТУ. 2000. 2.

3. Степанов П.Г. Принципы управления доступом к ресурсам в защищенной ОС "Феникс" // Проблемы информационной безопасности. Компьютерные системы. СПбГТУ. 1999. 4.

4. M. Harrison, W. Ruzzo, J. Uhlman. Protection in operating systems // Communications of the ACM. 1976.

5. Баранов А.П., Зегжда Д.П., Зегжда П.Д., Ивашко А.М., Корт С.С. Теоретические основы информационной безопасности (Дополнительные главы). СПб.: СПбГТУ. 1998.

6. Mohan Rao Cavale. Role-Based Access Control Using Windows Server 2003 Authorization Manager. http://msdn.microsoft.com/library/en-us/dnnetserv/html/AzManRoles.asp

Модел. и анализ информ. систем. Т.11, №2 (2004) 32–36

УДК 681.3

О разных усилениях понятия выпуклости

Карасёв Р.Н. [1]
Московский Физико-технический институт
e-mail: r_n_karasev@mail.ru

получена 15 сентября 2004

Аннотация

В этой работе рассматриваются два возможных усиления понятия выпуклости множества, обобщающие сильную выпуклость с радиусом R. Дается пример, показывающий, что в общем случае они не эквивалентны и дается достаточное условие их эквивалентности.

1. Введение

В этой работе приводятся результаты, связанных с понятием сильной выпуклости и порождающего множества. M-сильная выпуклость, определенная для любого замкнутого выпуклого множества M, является некоторым усилением понятия выпуклости, также в этой работе будет рассмотрено еще одно усиление понятие выпуклости, M-выпуклость.

Кроме того, нам понадобится понятие порождающего множества, так как нас будет интересовать случай сильной выпуклости, когда множество M — порождающее. Это понятие явно или неявно использовалось в работах [5, 3, 6, 7], при этом в работах [5, 3] было сформулировано собственно определение порождающего множества.

Введем некоторые обозначения, которые мы будем использовать:

1. $\operatorname{cl} X$, $\operatorname{int} X$, $\operatorname{bd} X$ — замыкание, внутренность и граница множества X, которое является подмножеством некоторого топологического пространства.

2. $\operatorname{conv} V$, $\operatorname{lin} V$, $\operatorname{aff} V$ — выпуклая, линейная, аффинная оболочка множества V, являющегося подмножеством некоторого линейного пространства.

3. Сумма Минковского непустых множеств A и B, которые являются подмножествами некоторого линейного пространства:
$$A + B = \{a + b : a \in A, b \in B\}$$
Если множество B состоит из одной точки b, то будем для краткости обозначать
$$A + \{b\} = A + b.$$

4. Геометрическая разность двух непустых множеств A и B, которые являются подмножествами некоторого линейного пространства:
$$A \overset{*}{-} B = \bigcap_{b \in B} (A - b)$$
или, эквивалентно,
$$A \overset{*}{-} B = \{c : B + c \subseteq A\}.$$

5. $\langle l, x \rangle$ — значение на векторе $x \in L$ линейного функционала $l \in L^*$. Здесь L — некоторое линейное пространство. Для $L = \mathbb{R}^n$ пространство L^* также отождествляется с $\mathbb{R}^n$.

6. (x, y) — скалярное произведение векторов в пространстве со скалярным произведением. Будем считать, что в $\mathbb{R}^n$ задано скалярное произведение $(x, y) = \sum_{i=1}^{n} x_i y_i$.

[1] Работа выполнена при поддержке РФФИ (гранты 03-01-00801 и 03-01-06207)

Все выпуклые множества в теоремах и определениях будем считать замкнутыми, если не упомянуто противное.

Во всех дальнейших теоремах E — рефлексивное банахово пространство, в частности это может быть конечномерное или гильбертово пространство.

Введем несколько определений, следуя работе [5].

Определение. Пусть $M \subset E$ — выпуклое множество. Множество $A \subset E$ называется *M-сильно выпуклым*, если оно является пересечением некоторого множества транслятов M, то есть $A = \bigcap_{t \in T}(M + t)$, где $T \subset E$.

Определение. Выпуклое множество $M \subset E$ называется *порождающим*, если для любого множества $T \subset E$, для которого $Y = \bigcap_{t \in T}(M + t)$ не пусто, найдется такое выпуклое множество Y^*, что $Y + Y^* = M$.

Как показано в работе [5], именно свойство множества M быть порождающим позволяет доказать аналоги многих свойств обычной выпуклости для сильной выпуклости.

Такое понятие выпуклости естественно приводит к определению выпуклой (сильно выпуклой) оболочки (см. [5]):

Определение. Пусть множество $S \subset E$ таково, что $M \stackrel{*}{-} S \neq \emptyset$. Тогда *$M$-выпуклой оболочкой* множества S называется множество

$$\bigcap_{t \in M \stackrel{*}{-} S} (M - t).$$

Мы будем обозначать его $\operatorname{conv}_M S$.

Замечание. Иначе говоря, $\operatorname{conv}_M S$ — это пересечение всех транслятов M, содержащих S. Также имеет место формула

$$\operatorname{conv}_M S = M \stackrel{*}{-} (M \stackrel{*}{-} S).$$

Можно заметить, что M-выпуклая оболочка является наименьшим по включению M-сильно выпуклым множеством, содержащим S.

В [2] приведено еще одно усиление понятия выпуклости, целью данной работы является сравнить его с определением сильной выпуклости:

Определение. Пусть $M \subset E$ — выпуклое множество. Множество $A \subset E$ называется *M-выпуклым*, если для любых двух точек $a, b \in A$ множество $\operatorname{conv}_M\{a, b\}$ определено и $\operatorname{conv}_M\{a, b\} \subseteq A$.

Данная работа посвящена сравнению этих определений, в частности будут доказаны теоремы:

Теорема 1. *Пусть дано множество $M \subset E$, не обязательно порождающее. Тогда любое M-сильно выпуклое множество является M-выпуклым.*

Теорема 2. *Пусть множество $M \subset E$ является порождающим, а множество A — M-выпуклым. Если $\operatorname{int} b(A) \neq \emptyset$, то A M-сильно выпукло. Если M ограничено, то условие $\operatorname{int} b(A) \neq \emptyset$ выполняется для любого M-выпуклого A.*

Эти теоремы позволяют, при условии, что множество M порождающее и ограниченное, распространить определение M-сильно выпуклого множества на незамкнутые выпуклые множества, так как, очевидно, в определении M-выпуклого множества можно вообще избавиться от требования замкнутости.

Также в этой работе будет дан пример M-выпуклого множества, которое не является M-сильно выпуклым. Множество M при этом ограничено, но не является порождающим.

Следующее утверждение было доказано в [7] в другой формулировке, мы сформулируем его с использованием определения порождающего множества:

Теорема 3. *Любое выпуклое множество $M \subset \mathbf{R}^2$ является порождающим.*

Поэтому, как уже было отмечено в [2], из результатов этой статьи в частности следует, что в двумерном случае определения M-выпуклости и M-сильной выпуклости эквивалентны.

2. Вспомогательные факты

Сформулируем некоторые вспомогательные факты и определения, необходимые при доказательстве наших теорем.

Определение. Выпуклое множество Y называется *слагаемым* множества X, если найдется такое выпуклое Y^*, что $Y + Y^* = X$.

Если $Y + Y^* = X$, то $X = \cup_{y^* \in Y^*}(Y + y^*)$, то есть X является объединением транслятов Y. Верно и обратное: если множество X выпукло и $X = \cup_{z \in Z}(Y + z)$, то взяв $Y^* = \operatorname{conv} Z$, получим $X = Y + Y^*$. Следовательно, чтобы установить, что Y — слагаемое X, достаточно проверить, покрывают ли те трансляты Y, которые содержатся в X, все множество X.

Определение. *Опорной функцией* множества X называется функция вектора $p \in E^*$, определяемая как

$$s(p, X) = \sup_{x \in X} \langle p, x \rangle.$$

Сформулируем понятия, необходимые для работы с неограниченными выпуклыми множествами.

Определение. *Барьерным конусом* множества X называется

$$b(X) = \{p \in E^* : s(p, X) < +\infty\}.$$

Лемма 1. *Пусть X — выпуклое множество и $p \in \operatorname{int} b(X)$. Тогда множество*

$$X_{p,a} = \{x \in X : \langle p, x \rangle \geq a\}$$

ограничено.

Доказательство. По условию, найдется некоторое открытое множество $U \subseteq E^*$ такое, что $p \in U$ и $U \subseteq b(X)$. Так как при этом очевидно $-p \in b(X_{p,a})$, то $U - p \subseteq b(X_{p,a})$ и при этом $U - p$ — окрестность нуля. Это означает, что $\langle p', x \rangle$ ограничено на X_{pa} при любом p'. По известной теореме (теорема 3.18 из [8]) отсюда следует ограниченность $X_{p,a}$. □

Приведем еще три известных леммы, которые также можно найти в [5].

Лемма 2. *Если $\operatorname{int} b(X) \neq \emptyset$, то для любой точки $y \notin X$ найдется $p \in \operatorname{int} b(X)$ такое, что $\langle p, y \rangle > s(p, X)$.*

Лемма 3. *Если $X = A + B$, то $s(p, X) = s(p, A) + s(p, B)$.*

Лемма 4. *Если $p \in \operatorname{int} b(X)$, то найдется точка $x \in X$ такая, что $\langle p, x \rangle = s(p, X)$.*

3. Доказательство теорем

Доказательство теоремы 1. Если A — M-сильно выпуклое множество, то для любых $a, b \in A$ имеем:

$$\{a, b\} \subset A \quad \rightarrow \quad \operatorname{conv}_M\{a, b\} \subseteq \operatorname{conv}_M A = A.$$

Это значит, что A — M-выпукло. □

Доказательство теоремы 2. Докажем сначала включение $\operatorname{int} b(A) \subseteq \operatorname{int} b(M)$. Возьмем некоторый $p \in \operatorname{int} b(A)$. Возьмем некоторые $a, b \in A$. Множество $C = \operatorname{conv}_M\{a, b\}$ содержится в A, значит, $p \in \operatorname{int} b(C)$. А так как C — пересечение транслятов M, то $\operatorname{int} b(C) = \operatorname{int} b(M)$.

Теперь докажем, что $\operatorname{int} b(M) \subseteq b(A)$. Предположим противное: найдется $p \in \operatorname{int} b(M)$ такой, что $p \notin b(A)$. Кроме того, по уже доказанному существует $p_0 \in \operatorname{int} b(M) \cap \operatorname{int} b(A)$. По лемме 4 найдутся такие $x_0 \in M$ и $a_0 \in A$, что

$$s(p_0, M) = \langle p_0, x_0 \rangle = m_M \quad s(p_0, A) = \langle p_0, a_0 \rangle.$$

Так как $p \notin b(A)$, то найдется такая последовательность точек $b_n \in A$, что $\langle p, b_n \rangle \to +\infty$ при $n \to \infty$. Обозначим $C_n = \operatorname{conv}_M\{a_0, b_n\}$.

Теперь, так как M порождающее, то для каждого C_n найдется C_n^* такое, что $C_n + C_n^* = M$. Зафиксируем некоторое $\varepsilon > 0$. По лемме 3, примененной к p_0, найдется такой вектор $c_n^* \in C_n^*$, что $s(p_0, C_n) + \langle p_0, c_n^* \rangle >$

$s(p_0, M) - \varepsilon$. При этом $C_n + c_n^* \subseteq M$, следовательно $a_0 + c_n^* \in M$, а так как $\langle p_0, a_0\rangle = s(p_0, A) \geq s(p_0, C_n)$ и $s(p_0, M) = \langle p_0, x_0\rangle = m_M$, то

$$\langle p_0, a_0 + c_n^*\rangle > m_M - \varepsilon.$$

$$M_{p_0, m_M - \varepsilon} = \{x \in M : \langle p_0, x\rangle \geq m_M - \varepsilon\},$$

то $a_0 + c_n^* \in M_{p_0, m_M - \varepsilon}$.

По лемме 1 множество $M_{p_0, m_M - \varepsilon}$ ограничено, значит, последовательность $\langle p, a_0 + c_n^*\rangle$ ограничена. Значит, последовательность $\langle p, c_n^*\rangle$ тоже ограничена.

Так как $C_n + c_n^* \subseteq M$, $b_n + c_n^* \in M$. Следовательно, последовательность $\langle p, b_n + c_n^*\rangle$ ограничена сверху, что влечет ограниченность сверху $\langle p, b_n\rangle$. Однако $\langle p, b_n\rangle \to +\infty$, что приводит к противоречию.

Так как $\operatorname{int} b(A) \subseteq \operatorname{int} b(M)$ и $\operatorname{int} b(M) \subseteq b(A)$, то $\operatorname{int} b(A) = \operatorname{int} b(M)$.

Так как M — порождающее множество, то A M-сильно выпукло тогда и только тогда, когда существует A^* такое, что $A + A^* = M$. Это, в свою очередь, равносильно тому, что разность опорных функций $s(p, M) - s(p, A)$ выпукла на $\operatorname{int} b(M) = \operatorname{int} b(A)$.

Докажем выпуклость разности $s(p, M) - s(p, A)$. Возьмем $p_1, p_2 \in \operatorname{int} b(A)$. По лемме 4 найдутся $a_1, a_2 \in A$ такие, что

$$s(p_1, A) = \langle p_1, a_1\rangle \quad s(p_2, A) = \langle p_2, a_2\rangle.$$

Обозначим $B = \operatorname{conv}_M\{a_1, a_2\}$. Так как A M-выпукло, то $B \subseteq A$. Следовательно,

$$s(p_1, B) \leq s(p_1, A) = \langle p_1, a_1\rangle,$$

но $a_1 \in B$, значит, $s(p_1, B) = s(p_1, A)$. Аналогично $s(p_2, B) = s(p_2, A)$.

Так как $B \subseteq A$, то для любого $p \in \operatorname{int} b(A)$

$$s(p, B) \leq s(p, A).$$

Так как B M-выпукло, то для любого $t \in [0, 1]$

$$s(tp_1 + (1-t)p_2, M) - s(tp_1 + (1-t)p_2, B) \leq t(s(p_1, M) - s(p_1, B)) + (1-t)(s(p_2, M) - s(p_2, B)).$$

Следовательно, для любого $t \in [0, 1]$

$$\begin{aligned} s(tp_1 + (1-t)p_2, M) - s(tp_1 + (1-t)p_2, A) \leq \\ \leq s(tp_1 + (1-t)p_2, M) - s(tp_1 + (1-t)p_2, B) \leq \\ \leq t(s(p_1, M) - s(p_1, B)) + (1-t)(s(p_2, M) - s(p_2, B)) = \\ = t(s(p_1, M) - s(p_1, A)) + (1-t)(s(p_2, M) - s(p_2, A)). \end{aligned}$$

То есть $s(p, M) - s(p, A)$ выпукла.

Осталось доказать, что если M ограничено, то $\operatorname{int} b(A) \neq \emptyset$. Докажем более сильное утверждение: A ограничено. Зафиксируем точку $a_0 \in A$. Для любой $b \in A$ по определению M-выпуклого множества $\operatorname{conv}_M\{a_0, b\} \subseteq A$. Это означает в частности, что найдется такое $t \in E$, что

$$a_0, b \in M + t.$$

Значит, $a_0 - t \in M$ и

$$b \in M + t = M + t - a_0 + a_0 \subseteq M - M + a_0,$$

где $M - M$ — сумма Минковского множеств M и $-M$. Отсюда следует, что $A \subseteq a_0 + M - M$. При этом ограниченность M влечет ограниченность $M - M$, значит, A тоже ограничено.

На этом теорема полностью доказана. □

Приведем пример M-выпуклого множества, которое не является M-сильно выпуклым. При этом множество M будет ограниченным, но не порождающим.

Пример. Пусть $E = \mathbb{R}^3$, S_0 — правильный тетраэдр. Рассмотрим четыре плоскости, каждая из которых параллельна одной из граней S_0 и делит соответствующую высоту S_0 пополам. Эти плоскости разбивают S_0 на октаэдр M и четыре меньших тетраэдра, один из которых обозначим A.

Ясно, что A не может быть M-сильно выпуклым, так как никакая трансляция M не покрывает A. Однако A является M-выпуклым. Докажем это.

Возьмем любые точки $a, b \in A$. Чтобы доказать, что $\mathrm{conv}_M\{a, b\} \subseteq A$, достаточно для любой грани F симплекса A найти такой транслят $M + t$, что $M + t \ni a, b$ и $M + t$ лежит по ту же сторону от F, что и A. Рассмотрим гомотетичный A симплекс A' минимального размера, содержащий a и b. Если обозначить грань A', параллельную F за ABC, а оставшуюся вершину за S, то с точностью до симметрий остается разобрать три случая:

1. $a = S$, $b \in ABC$;
2. $a = A$, $b \in SBC$;
3. $a \in AB$, $b \in SC$.

В каждом из этих случаев явно строится транслят $M + t$, содержащий a и b и лежащий по ту же сторону от ABC, что и A'.

4. Заключение

Автор благодарен Е.С. Половинкину и М.В. Балашову за обсуждения, следствием которых явились результаты, изложенные в этой статье. Автор благодарен В.Л. Дольникову за содержательное обсуждение и всестороннюю поддержку.

Список литературы

1. Danzer L., Grünbaum B., Klee V. Helly's theorem and its relatives //Convexity, Proc. of Symposia in Pure Math. Amer. Math. Soc. Providence, RI, 1963. V. 7. P. 101–180.
2. Данцер Л., Грюнбаум Б., Кли В. Теорема Хелли и ее применения. М.: Мир, 1968.
3. Половинкин Е.С. Сильно выпуклый анализ //Математический сборник. 1996. Т. 187, No. 2. С. 103–130.
4. Балашов М.В. Некоторые вопросы сильно выпуклого анализа : Дис. ... канд.физ.-мат. наук по спец. 01.01.09. М.: МФТИ, 1998.
5. Половинкин Е.С., Балашов М.В. M-сильно выпуклые подмножества и их порождающие множества //Математический сборник. 2000. Т. 191, No. 1. С. 27–64.
6. McMullen, P., Schneider R., Shepherd G.C. Monotypic polytopes and their intersection properties //Geom. Dedicata. 1974. V. 3. P. 99–129.
7. Geivaerts, M. Enkele eigenschappen van de relatie "homothetisch aanpasselijk" in de ruimte der konvexe lichamen //Med. Konink. Acad. Wetensch. België, 1972. V. 34. P. 3–19.
8. Рудин У. Функциональный анализ. М.: Мир, 1975.
9. Рокафеллар Р.Т. Выпуклый анализ. М.: Мир, 1973.
10. Bonnesen, T., Fenchel W. Theorie der konvexen Körper //Ergebn. d. Math. u. ihrer Grenzgeb. Berlin: Springer Verl., 1934. V. 8, No. 3. P. 77.

Модел. и анализ информ. систем. Т.11, №2 (2004) 37–38

УДК 681.3

Теорема об эпиморфизме для систем переходов

Белов Ю.А.[1]
Ярославский государственный университет
150 000, Ярославль, Советская, 14

получена 15 сентября 2004

Аннотация

Для систем с помеченными переходами (см. [1]) определяется понятие фактор-системы. Доказывается теорема об эпиморфизме, аналогичная подобной теореме для любых универсальных алгебр - [5]. Аналогом гомоморфизмов при этом являются корректные отображения, определенные в [4]. Приведенные конструкции являются обобщением корректных слияний позиций для сетей Петри ([2, 3]).

1. Определения

Система с помеченными переходами (labeled transitions system LTS - [1]) - это тройка $D = \langle S, L, T\rangle$, где S - произвольное абстрактное множество, называемое множеством состояний, L - множество меток (имен) переходов, $T \subseteq S \times L \times S$ - множество переходов. Элементы из T записываются в следующем виде: $s \xrightarrow{l} s'$, если $(s, l, s') \in T$ и читаются так: система D из состояния s под действием перехода с именем l перешла в состояние s'. Понятие бисимуляции (то есть возможности взаимного моделирования поведения двух систем) определяется следующим образом. Пусть имеются две системы $D = \langle S_1, L, T_1\rangle$ и $H = \langle S_2, L, T_2\rangle$ с одинаковым множеством L имен переходов. Бинарное отношение $R \subseteq S_1 \times S_2$ является **отношением бисимуляции**, если для любой пары $(s, q) \in R$ из того, что $s \xrightarrow{l} s'$ в D следует, что в H существует переход с той же меткой l вида $q \xrightarrow{l} q'$, при котором $(s', q') \in R$. Аналогично, если $q \xrightarrow{l} q'$ в H, то в D найдется переход $s \xrightarrow{l} s'$ с той же меткой l, при котором $(s', q') \in R$.

При исследовании поведения системы D с переходами обычно задается некоторое начальное состояние $s_0 \in S$ и изучается пара (D, s_0). Две системы $D = \langle S_1, L, T_1\rangle$ и $H = \langle S_2, L, T_2\rangle$ D и H с одинаковыми множествами переходов называются **бисимулярными при начальных состояниях** $s_{01} \in S_1$ и $s_{02} \in S_2$, если существует такое отношение бисимуляции R, что $(s_{01}, s_{02}) \in R$, что будет обозначаться так: $(D, s_{01}) \cong (H, s_{02})$. Состояния s_1 и s_2 одной системы D будем называть **бисимулярными**, если $(D, s1) \cong (D, s2)$. Короче это будем обозначать $s_1 \cong s_2$, если ясно, о какой системе идет речь.

Система $D = \langle S_1, L, T_1\rangle$ называется **изоморфной** системе $H = \langle S_2, L, T_2\rangle$, если существует биективное отображение α S_1 на S_2 такое, что $\forall s, s' \in S_1 s \xrightarrow{l} s'$ тогда и только тогда, когда $\alpha(s) \xrightarrow{l} \alpha(s')$.

Определим понятие **фактор-системы.** Пусть задана система $D = \langle S, L, T\rangle$ и на множестве состояний S определено некоторое отношение эквивалентности π. Тогда фактор-система D/π определяется следующим образом. Её множество состояний - фактор-множество $S/\pi = \bar{S}$, то есть множество классов эквивалентности по отношению π. Множество меток L - как в исходной системе D. Для тройки $\bar{s}, \bar{s}' \in \bar{S}, l \in L$ считаем, что $\bar{s} \xrightarrow{l} \bar{s}'$ тогда и только тогда, когда существуют такие состояния $q, r \in S$, что $\bar{q} = \bar{s}, \bar{r} = \bar{s}'$ и $q \xrightarrow{l} r$ в D.

Напомним некоторые свойства отображений состояний одной системы в состояния другой системы ([4]). Пусть заданы две системы с переходами - $D = \langle S_1, L, T_1\rangle$ и $H = \langle S_2, L, T_2\rangle$. Будем рассматривать однозначное сюръективное отображение $f : S_1 \to S_2$. Говорим,что f имеет **прямое свойство переноса**, если из того, что в D существует переход $s \xrightarrow{l} s'$ следует, что в H существует переход $f(s) \xrightarrow{l} f(s')$ с той же меткой l. Говорим, что f имеет **обратное свойство переноса**, если из того, что в H имеется переход $f(s) \xrightarrow{l} f(s')$ для некоторых $s, s' \in S_1$ следует, что существуют такие состояния $\tilde{s}, \tilde{s}'$, что: $f(\tilde{s}) = f(s)$, $f(\tilde{s}') = f(s')$ и существует переход $\tilde{s} \xrightarrow{l} \tilde{s}'$ в D.

Отметим, что по определению фактор-системы **естественное отображение** $S \to \bar{S}$ обладает прямым и обратным свойствами переноса. Со свойствами переноса тесно связано понятие корректного отображения систем. Отображение f называется **корректным**, если для любого $s \in S_1$ $(D, s) \cong (H, f(s))$ ([2, 4]).

[1]Работа выполнена при поддержке РФФИ (грант 03-01-00804-а)

2. Теорема

Предлагаемое утверждение является аналогом соответствующей теоремы для алгебраических структур, однако здесь её роль гораздо менее важна.

Теорема Пусть $D = \langle S_1, L, T_1 \rangle$ и $H = \langle S_2, L, T_2 \rangle$ - две системы переходов, $f : S_1 \to S_2$ - сюръекция. Тогда если f имеет прямое и обратное свойства переноса, то D/f и H изоморфны и f является суперпозицией естественного отображения D на D/f и изоморфизма между D/f и H.

Доказательство. Для доказательства отметим, что между S_2 и классами S_1/f имеется очевидная биекция: каждому элементу из S_2 соответствует его полный прообраз, и, наоборот, полный класс из S_1/f составляют элементы из S_1, отображающиеся с помощью f в один элемент из S_2. Свойства переноса, имеющиеся для f и естественного отображения, обеспечивают (как легко проверить) изоморфность •

Отметим, что если, кроме того, прообразы всех элементов из S_2 при отображении f состоят из бисимулярных состояний, то f и естественное отображение - корректны.

Список литературы

1. Finkel A. Reduction and covering of infinite reachibility trees. Information and Computation. V.82(2). 1990. P.144-179.

2. Сидорова Н.С. Бисимуляционно-эквивалентные преобразования сетей Петри./Яросл. Гос. ун-т, Ярославль, 1998. Препринт №1. 52 с.

3. Schnoeblin Ph., Sidorova N. Bisimulation and reduction of Petri nets. Proc. 21^{th} Int. Conf. Appl. and Theory of Petri Nets. Aarhus, Denmark, June 2000.

4. Белов Ю.А. Корректные отображения систем с переходами // Моделирование и анализ информационных систем. 2001. Том 8, №1. С. 47-49

5. Курош А.Г. Лекции по общей алгебре М.: Физ-мат. лит. 1962