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

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ

Том 11 №1 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.

СОДЕРЖАНИЕ

Моделирование и анализ информационных систем. Т.11, №1. 2004

Исследование системы уравнений с запаздыванием, моделирующей сальтаторное проведение возбуждения
Ануфриенко С.Е., Майоров В.В., Мышкин И.Ю., Громов С.А. 3

О разрешимости задачи проверки модели для автоматной логики и вполне структурированных систем переходов автоматного типа
Кузьмин Е.В. 8

Распространение возбуждения по сети импульсных нейронов, организованных в кольцо
Лагутина Н.С. 18

Асимптотическое поведение наилучших кусочно-полиномиальных приближений в пространствах L_p $(0 < p < 1)$
Морозов А.Н. 24

Анализ поведения решения одного дифференциального уравнения с запаздывающим аргументом
Кубышкин Е.П. 28

Оценка дисперсии и робастность обобщенного γ-критерия
Янкевич А.П. 32

Разработка диспетчера подключений к базам данных в .NET Framework
Майоров А.В. 39

Корректор А.А.Аладьева
Подписано в печать 10.04.2004. Формат 60x84$^1/_8$. Печать офсетная.
Усл.печ.л. 8,74. Уч.-изд.л. 4,95. Тираж 100 экз. 040/04

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

Модел. и анализ информ. систем. Т.11, №1 (2004) 3–7

УДК 541.1+612.82

Исследование системы уравнений с запаздыванием, моделирующей сальтаторное проведение возбуждения

Ануфриенко С.Е., Майоров В.В., Мышкин И.Ю., Громов С.А.
Ярославский государственный университет
150 000, Ярославль, Советская, 14

получена 3 июня 2003

В работе рассматривается модель импульсного проведения возбуждения по миелинизированному аксону, которая представляет собой цепочку из пороговых нейронов (перехватов Ранвье), связанных посредством "изолированных" участков. Получены формулы, описывающие динамику мембранных потенциалов всех элементов системы, рассчитаны временные рассогласования между спайками перехватов Ранвье.

Введение. Термин "сальтаторное проведение" был введен в 20-х годах двадцатого века [1] при изучении химической реакции в железной проволоке, погруженной в концентрированную кислоту. Проволоку покрывали стеклянными трубками, расположенными на некотором расстоянии друг от друга. При этом волна химической реакции распространялась вдоль проволоки от одного открытого участка к другому, "перескакивая" (saltare–прыгать) через области, покрытые трубками. Известно [2] — [3], что аксон покрыт липидным слоем, называемым миелиновой оболочкой. В этой оболочке имеются расположенные на расстоянии порядка 2 мм друг от друга небольшие разрывы — перехваты Ранвье, протяженность которых около 1 мк. Миелиновая оболочка обладает высоким электрическим сопротивлением, поэтому по аксону импульсы распространяются скачкообразно от перехвата к перехвату, при этом все перехваты проводят импульсы одинаковой величины — принцип "все или ничего" [2]. Роль миелиновой оболочки и ее разрывов (перехватов Ранвье) в электрическом возбуждении нервного волокна продемонстрирована экспериментально [4].

Модель порогового нейрона предложена в работах [5] — [6].

Модель сальтаторного проведения возбуждения. Мембранные потенциалы перехватов Ранвье $u_i(t) \geq 0$ и миелинизированных участков $v_i(t) \geq 0$ будем отсчитывать от уровня максимальной гиперполяризации. Из анализа ионных потоков и законов Кирхгофа вытекает следующая система дифференциальных уравнений:

$$\dot{u}_0 = \lambda\left[-1 - f_{Na}(u_0) + f_K(u_0(t-1))\right]u_0 + \varepsilon + \lambda e^{-\lambda\sigma}(v_1 - u_0), \tag{1}$$

$$\dot{u}_i = \lambda\left[-1 - f_{Na}(u_i) + f_K(u_i(t-1))\right]u_i + \varepsilon + \lambda e^{-\lambda\sigma}(v_i - 2u_i + v_{i+1}), \tag{2}$$

$$\dot{v}_i = \lambda(u_{i-1} - 2v_i + u_i), i = 1, ..., N, v_{N+1}(t) \equiv u_N(t). \tag{3}$$

Здесь параметр $\lambda \gg 1$ отражает высокую скорость протекания электрических процессов, параметр $0 < \varepsilon \ll 1$ учитывает токи утечки, проходящие через мембраны перехватов. Положительные достаточно гладкие функции $f_{Na}(u)$ и $f_K(u)$ монотонно убывают к нулю при $u \to \infty$ быстрее, чем $O(u^{-1})$. Они описывают состояние натриевых и калиевых каналов мембран перехватов.
Параметры $\alpha = 1 + f_{Na}(0) - f_K(0) > 0$, $\alpha_1 = f_K(0) - 1 > 1$, $0 < \sigma < \alpha_1$. Число $f_K(0) - f_{Na}(1) - 1 > 0$ характеризует пороговое значение: начало спайка i-го перехвата условно свяжем с моментом времени t_s, таким что $u_i(t_s) = 1$, $u_i(t) < 1$ при $t_s - 1 < t < t_s$. Токи утечки через миелиновые оболочки не учитываются.

Система описывает последовательность перехватов Ранвье, связанных между собой посредством миелинизированных участков. Отметим, что система уравнений имеет устойчивое состояние равновесия

$$u_i = v_i = u_* \approx \frac{\varepsilon}{\lambda\alpha}.$$

Проанализируем систему при $\lambda \to \infty$. Стимулировать возбуждение нулевого перехвата будем с помощью задания для него специальных начальных условий. Класс начальных функций для нулевого перехвата состоит из непрерывных на отрезке $s \in [-1, 0]$ функций $\varphi(s)$, удовлетворяющих условиям: $\varphi(0) = 1$ и $0 \le \varphi(s) \le max(e^{\lambda\alpha s/2}, 1/\lambda)$. Обозначим этот класс функций S. Для остальных перехватов будем считать, что $u_i(s) = u_*$ при $s \in [-1, 0]$. Для всех миелинизированных участков считаем, что $v_i(0) = u_*$.

Динамика мембранного потенциала порогового нейрона. Рассмотрим отдельно уравнение (1), учитывать воздействие на нулевой перехват со стороны первого миелинизированного участка пока не будем. Начало и окончание спайка условно свяжем с моментами времени, когда $u_0(t)$ пересекает единичное значение соответственно с положительной и отрицательной скоростью. Уравнение имеет вид:

$$\begin{aligned} \dot{u}_0 &= \lambda\left[-1 - f_{Na}(u_0) + f_K(u_0(t-1))\right]u_0 + \varepsilon, \\ u_0(s) &= \varphi(s) \qquad \text{при } s \in [-1, 0]. \end{aligned} \tag{4}$$

Поскольку $u_0(0) = \varphi(0) = 1$ и $u_0(-1) = o(1)$, где $o(1)$ — слагаемые, которые стремятся к нулю при $\lambda \to \infty$, имеем:

$$\dot{u}_0(0) = \lambda\left[-1 - f_{Na}(1) + f_K(0) + o(1)\right] + \varepsilon \gg 1.$$

Следовательно, $t = 0$ является точкой возрастания функции $u_0(t)$.
Рассмотрим помежуток $t \in [\delta, 1-\delta]$, где $0 < \delta \ll 1$ произвольно малое фиксированное число. Имеем: $u_0(t-1) = o(1)$, $u_0(t) \gg 1$ и $\varepsilon/(\lambda u_0(t)) = o(1)$. Уравнение (4) примет вид:

$$\begin{aligned} \dot{u}_0 &= \lambda\left[\alpha_1 + o(1)\right]u_0, \\ u_0(0) &= 1. \end{aligned}$$

Его решение:

$$u_0(t) = e^{\lambda\alpha_1(t+o(1))}.$$

На следующем промежутке $t \in [1+\delta, t_1 - \delta]$, где t_1 определяется из условия $u_0(t_1) = 1$, $u_0(t) \gg 1$ и $u_0(t-1) \gg 1$, следовательно, уравнение (4) примет вид:

$$\begin{aligned} \dot{u}_0 &= \lambda\left[-1 + o(1)\right]u_0, \\ u_0(1) &= e^{\lambda\alpha_1(1+o(1))}. \end{aligned}$$

Его решение:

$$u_0(t) = e^{\lambda(\alpha_1 - (t-1) + o(1))}.$$

Вычислим значение t_1:

$$\begin{aligned} u_0(t_1) &= e^{\lambda(\alpha_1 - (t_1-1) + o(1))} = 1, \\ t_1 &= \alpha_1 + 1 + o(1). \end{aligned}$$

Поскольку $\alpha_1 > 1$, то длительность нисходящего участка больше, чем восходящего.
На следующем промежутке $t \in [\alpha_1 + 1 + \delta, \alpha_1 + 2 - \delta]$ имеем: $u_0(t) \ll 1$ и $u_0(t-1) \gg 1$. Уравнение (4) примет вид:

$$\begin{aligned} \dot{u}_0 &= \lambda\left[-1 - f_{Na}(0) + o(1)\right]u_0 + \varepsilon, \\ u_0(\alpha_1 + 1) &= 1 + o(1). \end{aligned}$$

Выделяя главное слагаемое, получим:

$$u_0(t) = \frac{\varepsilon + o(1)}{\lambda(\alpha + f_K(0))}.$$

При $t \ge \alpha_1 + 2 + \delta$ имеем: $u_0(t) \ll 1$ и $u_0(t-1) \ll 1$. Уравнение (4) примет вид:

$$\begin{aligned} \dot{u}_0 &= \lambda\left[\alpha + o(1)\right]u_0 + \varepsilon, \\ u_0(\alpha_1 + 2) &= \frac{\varepsilon + o(1)}{\lambda(\alpha + f_K(0))}. \end{aligned}$$

Выделяя главное слагаемое, получим:

$$u_0(t) = \frac{\varepsilon + o(1)}{\lambda\alpha}.$$

Таким образом, при $t \ge \alpha_1 + 2 + \delta$ получаем: $u_0 \approx u_*$ до тех пор, пока нулевой перехват не сгенерирует новый спайк, который может быть вызван только внешним сигналом. Из явного вида уравнения (1) следует: если $v_1(t) = o(u_0(t)e^{\lambda\sigma})$ (ниже будет показано, что это условие выполняется для любых t), то спайка не произойдет. Аналогично выписываются условия возбуждения других перехватов.

Динамика мембранного потенциала миелинизированного слоя. Перейдем к анализу уравнения (3) при $i = 1$:

$$\dot{v}_1 = \lambda(u_0 - 2v_1 + u_1). \tag{5}$$

Рассмотрим промежуток $t \in [\delta, \tau - \delta]$, где τ — первый корень уравнения $u_1(\tau) = 1$. Скоро мы покажем, что $\tau < 1$, поэтому $u_0(t) \gg 1$ и $u_1(t) \ll 1$. Следовательно, уравнение (5) примет вид:

$$\dot{v}_1 = -2\lambda v_1 + \lambda u_0(1 + o(1)).$$

Учитывая явный вид функции $u_0(t)$, перепишем уравнение:

$$\dot{v}_1 = -2\lambda v_1 + \lambda e^{\lambda\alpha_1(t+o(1))},$$

$$v_1(0) = u_*.$$

Выделяя главное слагаемое, получим:

$$v_1(t) = e^{\lambda\alpha_1(t+o(1))}.$$

Для нахождения значения τ, рассмотрим уравнение (2) при $i = 1$.

$$\dot{u}_1 = \lambda\left[-1 - f_{Na}(u_1) + f_K(u_1(t-1))\right]u_1 + \varepsilon + \lambda e^{-\lambda\sigma}(v_1 - 2u_1 + v_2), \tag{6}$$

Перепишем его, учитывая, что на промежутке $t \in [\delta, \tau - \delta]$
$u_1(t) = o(1)$ и $v_2(t) = o(1)$:

$$\dot{u}_1 = -\lambda\alpha\left[1 + o(1)\right]u_1 + \varepsilon + \lambda e^{\lambda(\alpha_1 t - \sigma + o(1))},$$

$$u_1(0) = u_*.$$

Его решение:

$$u_1(t) = \frac{\varepsilon}{\lambda\alpha} + \frac{1}{\alpha + \alpha_1}\left(e^{\lambda(\alpha_1 t - \sigma + o(1))} - e^{-\lambda(\alpha t + \sigma + o(1))}\right).$$

Выделяя главное слагаемое, найдем τ из условия $u_1(\tau) = 1$:

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

Отметим, что при $s \in [-1, 0]$, имеем: $u_1(\tau + s) = \varphi(s) \in S$.
Рассмотрим промежуток $t \in [\tau, t_2]$, где $t_2 < 1$ и $u_1(t) < u_0(t)$ для всех $t < t_2$. Имеем:

$$v_1(t) = e^{\lambda\alpha_1(t+o(1))}.$$

Последнее означает, что при $t < t_2$ выполнено: $v_1(t) \approx u_0(t)$.
Точно так же можно показать, что $v_2(t) \approx u_1(t)$ с точностью до слагаемых $o(1)$. По теореме сравнения из уравнения (6) получаем: $u_1(t) > u_0(t - \tau)$ при $t > \tau$. Учитывая, что $\tau < \sigma$, имеем:

$$v_1(t) - u_1(t) < e^{\lambda\alpha_1(t+o(1))} - e^{\lambda\alpha_1(t-\tau+o(1))} = o(u_1(t)e^{\lambda\sigma}).$$

Поскольку t_2 произвольно, то данная оценка справедлива на всем промежутке $t \in [\tau, 1 - \delta]$. Приведенные рассуждения показывают, что на данном промежутке внешнее воздействие на первый перехват не оказывается. Дальнейший анализ свидетельствует об отсутствии внешнего влияния на u_1 при всех $t > \tau$. Следовательно, при $t > \tau$ справедливо соотношение: $u_1(t) \approx u_0(t - \tau)$.

Продолжим исследование уравнения (5). На промежутке $t \in [\tau, 1 - \delta]$ оно примет вид:

$$\dot{v}_1 = -2\lambda v_1 + \lambda e^{\lambda\alpha_1(t+o(1))}.$$

Таким образом, на всем промежутке $t \in [\delta, 1 - \delta]$ имеем:

$$v_1(t) = e^{\lambda\alpha_1(t+o(1))}.$$

На следующем промежутке $t \in [1 + \delta, 1 + \tau - \delta]$ уравнение (5) примет вид:

$$\dot{v}_1 = -2\lambda v_1 + \lambda e^{\lambda(\alpha_1 - (t-1) + o(1))} + \lambda e^{\lambda\alpha_1(t-\tau+o(1))}.$$

Здесь следует рассмотреть два случая.
Случай первый: $\alpha_1 - (t-1) > \alpha_1(t-\tau)$, т.е.

$$t < 1 + \frac{\alpha_1 \tau}{\alpha_1 + 1} = t_* < 1 + \tau.$$

Уравнение (5) перепишется в виде:

$$\dot{v}_1 = -2\lambda v_1 + \lambda e^{\lambda(\alpha_1 - (t-1) + o(1))},$$

$$v_1(1) = e^{\lambda \alpha_1 (1 + o(1))}.$$

Его решение:

$$v_1(t) = e^{\lambda(\alpha_1 - (t-1) + o(1))}.$$

Случай второй: $t \in [t_* + \delta, 1 + \tau - \delta]$.
Здесь уравнение (5) примет вид:

$$\dot{v}_1 = -2\lambda v_1 + \lambda e^{\lambda \alpha_1 (t - \tau + o(1))},$$

$$v_1(t_*) = e^{\lambda(\alpha_1 - \frac{\alpha_1 \tau}{\alpha_1 + 1} + o(1))}.$$

Его решение:

$$v_1(t) = e^{\lambda \alpha_1 (t - \tau + o(1))}.$$

На промежутке $t \in [1 + \tau + \delta, 1 + \alpha_1 + \tau - \delta]$ уравнение примет (5) вид:

$$\dot{v}_1 = -2\lambda v_1 + \lambda e^{\lambda(\alpha_1 - (t - \tau - 1) + o(1))},$$

$$v_1(1 + \tau) = e^{\lambda \alpha_1 (1 + o(1))}.$$

Его решение:

$$v_1(t) = e^{\lambda(\alpha_1 - (t - \tau - 1) + o(1))}.$$

При $t > 1 + \alpha_1 + \tau$ имеем: $v_1 \approx 0.5(u_0 + u_1)$, поэтому при $t > 2 + \alpha_1 + \tau$ получаем: $v_1 \approx u_*$.

Результаты исследования показывают, что $v_1(t) = o(u_0(t) e^{\lambda \sigma})$ при $t > 0$, и $v_1(t) = o(u_1(t) e^{\lambda \sigma})$ при $t > \tau$. Поэтому миелинизированные участки не могут повлиять на перехват Ранвье, когда тот генерирует спайк, и в течение некоторого времени после спайка, пока перехват не достигнет состояния равновесия ($u_i(t) = u_* + o(1)$). Этот промежуток времени называется периодом рефрактерности, его продолжительность: $T_R = \alpha_1 + 2 + o(1)$. В указанный период уравнения (1) и (2) могут быть проинтегрированы независимо от других. Описанное явление имеет простой биологический смысл. Известно [2], что миелинизированное волокно не может проводить импульсный сигнал большой частоты, потому что перехваты должны "восстановиться".

Из приведенных формул следует, что $v_1(t) \approx u_0(t)$ при $0 < t < t_*$ и $v_1(t) \approx u_1(t)$ при $t > t_*$. Тем самым, мембранный потенциал миелинизированного участка имеет две точки максимума $t^1_{max} = 1 + o(1)$, $t^2_{max} = 1 + \tau + o(1)$ и находящуюся между ними точку локального минимума $t_{min} = t_* + o(1)$. Это согласуется с биологическими данными [2].

Основной результат. Рассуждения, аналогичные проведенным выше, показывают, что спайк i–ого перехвата Ранвье начинается в момент времени $t = i\tau + o(1)$, а мембранный потенциал i–ого миелинизированного участка получается временным сдвигом потенциала первого участка $v_i(t) \approx v_1(t - (i-1)\tau)$ при $t > (i-1)\tau$.

Таким образом, при возбуждении любого перехвата Ранвье по цепочке перехватов будет распространяться волна импульсов (спайков). Если возбуждается перехват в середине цепочки, то волна распространяется в двух направлениях. Возбуждение крайнего перехвата порождает волну импульсов, распространяющуюся в сторону возрастания (или убывания) номеров перехватов.

Система (1) — (3) исследовалась численно при следующих значениях параметров:

$$\lambda = 7, \qquad \varepsilon = 0.05, \qquad \sigma = 1.011, \qquad f_{Na}(u) = R_1 e^{-u^2}, \qquad f_K(u) = R_2 e^{-u^2}, \qquad R_1 = 1.2, \qquad R_2 = 2.1.$$

Результаты счета приведены на рисунке (1). Слева приведен график потенциала миелинизированного участка, находящегося между перехватами с номерами 3 и 4. Он имеет две точки максимума, которые совпадают с максимумами соответствующих перехватов, и находящуюся между ними точку минимума.

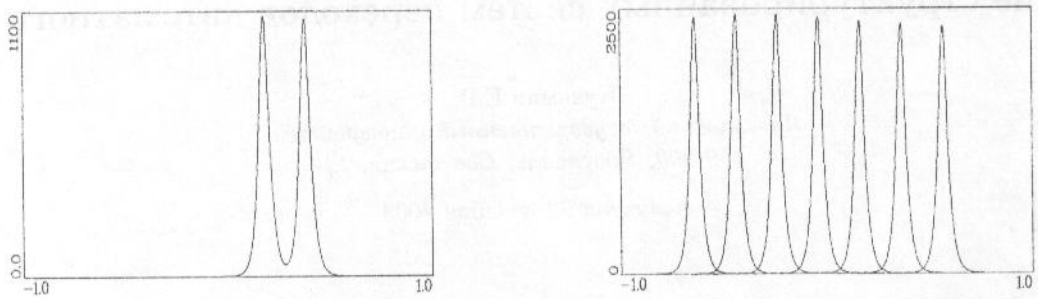

Рис. 1. Графики потенциалов миелинизированного слоя (слева) и перехватов Ранвье (справа)

Справа изображены мембранные потенциалы перехватов при N=6. Спайки перехватов состоят из участков быстрого роста и почти такого же быстрого убывания мембранного потенциала. Продолжительность нисходящего участка немногим более восходящего ($\alpha_1 = 1.1$). Высота спайков близка к расчетной ($\lambda\alpha_1$). Временной промежуток между спайками соседних перехватов чуть больше 1. Это объясняется тем, что значение параметра α_1 близко к 1, а параметра σ — к α_1.

Литература

[1] Lillie R.S. Factors affecting trasmission and recover in the passive iron nerve model //J. Gen. Physiol. V.7. P. 473-507.

[2] Тасаки И. Нервное возбуждение. М.: Мир, 1971.

[3] Шаде Дж., Форд Д. Основы неврологии. М.: Мир, 1976.

[4] Kato G. The Microphysiology of Nerve. Tokyo : Maruzen, 1934.

[5] Майоров В.В., Мышкин И.Ю. Математическое моделирование нейронов сети на основе уравнений с запаздыванием // Математическое моделирование. 1990. Т. 2, №11. С. 64 - 76.

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

УДК 681.3

О разрешимости задачи проверки модели для автоматной логики и вполне структурированных систем переходов автоматного типа

Кузьмин Е.В.
Ярославский государственный университет
150 000, Ярославль, Советская, 14

получена 22 декабря 2003

В работе исследуются вопросы разрешимости задачи проверки модели для автоматной логики и вполне структурированных систем переходов автоматного типа, обладающих свойствами верхней и нижней совместимости.

1. Введение

Одной из важных задач при проектировании программного обеспечения является проверка корректности параллельных систем. Под корректностью понимается полное соответствие системы задачам, для которых она создаётся. Корректность определяется в соответствии с формальной спецификацией, описывающей желаемое поведение системы. Процесс проверки соответствия системы требованиям, заданным в спецификации, называется верификацией.

Проверка модели (model checking) – один из подходов к решению проблемы верификации. В качестве языков спецификации для выражения свойств систем при этом подходе используются темпоральные логики. Задача проверки модели состоит в определении выполнимости для системы, заданной формальным образом, свойства, записанного в виде формулы темпоральной логики.

Проверка модели широко используется при верификации систем с конечным числом состояний, но эта автоматическая техника может быть применена для некоторых классов систем с бесконечным числом состояний и подмножеств темпоральных логик.

Существует большое количество формальных моделей, которые используются для описания параллельных систем. Многие из этих моделей могут быть рассмотрены как вполне структурированные системы переходов [3].

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.

В данной работе рассматривается класс вполне структурированных систем переходов автоматного типа [10]. Доказывается, что этот класс совпадает с классом вполне структурированных систем переходов, удовлетворяющих свойствам верхней и нижней совместимости, примером которых может служить формализм взаимодействующих раскрашивающих автоматов [4].

Исследуется разрешимость задачи проверки модели для данного класса систем переходов и различных подмножеств автоматной логики, в рамках которой элементарные высказывания интерпретируются верхними и нижними конусами.

Автоматная логика AL является одной из самых выразительных программных логик. Например, логика AL более мощная по выразительной силе, чем такие широко используемые логики, как CTL и LTL, поскольку формулы этих логик могут быть адекватно интерпретированы в терминах автоматов.

Логика AL определяется в духе расширенной темпоральной логики ETL [8, 9]. Ранее автоматная логика была представлена в работе [1], где исследовался вопрос разрешимости свойств, заданных автоматной логикой, для сетей Петри с потерями.

В рамках AL возможно построение формул, выражающих темпоральные свойства систем с учётом действий при переходе из одного состояния в другое и элементарных высказываний над множеством состояний системы. Логика позволяет работать с конечными и бесконечными исполнениями системы переходов, а также учитывать временное ветвление, т.е. позволяет строить формулы, снабжённые кванторами всеобщности и существования.

Автоматная логика позволяет рассматривать разрешимость некоторых классов темпоральных свойств, не охваченных ранее [10]. Более того, поскольку вполне структурированная система переходов автоматного типа по своей управляющей структуре является конечным автоматом, применение автоматной логики

для задания свойств, а также исследование вопросов разрешимости этих свойств представляется весьма интересным и логичным.

В работе доказывается разрешимость задачи проверки модели для вполне структурированных систем переходов автоматного типа и фрагментов автоматной логики $AL_f(P_{DC})$, $AL_{\omega c}(P_{UC})$ и $AL_f(P_{UC})$.

Кроме того, с использованием теории счётчиковых машин показана и общая неразрешимость задачи проверки модели для ряда подмножеств автоматной логики и вполне структурированных систем переходов автоматного типа.

2. Основные понятия и определения

2.1. Системы переходов

Бинарное отношение $\leq$ называется отношением *частичного порядка*, если оно рефлексивно, транзитивно и антисимметрично. Антисимметричность означает, что $(x \leq y) \wedge (y \leq x) \rightarrow (x = y)$. Если отношение только рефлексивно и транзитивно, то оно называется отношением *квазипорядка*. Если $x \leq y$ и $x \neq y$, то пишут $x < y$ и говорят, что x строго меньше y. Очевидно, что строгий порядок есть транзитивное и иррефлексивное отношение.

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* (соот. *нижним конусом*) в X называется подмножество $I \subseteq X$, такое что для $x \in I$, $y \in X$ и $x \leq y$ (соот. $x \geq y$), 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 $I = \cup_{x \in min(I)} \uparrow x$.

Утверждение 1 *Если $\leq$ – вполне упорядочиваемый квазипорядок на множестве X, то всякий верхний конус I имеет конечный базис.*

Утверждение 2 *Если $\leq$ – вполне упорядочиваемый квазипорядок на множестве X, то всякая бесконечно возрастающая (по отношению вложения множеств) последовательность $I_0 \subseteq I_1 \subseteq I_2 \subseteq \ldots$ верхних конусов стабилизируется, т.е. найдётся такое $k \in \mathbf{N}$, что $I_k = I_{k+1} = I_{k+2} = \ldots$.*

Доказательства этих утверждений см., например, в [3].

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 и бесконечным числом состояний.

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.

Определение 1 *Вполне структурированной системой переходов (с совместимостью по возрастанию) называется система переходов $LTS = (S, T, \rightarrow, s_0)$, дополненная отношением квазипорядка $\leq \subseteq S \times S$, удовлетворяющим следующим условиям:*

1. *отношение $\leq$ является вполне упорядочиваемым квазипорядком;*
2. *квазипорядок $\leq$ совместим 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 $s_1' \leq s_2'$;*
3. *для каждого состояния $s \in S$ и пометки $t \in T$ можно вычислить предбазис, т.е. вычислимо множество $min(Pred_t(\uparrow s))$.*

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

Определение 2 Вполне структурированной системой переходов *с совместимостью по убыванию называется система переходов* $LTS = (S, T, \rightarrow, s_0)$, *дополненная отношением квазипорядка* $\leq \subseteq S \times S$, *удовлетворяющим следующим условиям:*

1. *отношение* $\leq$ *является вполне упорядочиваемым квазипорядком;*
2. *квазипорядок* $\leq$ *совместим по убыванию с отношением переходов* $\rightarrow$, *а именно для любых состояний* $s_1 \leq s_2$ *и перехода* $s_2 \xrightarrow{t} s_2'$ *существует переход* $s_1 \xrightarrow{t} s_1'$, *такой что* $s_1' \leq s_2'$.

Для анализа некоторых свойств вполне структурированных систем переходов используется конструкция *покрывающего дерева* [2].

Определение 3 Покрывающим деревом *вполне структурированной системы переходов* $(LTS, \leq)$ *для состояния* $s_0 \in S$ *называется конечный ориентированный граф (дерево), такой что*

1. *Вершины дерева помечены состояниями системы* LTS;
2. *Каждая вершина объявлена либо* живой, *либо* мёртвой;
3. *Корню дерева приписана пометка* s_0, *и эта вершина объявлена живой;*
4. *Мёртвые вершины не имеют потомков;*
5. *Живая вершина с пометкой* s *имеет по одному потомку, помеченному* s', *для каждого состояния* $s' \in Succ(s)$;
6. *Если на пути от корня до некоторой вершины с пометкой* s' *встречается вершина с пометкой* s *такой, что* $s \leq s'$, *то говорят, что* s' покрывает s, *и вершина* s' *объявляется мёртвой; в противном случае* s' – *живая вершина.*

Листья в покрывающем дереве помечены финальными или покрывающими состояниями. Для вполне структурированных систем переходов частичный порядок $\leq$ является вполне упорядочиваемым. Благодаря этому все пути в покрывающем дереве конечны, так как всякий бесконечный путь должен был бы содержать покрывающую вершину. В силу леммы Кёнига о конечных деревьях, если покрывающее дерево не имеет бесконечных ветвлений, то оно конечно.

Более того, очевидно, имеет место следующее

Утверждение 3 *Если отношение порядка* $\leq$ *разрешимо (т.е. существует эффективная процедура проверки истинности* $s \leq s'$ *для любых* s *и* s'*) и отображение* $Succ$ *вычислимо (т.е. существует эффективная процедура вычисления множества* $Succ(s)$ *для любого состояния* s*), то покрывающее дерево для вполне структурированной системы переходов может быть эффективно построено.*

Для построения покрывающего дерева не требуется совместимость между отношениями упорядоченности $\leq$ и перехода $\rightarrow$. Однако именно при выполнении условия совместимости покрывающее дерево содержит полезную информацию о свойствах поведения системы.

2.2. Счётчиковые машины

Машина Минского (или *счётчиковая машина*) M – это набор $(\{q_0, \ldots, q_n\}, \{x_1, \ldots, x_m\}, \{\delta_0, \ldots, \delta_{n-1}\})$, где x_i – счётчик, q_i – состояние, q_0 – начальное состояние, q_n – финальное (заключительное) состояние, δ_i – правило переходов для q_i $(0 \leq i \leq n-1)$.

Состояния q_i, $0 \leq i \leq n-1$, подразделяются на два типа. Состояния первого типа имеют правила переходов вида $(1 \leq j \leq m, 0 \leq k \leq n)$:

$$\delta_i : \mathtt{x_j := x_j + 1;\ goto\ q_k}.$$

Для состояний второго типа имеем $(0 \leq k' \leq n)$:

$$\delta_i : \mathtt{if\ x_j > 0\ then\ (x_j := x_j - 1;\ goto\ q_k)\ else\ goto\ q_{k'}}.$$

Конфигурация машины Минского – это набор $(q_i, c_1, \ldots, c_m)$, где q_i – состояние машины, $c_1, \ldots, c_m$ – натуральные числа, являющиеся значениями соответствующих счётчиков.

Исполнение машины – это последовательность конфигураций $s_0 \to s_1 \to s_2 \to \ldots$, начинающаяся в начальной конфигурации s_0 и индуктивно определяемая в соответствии с правилами переходов. Необходимо отметить, что исполнение детерминировано, т.к. каждое состояние имеет не более одного правила переходов. Машина Минского останавливается, если исполнение содержит конфигурацию с состоянием q_n, т.е. достигает финального состояния. Поскольку машина Минского уже всего с двумя счётчиками может моделировать машину Тьюринга, проблема (останова) достижения финального состояния из некоторой начальной конфигурации (q_0, c_1, c_2) для 2-х счётчиковой машины является неразрешимой [11].

3. Системы переходов автоматного типа

Определение 4 *Вполне структурированная система переходов автоматного типа WSA – это вполне структурированная система переходов $WSTS = (S, T, \to, s_0)$, дополненная отношением вполне упорядочиваемого квазипорядка $\leq \subseteq S \times S$, где $S = Q \times D$, Q – конечное множество управляющих состояний, D – конечное или бесконечное множество значений некоторых характеристик, $s_0 \in S$ – начальное состояние, $\to \subseteq S \times T \times S$ такое, что $\exists \to_\bullet \subseteq Q \times T \times Q$: для любых состояний (q, d), (q', d') и некоторой метки перехода t переход $(q, d) \xrightarrow{t} (q', d')$ существует тогда и только тогда, когда существует переход $q \xrightarrow{t}_\bullet q'$; отношение $\leq$ определяется так, что $(q, d) \leq (q', d') \iff q = q'$ и $d \leq d'$.*

Утверждение 4 *Система переходов автоматного типа $(WSA, \leq)$ удовлетворяет условию совместимости по убыванию отношения $\leq$ с отношением переходов $\to$, т.е. для любых состояний $s_1 \leq s_1'$ и перехода $s_1' \xrightarrow{t} s_2'$ существует переход $s_1 \xrightarrow{t} s_2$, такой что $s_2 \leq s_2'$.*

Доказательство. Возьмём некоторые состояния $s_1 = (q_1, d_1)$, $s_1' = (q_1', d_1')$ системы $(WSA, \leq)$, такие что $s_1 \leq s_1'$, т.е. $q_1 = q_1'$ и $d_1 \leq d_1'$. По определению $(WSA, \leq)$ переходы $s_1 \xrightarrow{t} s_2$ и $s_1' \xrightarrow{t} s_2'$, где $s_2 = (q_2, d_2)$, $s_2' = (q_2, d_2')$, возможны тогда и только тогда, когда существует переход $q_1 \xrightarrow{t}_\bullet q_2$. Из вполне структурированности $(WSA, \leq)$ следует, что $s_2 \leq s_2'$, т.е. $q_2 = q_2'$ и $d_2 \leq d_2'$. □

Утверждение 5 *Любая вполне структурированная система переходов с совместимостью по возрастанию и убыванию является вполне структурированной системой переходов автоматного типа.*

Доказательство. Пусть $(WTS_{ud}, \leq)$ – вполне структурированная система переходов с совместимостью по возрастанию и убыванию, где $WTS_{ud} = (S, T, \to, s_0)$.

Для того, чтобы показать, что система WTS_{ud} является системой переходов автоматного типа WTA, для которой опущены управляющие состояния, т.е. частным случаем системы WTA, необходимо в множестве состояний системы WTS_{ud} выявить конечное множество таких непересекающихся групп (подмножеств) состояний, что между этими группами будет существовать отношение переходов, удовлетворяющее определению 4. Основная идея состоит в том, чтобы разбить множество S на непересекающиеся верхние конусы и задать отношение переходов на множестве этих конусов таким образом, чтобы любой переход из одного состояния в другое системы WTS_{ud} был возможен тогда и только тогда, когда существовал бы соответствующий переход между конусами, к которым они принадлежат.

Рассмотрим базис множества S, т.е. множесто $min(S)$ минимальных по отношению $\leq$ состояний системы WTS_{ud}. Сопоставим каждому состоянию $s_i^b \in min(S)$ локальное управляющее состояние q_{k_i}, где $1 \leq k_i \leq n$, $n = |min(S)|$, таким образом, что если для пар (q_{k_i}, s_i^b) и (q_{k_j}, s_j^b) выполняется условие $\uparrow s_i^b \cap \uparrow s_j^b \neq \emptyset$, то $q_{k_i} = q_{k_j}$, т.е. $k_i = k_j$.

Построим множество $S_q \subseteq Q \times S$, $S_q = \{(q_{k_i}, s) \mid q_{k_i} \in Q, s_i^b \leq s\}$, где $Q = \{q_{k_1}, \ldots, q_{k_n}\}$. Множество S_q является верхним конусом и имеет базис $min(S_q) = \{(q_{k_1}, s_1^b), \ldots, (q_{k_n}, s_n^b)\}$.

Зададим на множестве S_q частичный порядок $\leq_q$ таким образом, что $(q_{k_i}, s) \leq_q (q_{k_j}, s') \iff q_{k_i} = q_{k_j} \wedge s \leq s'$. Очевидно, что отношение $\leq_q$ является вполне упорядочиваемым квазипорядком.

Также зададим на множестве S_q отношение переходов $\to_q$ таким образом, что $(q_{k_i}, s) \xrightarrow{t}_q (q_{k_j}, s') \iff s \xrightarrow{t} s' \wedge s_j^b \leq s'$. Очевидно, что если $s' \geq s_j^b$ и $s' \geq s_l^b$, то для (q_{k_j}, s_j^b) и (q_{k_l}, s_l^b) имеем $q_{k_j} = q_{k_l}$.

Построим на множестве Q ещё одно отношение переходов $\to_\bullet$, что $q_{k_i} \xrightarrow{t}_\bullet q_{k_j} \iff (q_{k_i}, s_i^b) \xrightarrow{t}_q (q_{k_j}, s') \iff s_i^b \xrightarrow{t} s' \wedge s_j^b \leq s'$.

Необходимо показать, что для любого состояния $(q_{k_i}, s) \in S_q$ существует некоторый переход с пометкой $t \in T$ $(q_{k_i}, s) \xrightarrow{t}_q (q_{k_j}, s')$ тогда и только тогда, когда существует переход $q_{k_i} \xrightarrow{t}_\bullet q_{k_j}$.

Предположим, что существует переход $(q_{k_i}, s) \xrightarrow{t}_q (q_{k_j}, s')$ и не существует перехода $q_{k_i} \xrightarrow{t}_\bullet q_{k_j}$. Тогда также не существует и перехода $(q_{k_i}, s_i^b) \xrightarrow{t}_q (q_{k_j}, s'')$. Но так как $s_i^b \le s$ и $s \xrightarrow{t} s'$, то должен быть переход $s_i^b \xrightarrow{t} s''$, $s'' \le s'$, что означает наличие и перехода $(q_{k_i}, s_i^b) \xrightarrow{t}_q (q_{k_l}, s'')$, где $l \neq j$. Получается, что, с одной стороны, $s_j^b \le s'$, а с другой $s_l^b \le s'' \le s'$. Следовательно, $s' \in \uparrow s_j^b \cap \uparrow s_l^b$, но тогда $q_{k_j} = q_{k_l}$, т.е. $k_j = k_l$. Пришли к противоречию.

Обратно, из существования перехода $q_{k_i} \xrightarrow{}_\bullet q_{k_j}$ имеем $(q_{k_i}, s_i^b) \xrightarrow{t}_q (q_{k_j}, s'')$ и, следовательно, $s_i^b \xrightarrow{t} s''$. Т.к. $s_i^b \le s$ и $s_i^b \xrightarrow{t} s''$, то должен быть переход $s \xrightarrow{t} s'$ такой, что $s'' \le s'$. Тогда, т.к. $s_j^b \le s'' \le s'$ и $s_i^b \le s$, существует состояние $(q_{k_j}, s') \in S_q$, в которое есть переход с пометкой t из состояния (q_{k_i}, s).

Итак, получили вполне структурированную систему переходов автоматного типа $WSA{=}(S_q, T, \to_q, s_q^0)$ с отношением порядка на множестве состояний $\le_q$, где $s_q^0 = (q_{k_i}, s_0)$, $s_i^b \le s_0$.

По построению система WSA имеет некоторое исполнение $(q_{k_i}, s_0) \xrightarrow{t_1}_q (q_{k_j}, s_1) \xrightarrow{t_2}_q (q_{k_l}, s_2) \xrightarrow{t_3}_q \ldots$ тогда и только тогда, когда система WTS_{ud} имеет исполнение $s_0 \xrightarrow{t_1} s_1 \xrightarrow{t_2} s_2 \xrightarrow{t_3} \ldots$. □

Следствие 1 *Класс вполне структурированных систем переходов с совместимостью по возрастанию и убыванию совпадает с классом вполне структурированных систем переходов автоматного типа.*

Доказательство. Следует из утверждений 4 и 5. □

Из этого утверждения следует разрешимость задачи субпокрытия [3] для класса вполне структурированных систем переходов автоматного типа. Это означает, что для двух состояний s и s' разрешима задача достижимости из s состояния s'', такого что $s'' \le s'$.

4. Автоматная логика

Автоматная логика AL определяется над системами с тотальным отношением переходов, т.е. над системами, где из каждого состояния имеется хотя бы один переход. Рассмотрим произвольную вполне структурированную систему переходов автоматного типа $\mathcal{S}$. Выделим управляющие состояния системы $\mathcal{S}$, из которых не существует переходов. В отношение переходов $\to_\bullet$ добавим для каждого такого локального состояния q пустой переход τ, т.е. переход, переводящий некоторое локальное состояние q само в себя $q \xrightarrow{\tau}_\bullet q$. Также добавим в отношение $\to$ соответствующий τ переход для каждого состояния вида $s = (q, d)$, т.е. $s = (q, d) \xrightarrow{\tau} s = (q, d)$. Таким образом мы получили вполне структурированную систему переходов автоматного типа $\mathcal{S}'$, у которой любое исполнение может продолжаться бесконечно долго. Далее будет предполагаться, что все рассматриваемые системы имеют тотальное отношение переходов.

Определение 5 *Конечный автомат над алфавитом Σ, принимающий конечные последовательности (исполнения), – это набор $\mathcal{A}_f = (\Sigma, Q, q_0, \delta, F)$, где Q – это конечное множество состояний, q_0 – начальное состояние, $\delta \subseteq Q \times \Sigma \times Q$ – функция переходов (наряду с $(q, a, q') \in \delta$ используется и обозначение $q \xrightarrow{a} q'$), $F \subseteq Q$ – множество финальных (принимающих) состояний. Конечная последовательность $a_0 a_1 \ldots a_n \in \Sigma^*$ принимается конечным автоматом $\mathcal{A}_f$, если существует последовательность переходов $q_0 a_0 q_1 a_1 \ldots a_n q_n$ автомата $\mathcal{A}_f$ из состояния q_0 такая, что $\forall i$ $(0 \le i \le n)$: $q_i \in Q$, $a_i \in \Sigma$, и $q_n \in F$. Обозначим $L(\mathcal{A}_f)$ (язык) множество последовательностей, принимаемых конечным автоматом $\mathcal{A}_f$.*

Определение 6 *Автомат Бюхи над алфавитами Σ, принимающий бесконечные последовательности (исполнения), – это набор $\mathcal{A}_\omega = (\Sigma, Q, q_0, \delta, F)$. Бесконечная последовательность $a_0 a_1 \ldots \in \Sigma^\omega$ принимается автоматом Бюхи $\mathcal{A}_\omega$, если существует бесконечная последовательность переходов $q_0 a_0 q_1 a_1 \ldots$ автомата $\mathcal{A}_\omega$ из состояния q_0 такая, что $\forall i \ge 0$: $q_i \in Q$, $a_i \in \Sigma$, и $\overset{\infty}{\exists}\, i \ge 0 : q_i \in F$, т.е. последовательность бесконечно часто проходит через некоторое финальное состояние. Обозначим $L(\mathcal{A}_\omega)$ (язык) множество последовательностей, принимаемых автоматом Бюхи $\mathcal{A}_\omega$.*

Определение 7 *Замкнутый ω-автомат – это автомат Бюхи $\mathcal{A}_{\omega c}{=}(Q, q_0, \Pi, \delta, F)$ такой, что $F{=}Q$.*

Определение 8 (Автоматная логика) *Пусть P есть множество элементарных высказываний. Тогда множество формул логики $AL(P)$ определяется по следующей грамматике ($p \in P$):*

$$\varphi, \varphi_i ::= p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists \mathcal{A}_f(\varphi_1, \ldots, \varphi_m) \mid \forall \mathcal{A}_f(\varphi_1, \ldots, \varphi_m) \mid \exists \mathcal{A}_\omega(\varphi_1, \ldots, \varphi_m) \mid \forall \mathcal{A}_\omega(\varphi_1, \ldots, \varphi_m),$$

где $\mathcal{A}_f$ (соот. $\mathcal{A}_\omega$) – конечный автомат (соот. автомат Бюхи) над алфавитом $\Lambda \times \Sigma$, $\Lambda = \{\varphi_1, \ldots, \varphi_m\}$.

Определение 9 *Пусть $\mathcal{S} = (S, T, \to, s_0)$ – вполне структурированная система переходов автоматного типа. Определим отношение выполнимости между состоянием s системы $\mathcal{S}$ и формулой логики $AL(P)$ следующим образом ($\star$ используется вместо f и ω):*

$$
\begin{aligned}
s \models p,\ p \in P &\iff s \in [\![p]\!] \\
s \models \varphi_1 \vee \varphi_2 &\iff s \models \varphi_1 \text{ или } s \models \varphi_2 \\
s \models \varphi_1 \wedge \varphi_2 &\iff s \models \varphi_1 \text{ и } s \models \varphi_2 \\
s \models \exists \mathcal{A}_\star(\varphi_1, \ldots, \varphi_m) &\iff \exists \pi = s_0 a_0 s_1 a_1 \ldots \in Run(s, \mathcal{S}) \colon \exists \sigma = q_0(\varphi_{i_0}, a_0) q_1(\varphi_{i_1}, a_1) \ldots \in L(\mathcal{A}_\star) \colon \\
&\qquad \forall j : 0 \le j < |\sigma| : s_j \models \varphi_{i_j} \\
s \models \forall \mathcal{A}_\star(\varphi_1, \ldots, \varphi_m) &\iff \forall \pi = s_0 a_0 s_1 a_1 \ldots \in Run(s, \mathcal{S}) \colon \exists \sigma = q_0(\varphi_{i_0}, a_0) q_1(\varphi_{i_1}, a_1) \ldots \in L(\mathcal{A}_\star) \colon \\
&\qquad \forall j : 0 \le j < |\sigma| : s_j \models \varphi_{i_j}
\end{aligned}
$$

Для каждой формулы φ полагаем, что $[\![\varphi]\!]_{\mathcal{S}} = \{s \in S \mid s \models \varphi\}$. $Run(s, \mathcal{S})$ обозначает множество исполнений системы $\mathcal{S}$ из состояния s.

Определение 10 (Подмножества AL) *Логика $\exists AL(P)$ (соот. $\forall AL(P)$) – это подмножество логики AL, содержащее формулы, построенные из элементарных высказываний $p \in P$, операторов конъюнкции и дизъюнкции и квантора существования (соот. всеобщности). Пусть $\mathcal{L}$ – подмножество логики AL. Тогда $\mathcal{L}_f$ (соот. $\mathcal{L}_\omega$, $\mathcal{L}_{\omega c}$) обозначается подмножество логики $\mathcal{L}$, где используются только конечные автоматы (соот. автоматы Бюхи, замкнутые ω-автоматы).*

Условимся обозначать P_{UC} множество элементарных высказываний, интерпретирующихся верхними конусами состояний системы переходов $\mathcal{S} = (S, T, \to, s_0)$, т.е. для любого $p \in P_{UC}$ выполняется условие, что $s \models p \Longrightarrow \forall s' \ge s :\ s' \models p$, где $s, s' \in S$, а P_{DC} – множество элементарных высказываний, интерпретирующихся нижними конусами, т.е. для любого $p \in P_{DC}$ выполняется условие, что $s \models p \Longrightarrow \forall s' \le s :\ s' \models p$.

Определение 11 Локальная проверка модели*: Для заданной модели (системы переходов) $\mathcal{S}$ определить, истинна ли формула φ в состоянии s, т.е. проверить для $\mathcal{S}$, выполняется ли $s \models \varphi$.*

Определение 12 Глобальная проверка модели*: Построить множество $[\![\varphi]\!]_{\mathcal{S}}$ всех состояний модели $\mathcal{S}$, в которых истинна формула φ.*

5. Разрешимые свойства

Утверждение 6 *Задача локальной проверки модели для вполне структурированных систем переходов автоматного типа и логики $AL_f(P_{DC})$ разрешима, т.е. существует алгоритм, который для произвольной формулы φ логики $AL_f(P_{DC})$ и произвольной системы переходов автоматного типа $\mathcal{S} = (S, T, \to, s_0)$ проверяет выполнимость φ для $\mathcal{S}$, $s_0 \models \varphi$.*

Доказательство. Доказательство проводится структурной индукцией по вложению подформул в формуле φ логики $AL_f(P_{DC})$.

Базис индукции. Рассмотрим разрешимость данной задачи для произвольного состояния s_0 системы $\mathcal{S}$ и формул вида $\exists \mathcal{A}_f(\varphi_1, \ldots, \varphi_n)$ и $\forall \mathcal{A}_f(\varphi_1, \ldots, \varphi_n)$, где φ_i – формула логики $AL_f(P_{DC})$, построенная только из элементарных высказываний, принадлежащих множеству P_{DC}, и допустимых булевских операторов. Поскольку пересечение и объединение нижних конусов являются также нижними конусами, будем считать формулы φ_i элементарными высказываниями, т.е. $\varphi_i = p_i \in P_{DC}$.

Итак, рассмотрим произвольную вполне структурированную систему переходов автоматного типа $\mathcal{S} = (S, T, \to, s_0)$, а также конечный автомат $\mathcal{A}_f = (\Sigma, Q, q_0, \delta, F)$, где $\Sigma = P_{DC} \times T$.

Построим вполне структурированную систему переходов $\mathcal{S}' = (S', T', \to', s'_0)$, представляющую собой произведение системы $\mathcal{S}$ и конечного автомата $\mathcal{A}_f$. Состояния системы $\mathcal{S}'$ имеют вид $s' = (s, q)$, где s – состояние системы $\mathcal{S}$, q – состояние автомата $\mathcal{A}_f$. Зададим на множестве $\mathcal{S}'$ вполне упорядочиваемый квазипорядок $\le$ таким образом, что $s'_1 = (s_1, q_1) \le s'_2 = (s_2, q_2) \iff (s_1 \le s_2) \wedge (q_1 = q_2)$. Начальное состояние $s'_0 = (s_0, q_0)$. Состояние $s' = (s, q)$ системы $\mathcal{S}'$ является финальным, если состояние q автомата $\mathcal{A}_f$ финальное, т.е. $q \in F$. Переход $t \in T'$ из состояния $s'_1 = (s_1, q_1)$ в некоторое состояние $s'_2 = (s_2, q_2)$ существует тогда и только тогда, когда одновременно существуют переходы $s_1 \xrightarrow{t} s_2$, $q_1 \xrightarrow{(\varphi, t)} q_2$ и $s_1 \models \varphi$.

Для системы $\mathcal{S}'$ построим дерево покрытия с корнем в начальном состоянии s'_0.

Состояние s_0 системы $\mathcal{S}$ будет удовлетворять формуле $\varphi = \exists\mathcal{A}_f(p_1,\ldots,p_n)$ тогда и только тогда, когда в дереве покрытия будет хотя бы одно финальное состояние. Действительно, поскольку элементарные высказывания интерпретируются нижними конусами, если из дерева покрытия достраивать дерево достижимости, то новые состояния будут только покрывающими (новых состояний может и не быть в случае тупикового листа) и поэтому будут принадлежать к тому типу, к какому принадлежат покрываемые состояния. Следовательно, если в дереве покрытия не будет ни одного финального состояния, то не будет финальных состояний и во всём дереве достижимости.

Состояние s_0 системы $\mathcal{S}$ будет удовлетворять формуле $\varphi = \forall\mathcal{A}_f(p_1,\ldots,p_n)$ тогда и только тогда, когда в дереве покрытия с корнем в начальном состоянии s'_0 системы $\mathcal{S}'$ будет существовать поддерево с тем же корнем, удовлетворяющее следующим свойствам. Во-первых, каждое состояние поддерева $s' = (s_1, q_1)$ для всех состояний s_2 и переходов t системы $\mathcal{S}$ таких, что $s_1 \xrightarrow{t} s_2$, должно иметь переходы $(s_1, q_1) \xrightarrow{t} (s_2, q_2)$. Во-вторых, каждая ветка поддерева должна содержать финальное состояние.

Действительно, первое требование необходимо для выполнения условия всеобщности. Нарушение второго требования хотя бы для одной ветки означает наличие конечного или бесконечного исполнения системы $\mathcal{S}'$, не содержащего в себе финальных состояний, т.е. исполнение системы $\mathcal{S}$ не принимается.

Необходимо отметить, что, т.к. все элементарные высказывания интерпретируются нижними конусами, если для состояния s_0 формула φ вида $\exists\mathcal{A}_f(p_1,\ldots,p_n)$ или $\forall\mathcal{A}_f(p_1,\ldots,p_n)$ не выполнена, т.е. $s_0 \not\models \varphi$, то эта формула будет также не верна и для любого состояния $s \geq s_0$.

Предположение индукции. Рассмотрим теперь автомат $\mathcal{A}_f(\varphi_1,\ldots,\varphi_n)$ с алфавитом $\Sigma = \{\varphi_1,\ldots,\varphi_n\} \times T$, где φ_i – произвольные формулы логики $AL_f(P_{DC})$. Предполагая, что для произвольного состояния s системы $\mathcal{S}$ и формулы φ_i существует алгоритм проверки $s \models \varphi_i$, и проводя построения и рассуждения, аналогичные указанным выше, получим алгоритм проверки формул в общем случае. □

Утверждение 7 *Задача локальной проверки модели для вполне структурированных систем переходов автоматного типа и логики $AL_{\omega c}(P_{UC})$ разрешима.*

Доказательство. Проводится аналогично доказательству утверждения 6 структурной индукцией по вложению подформул в формуле φ логики $AL_{\omega c}(P_{UC})$.

Базис индукции. Рассмотрим разрешимость данной задачи для произвольного состояния s_0 системы $\mathcal{S}$ и формул вида $\exists\mathcal{A}_{\omega c}(p_1,\ldots,p_n)$ и $\forall\mathcal{A}_{\omega c}(p_1,\ldots,p_n)$, где $p_i \in P_{UC}$.

Как и раньше (см. доказательство утверждения 6), построим вполне структурированную систему переходов $\mathcal{S}' = (S', T', \to', S'_0)$, представляющую собой произведение системы $\mathcal{S} = (S, T, \to, s_0)$ и замкнутого ω-автомата $\mathcal{A}_{\omega c} = (\Sigma, Q, q_0, \delta, F{=}Q)$, где $\Sigma = P_{UC} \times T$. Затем для системы $\mathcal{S}'$ построим дерево покрытия с корнем в начальном состоянии $s'_0 = (s_0, q_0)$.

Состояние s_0 системы $\mathcal{S}$ будет удовлетворять формуле $\varphi = \exists\mathcal{A}_{\omega c}(p_1,\ldots,p_n)$ тогда и только тогда, когда в дереве покрытия будет существовать лист, представляющий собой покрывающее состояние для некоторого предыдущего состояния на этой же ветке. Действительно, поскольку элементарные высказывания интерпретируются верхними конусами, наличие такого покрывающего состояния (листа) обеспечивает существование бесконечного пути, состоящего из финальных состояний. Все состояния этого пути будут покрывающими и, следовательно, будут удовлетворять тому же элементарному высказыванию, что и покрываемое состояние. Если же все листья дерева покрытия не являются покрывающими, т.е. все листья – тупиковые состояния, которые не имеют потомков, это означает, что не существует бесконечных исполнений системы $\mathcal{S}$, удовлетворяющих формуле φ.

Состояние s_0 системы $\mathcal{S}$ будет удовлетворять формуле $\varphi = \forall\mathcal{A}_{\omega c}(p_1,\ldots,p_n)$ тогда и только тогда, когда в дереве покрытия с корнем в начальном состоянии системы $\mathcal{S}'$ будет существовать поддерево с тем же корнем, удовлетворяющее следующим свойствам. Во-первых, каждое состояние поддерева $s'_1 = (s_1, q_1)$ для всех состояний s_2 и переходов t системы $\mathcal{S}$ таких, что $s_1 \xrightarrow{t} s_2$, должно иметь переходы $(s_1, q_1) \xrightarrow{t} (s_2, q_2)$. Во-вторых, каждая ветка поддерева должна заканчиваться покрывающим состоянием, т.е. не должно быть ни одного тупикового листа. Действительно, первое требование необходимо для выполнения условия всеобщности. Нарушение второго требования хотя бы для одной ветки означает наличие конечного исполнения системы $\mathcal{S}'$, т.е. некоторое бесконечное исполнение системы $\mathcal{S}$ в этом направлении (этой веткой) не принимается.

Необходимо отметить, что, т.к. все элементарные высказывания интерпретируются верхними конусами, если для состояния s_0 формула φ вида $\exists\mathcal{A}_{\omega c}(p_1,\ldots,p_n)$ или $\forall\mathcal{A}_{\omega c}(p_1,\ldots,p_n)$ выполнена, т.е. $s_0 \models \varphi$, то эта формула будет также верна и для любого состояния $s \geq s_0$.

Предположение индукции. Рассмотрим теперь автомат $\mathcal{A}_{\omega c}(\varphi_1,\ldots,\varphi_n)$ с алфавитом $\Sigma = \{\varphi_1,\ldots,\varphi_n\} \times T$, где φ_i – произвольные формулы логики $AL_{\omega c}(P_{UC})$. Предполагая, что для произвольного состояния s

системы $\mathcal{S}$ и формулы φ_i существует алгоритм проверки $s \models \varphi_i$, и проводя построения и рассуждения, аналогичные указанным выше, получим алгоритм проверки формул в общем случае. □

Утверждение 8 *Задача глобальной проверки модели для вполне структурированных систем переходов автоматного типа и формул вида $\exists\mathcal{A}_f(p_1,\ldots,p_n)$ логики $AL_f(P_{UC})$ является разрешимой.*

Доказательство. Рассмотрим произвольную вполне структурированную систему переходов автоматного типа $\mathcal{S} = (S, T, \to, s_0)$, а также конечный автомат $\mathcal{A}_f = (\Sigma, Q, q_0, \delta, F)$ с алфавитом $\Sigma = \{p_1,\ldots,p_n\} \times T$, где $p_i \in P_{UC}$. Как и раньше, построим вполне структурированную систему переходов $\mathcal{S}' = (S', T', \to', s_0')$, представляющую собой произведение системы $\mathcal{S}$ и конечного автомата $\mathcal{A}_f$.

Рассмотрим множество S'_F финальных состояний системы $\mathcal{S}'$, $S'_F = \{(s,q) \mid s \in S, q \in F\}$. Множество S'_F представляет собой верхний конус. Для S'_F построим множество состояний-предшественников $Pred(S'_F)$. Множество $Pred(S'_F)$ также является верхним конусом. Построим последовательность S'_0, S'_1, …, S'_k, где $S'_0 = S'_F$, $S'_{i+1} = S'_i \cup S''_i$, а S''_i представляет собой множество $Pred(S'_i)$ состояний. Из утверждения 2 следует, что эта последовательность верхних конусов стабилизируется при некотором k. Выберем из множества S'_k только те состояния, в которых второй компонент является начальным состоянием конечного автомата $\mathcal{A}_f$, и спроецируем полученное множество на множество состояний системы $\mathcal{S}$. Получим верхний конус $[\![\varphi]\!]_{\mathcal{S}}$ состояний, которые удовлетворяют формуле $\varphi = \exists\mathcal{A}_f(p_1,\ldots,p_n)$. □

6. Неразрешимые свойства

Рассмотрим машину Минского с двумя счётчиками $2cM$. Построим для машины $2cM$ её "слабую" модель $2cMw$ следующим образом. Добавим оператор недетерминированного выбора []. И в $2cM$ заменим каждое правило перехода `if` $\mathtt{x_j} > 0$ `then` $\mathtt{comm_1}$ `else` $\mathtt{comm_2}$ на правило $\mathtt{comm_1}$ [] $\mathtt{comm_2}$. В выражении $x_j := x_j - 1$ оператор '$-$' заменим оператором вычитания до нуля '$\ominus$'. Таким образом, мы получили недетерминированную машину $2cMw$.

Рассмотрим $2cMw$ как помеченную систему переходов. Для каждого счётчика x_j все переходы (из состояний первого типа) с выражением $x_j := x_j + 1$ пометим как inc_j, переходы (из состояний второго типа) с выражением $x_j := x_j \ominus 1$ как dec_j, а переходы, не изменяющие значения счётчика x_j, обозначим nul_j $(j = 1, 2)$. Если переходы inc_j, dec_j, nul_j переводят машину в финальное состояние, то обозначим (переименуем) их соответственно $halt_j^+$, $halt_j^-$, $halt_j^0$. В соответствии с новыми правилами переходов определим отношение переходов на множестве конфигураций $\to \in S \times T \times S$, где $T = \{inc_1, inc_2, dec_1, dec_2, nul_1, nul_2\} \cup \{halt_1^+, halt_2^+, halt_1^-, halt_2^-, halt_1^0, halt_2^0\}$.

Зададим естественным образом отношение $\leq$ частичного порядка на множестве конфигуаций S машины $2cMw$. Для двух конфигураций системы $2cMw$ имеем: $(q, c_1, c_2) \leq (q', c_1', c_2')$, если $q = q'$ и $c_i \leq c_i'$, $i = 1, 2$. Это отношение является вполне упорядочиваемым квазипорядком по лемме Диксона.

Очевидно, что система переходов $(2cMw, \leq)$ является вполне структурированной системой переходов автоматного типа.

Утверждение 9 *Задача проверки модели неразрешима для класса вполне структурированных систем переходов автоматного типа и автоматных логик $\exists AL_f(P_{UC}, P_{DC})$ и $\exists AL_{\omega c}(P_{UC}, P_{DC})$.*

Доказательство. Предположим противное. Допустим, что существует алгоритм позволяющий ответить на вопрос о разрешимости произвольных формул автоматных логик $\exists AL_f(P_{UC}, P_{DC})$ и $\exists AL_{\omega c}(P_{UC}, P_{DC})$ для произвольной системы автоматного типа.

Рассмотрим 2-х счётчиковую машину Минского $2cM$ с начальной конфигурацией $(q_0, 0, 0)$ и построенную на её основе недетерминированную систему переходов $2cMw$.

Для доказательства этого утверждения достаточно построить конечный автомат $\mathcal{A}_f$ и автомат Бюхи $\mathcal{A}_{\omega c}$ такие, что выполнимость формул автоматных логик $\exists A_f(P_{UC}, P_{DC})$ и $\exists AL_{\omega c}(P_{UC}, P_{DC})$ соответственно для состояния s_0, т.е. $s_0 \models \exists A_f(P_{UC}, P_{DC})$ и $s_0 \models \exists A_{\omega c}(P_{UC}, P_{DC})$, означала бы существование у системы $2cMw$ исполнения, равного соответственно конечному и бесконечному исполнению машины Минского $2cM$.

Итак, построим такой конечный автомат $\mathcal{A}_f = (\Sigma, Q, q_0, \delta, F)$ над алфавитом Σ, где $\Sigma = \{(p_j, inc_j), (p_j^+, dec_j), (p_j^-, nul_j), (p_j, halt_j^+), (p_j^+, halt_j^-), (p_j^-, halt_j^0); j = 1, 2\}$, $P_{UC} = \{p_1^+, p_2^+\} \cup \{p_1, p_2\}$, $P_{DC} = \{p_1^-, p_2^-\}$, $p_j^+ =$"$x_j > 0$", $p_j^- =$"$x_j \leq 0$", $p_j = p_j^+ \vee p_j^- =$"$x_j \geq 0$" $(j = 1, 2)$.

Множество состояний $Q = \{q_0, q_f\}$, где q_0 – начальное состояние, а $F = \{q_f\}$. Функцию переходов определим следующим образом: $\delta(q_0, (p_1, inc_1)) = \{q_0\}$, $\delta(q_0, (p_2, inc_2)) = \{q_0\}$, $\delta(q_0, (p_1^+, dec_1)) = \{q_0\}$, $\delta(q_0, (p_2^+, dec_2)) = \{q_0\}$, $\delta(q_0, (p_1^-, nul_1)) = \{q_0\}$, $\delta(q_0, (p_2^-, nul_2)) = \{q_0\}$, а также $\delta(q_0, (p_1, halt_1^+)) =$

$\{q_f\}$, $\delta(q_0,(p_2,halt_2^+)) = \{q_f\}$, $\delta(q_0,(p_1^+,halt_1^-)) = \{q_f\}$, $\delta(q_0,(p_2^+,halt_2^-)) = \{q_f\}$, $\delta(q_0,(p_1^-,halt_1^0)) = \{q_f\}$, $\delta(q_0,(p_2^-,halt_2^0)) = \{q_f\}$.

По построению конечный автомат $\mathcal{A}_f$ допускает только те переходы системы $2cMw$, которые соответствуют переходам машины Минского $2cM$, и формула $\exists A_f(P_{UC},P_{DC})$ будет истинной для $2cMw$ тогда и только тогда, когда машина Минского останавливается. Следовательно, если существует алгоритм проверки истинности данной формулы $\exists A_f(P_{UC},P_{DC})$ для системы $2cMw$, то существует алгоритм решения задачи останова машины Минского. Пришли к противоречию.

Построим автомат Бюхи $\mathcal{A}_{\omega c} = (\Sigma, Q, q_0, \delta, F)$ над алфавитом $\Sigma = \{(p_j, inc_j), (p_j^+, dec_j), (p_j^-, nul_j);$ $j = 1,2\}$. Множество состояний $Q = \{q_0\}$, $F = \{q_0\}$. Функцию переходов определим следующим образом: $\delta(q_0,(p_1,inc_1)) = \{q_0\}$, $\delta(q_0,(p_2,inc_2)) = \{q_0\}$, $\delta(q_0,(p_1^+,dec_1)) = \{q_0\}$, $\delta(q_0,(p_2^+,dec_2)) = \{q_0\}$, $\delta(q_0,(p_1^-,nul_1)) = \{q_0\}$, $\delta(q_0,(p_2^-,nul_2)) = \{q_0\}$.

Формула $\exists A_{\omega c}(P_{UC},P_{DC})$ будет истинной для $2cMw$ тогда и только тогда, когда машина Минского работает бесконечно долго. Следовательно, если существует алгоритм проверки истинности данной формулы для системы $2cMw$, то опять имеем алгоритм решения задачи останова машины Минского. □

Рассмотрим *счётчиковую машину с обнулениями* rcM [5], у которой правило переходов для состояний q_i второго типа имеет вид:

$$\delta_i:\ \text{if } x_j > 0 \text{ then } (x_j := x_j - 1;\ \text{goto } q_k)\ []\ x_j := 0;\ \text{goto } q_{k'},$$

где [] – оператор недетерминированного выбора. Счётчиковая машина с обнулениями (RCM – reset counter machine) принадежит к классу *счётчиковых машин с потерями* [5].

Необходимо отметить, *для счётчиковых машин с потерями задача существования исполнения в состояние, из которого существует бесконечное исполнение, является неразрешимой* [5].

Построим для счётчиковой машины с обнулениями rcM её "слабую" модель $rcMw$ следующим образом. В машине rcM во всех правилах переходов заменим выражение $x_j := x_j - 1$ на $x_j := x_j \ominus 1$ и уберём условие if $x_j > 0$ then . Как и раньше, рассмотрим недетерминированную счётчиковую машину $rcMw$ как вполне структурированную систему переходов автоматного типа. Для каждого счётчика x_j все переходы (из состояний первого типа) с выражением $x_j := x_j + 1$ пометим как inc_j, переходы (из состояний второго типа) с выражением $x_j := x_j \ominus 1$ как dec_j, а переходы с выражением $x_j := 0$ обозначим nul_j $(j = 1,\ldots,m)$. В соответствии с новыми правилами переходов определим отношение переходов на множестве конфигураций $\rightarrow \in S \times T \times S$, где T – множество, состоящее из меток переходов inc_j, dec_j, nul_j $(j = 1,\ldots,m)$.

Утверждение 10 *Задача проверки модели неразрешима для класса вполне структурированных систем переходов автоматного типа и автоматной логики* $\exists AL_\omega(P_{UC})$.

Доказательство. Рассмотрим m-счётчиковую машину с обнулениями rcM с начальной конфигурацией $(q_0,0,\ldots,0)$ и построенную на её основе недетерминированную систему переходов $rcMw$.

Для доказательства этого утверждения достаточно построить автомат Бюхи $\mathcal{A}_\omega$ такой, что выполнимость формулы автоматной логики $\exists A_\omega(P_{UC})$ для начального состояния s_0, т.е. $s_0 \models \exists A_\omega(P_{UC})$, означала бы существование у системы $rcMw$ исполнения, равного исполнению машины rcM, существование которого проверить алгоритмически невозможно.

Итак, построим автомат Бюхи $\mathcal{A}_\omega = (\Sigma, Q, q_0, \delta, F)$ над алфавитом $\Sigma = P_{UC} \times T$, где $P_{UC} = \{true, p_1^+, \ldots, p_m^+\}$, $p_j^+ =$"$x_j > 0$" $(j = 1,\ldots,m)$, $T = \{inc_1,\ldots,inc_m, dec_1,\ldots,dec_m, nul_1,\ldots,nul_m\}$. Множество состояний $Q = \{q_0, q_f\}$, где q_0 – начальное состояние, а $F = \{q_f\}$.

Функция переходов определяется следующим образом: $\delta(q_0,(true,inc_j)) = Q$, $\delta(q_0,(true,nul_j)) = Q$, $\delta(q_0,(p_j^+,dec_j)) = Q$, $\delta(q_f,(true,inc_j)) = F$, $\delta(q_f,(true,nul_j)) = F$, $\delta(q_f,(p_j^+,dec_j) = F$ $(j = 1,\ldots,m)$.

По построению автомат Бюхи $\mathcal{A}_\omega$ допускает только те переходы системы $rcMw$, которые соответствуют переходам машины rcM. И следовательно, автомат $\mathcal{A}_\omega$ допускает для машины rcM существование исполнения в конфигурацию, из которой существует бесконечное исполнение. Задача существования такого исполнения для счётчиковых машин с потерями является алгоритмически неразрешимой [5]. □

Список литературы

1. Bouajjani A., Mayr R. Model Checking Lossy Vector Addition Systems // Proc. 16th Intern. Symp. on Theoretical Aspects in Computer Science (STACS'99) , LNCS 1563, Trier (Germany), March 1999.

2. Finkel A. Reduction and covering of infinite reachability trees // Information and Computation. 1990. 89(2). P. 144-179.

3. Finkel A., Schnoebelen Ph. Well-structured transition systems everywhere! // Theoretical Computer Science. 2001. 256(1-2). P. 63-92.

4. Kouzmin E., Sokolov V. Communicating Colouring Automata // Proc. International Workshop on Program Understanding, 2003.

5. Mayr R. Lossy counter machines // Tech. Report TUM-I9830, Institut für Informatik, TUM, Munich, Germany, October 1998.

6. Stirling C. P. Modal and temporal logics // Handbook of Logic in Computer Science. 1992. Vol. 2. Oxford University Press. P. 477-563.

7. Stirling C. P. Modal and temporal logics for processes // LNCS 1043. 1996. P. 149-237.

8. Vardi M.Y., Wolper P. Reasoning about infinite computations // Information and Computation. 1994. 115(1): 1-37.

9. Wolper P. Temporal logic can be more expressive // Information and Control. 1983. 56.

10. Кузьмин Е. О разрешимости задачи проверки модели для модального μ-исчисления и некоторых классов вполне структурированных систем переходов // Моделирование и анализ информационных систем. 2003. Т.10. №1. С. 50-55.

11. Минский М. Вычисления и автоматы. М.: Мир, 1971.

12. Хопкрофт Д., Мотвани Р., Ульман Д. Введение в теорию автоматов, языков и вычислений. 2-е изд.: Пер. с англ. М.: Вильямс, 2002. 528 с.

Модел. и анализ информ. систем. Т.11, №1 (2004) 18–23

УДК 541.1+612.82

Распространение возбуждения по сети импульсных нейронов, организованных в кольцо.

Лагутина Н.С.
Ярославский государственный университет
150 000, Ярославль, Советская, 14

В работе рассмотрена модель сети импульсных нейронов, организованная в кольцевую структуру. Показано, что можно выбрать синаптические коэффициенты воздействия таким образом, чтобы спайки элементов начинались через заранее заданное время. Система может бесконечно долго хранить периодическую последовательность импульсов.

1. Введение

Рассмотрим модель импульсного нейрона, учитывающую некоторые особенности биологической нервной клетки. В основе модели лежит представление о том, что работа нейронов согласуется как по пространству, так и по времени. Хранение информации рассматривается как динамический процесс, связанный с образованием волн нейронной активности.

Состояние нервной клетки характеризуется разностью потенциалов между внутренней и внешней поверхностью мембраны (границы) тела нейрона — мембранным потенциалом. Если потенциал достигает некоторого порогового уровня, то нейрон генерирует кратковременный высокоамплитудный импульс (спайк), обусловленный ионными токами, проходящими через активные каналы мембраны. Рожденный нейроном спайк распространяется по аксону (длинному древовидному отростку от тела нейрона) и его разветвлениям. Последние образуют с другими нейронами контакты — синапсы. При поступлении сигнала по аксону к синапсу в нем освобождаются химические вещества (медиаторы), которые вызывают возбуждение или торможение в нейроне-приемнике. На некотором промежутке времени после своего спайка нейрон абсолютно невосприимчив к внешнему воздействию. Этот временной промежуток называется периодом рефрактерности.

Модель взаимодействия нейронов, рассматриваемая в данной работе, учитывает, что нейрон испытывает воздействие другого нейрона только при одновременном выполнении трёх условий: он вышел из состояния рефрактерности; на соответствующих синапсах выделился медиатор; спайк воздействующего нейрона начался в тот момент, когда рассматриваемый нейрон не находился в рефрактерном состоянии. Эта модель построена на основе дифференциально-разностного уравнения [1,2], описывающего изменение мембранного потенциала нейрона, основанного на анализе натриевого и калиевого токов, проходящих через активные каналы мембраны. Учитывается экспериментальный факт запаздывания калиевого тока по отношению к натриевому. Величина задержки принята за единицу времени.

Уравнение мембранного потенциала нейрона, находящегося под воздействием m других нейронов, имеет вид:

$$\dot{u} = \lambda\left[-1 - f_{Na}(u) + f_K(u(t-1))\right]u +$$

$$+\left[\lambda\alpha\sum_{k=1}^{m} g_k \cdot H(u)\cdot\psi(v_k)\cdot\theta\left(\int_{t-(2-\varepsilon)T_1}^{t}(H(u)-\psi(v_k))dt+\delta\right)\right]u(t). \qquad (1)$$

Здесь $f_{Na}(u)$ и $f_K(u)$ — положительные достаточно гладкие функции, характеризующие состояние натриевых и калиевых каналов, с помощью которых происходит перенос ионов через мембрану. Будем считать, что $f_{Na}(u) \to 0$ и $f_K(u) \to 0$ при $u \to \infty$ быстрее, чем $O(u^{-1})$. По смыслу задачи параметр, определяющий скорость протекания электрических процессов, $\lambda \gg 1$. Пусть также выполнено условие:

$$\alpha = f_K(0) - f_{Na}(0) - 1 > 0. \qquad (2)$$

Параметр T_1 — длительность спайка нейрона. При отсутствии внешнего воздействия нейрон генерирует импульс через время:

$$t_2 = T_2 + o(1) = T_1 + 1 + \frac{f_{Na}(0)+1}{\alpha} + o(1),$$

доказательство этого утверждения рассматривается в [3,4].

Далее g_k — вес, а $v_k(t)$ — мембранный потенциал, соответствующие k-му нейрону; функция ψ является индикатором наличия медиатора:

$$\psi(v) = 1 - \theta(1 - v(t))\theta(1 - v(t - (1-\varepsilon)T_1)), \tag{3}$$

где θ — функция Хевисайта, а $0 < \varepsilon < 0.5$; функционал $H(u)$ обеспечивает наличие периода рефрактерности $T_R \approx 3T_1$ у нейрона, находящегося под воздействием:

$$H(u) = \theta(1 - u(t))\theta(1 - u(t - (1-\varepsilon)T_1))\theta(1 - u(t - 2(1-\varepsilon)T_1)); \tag{4}$$

$0 < \delta < 0.5$.

Следует отметить два существенных отличия этой модели взаимодействия нейронов от рассматриваемой в [3] и [4]. Во-первых, увеличивается время жизни медиатора до $(2-\varepsilon)T_1$, во-вторых, учитывается особенность выделения медиатора в то время, когда нейрон находится в рефрактерном состоянии.

2. Организация колебаний в кольцевой структуре из N импульсных нейронов

Рассмотрим кольцевую структуру из $N > 2$ одинаковых импульсных нейронов. Пронумеруем ее элементы от 1 до N. Пусть в рассматриваемой сети i-й нейрон ($i = 1, ..., N$) имеет синапсы, оканчивающиеся на $i-1$-ом и $i+1$-ом нейронах. Далее будем отождествлять $cN \pm i$-ый элемент системы с i-ым ($c = 1, 2, \ldots$), а предшественником первого нейрона будем считать N-ый. Исследуем процесс последовательной генерации импульсов нейронами рассматриваемой сети.

Пусть в нулевой момент времени $t_1^1 = 0$ начался спайк первого элемента, а спустя некоторое время ξ_2^1 под воздействием спайка первого нейрона генерирует спайк второй элемент сети. Генерацию спайка вторым нейроном отнесем к моменту времени $t_2^1 = \xi_2^1$. В момент времени $t_3^1 = \xi_2^1 + \xi_3^1$ начался спайк третьего элемента. Далее, в порядке возрастания номеров, следуют моменты спайков остальных элементов кольца: $t_4^1 < \ldots < t_N^1$. Временные промежутки или рассогласования между моментами начала спайков $i-1$-го и i-го нейронов обозначим через $\xi_i^1 = t_i^1 - t_{i-1}^1$, ($i = 2, ..., N$). Спайк последнего N–го элемента состоится в момент времени $t_N^1 = \sum_{i=2}^{N} \xi_i^1$. Описанный на промежутке $[t_1^1; t_N^1]$ процесс можно назвать однократным прохождением волны возбуждения по кольцу или первым тактом прохождения волны импульсов.

Если первый нейрон к моменту времени t_N^1 уже вышел из рефрактерного состояния, то он попадает под влияние спайка N-го элемента кольца. Можно вычислить момент времени t_1^2 генерации второго спайка первым нейроном и указать рассогласование ξ_1^2 между спайками первого и N элементов структуры. Далее, проводя аналогичные рассуждения последовательно для второго, третьего и так далее нейронов, можно определить рассогласования ξ_i^2, $i = 2, 3, \ldots, N$ между спайками соответствующих элементов сети на втором такте прохождения волны возбуждения. Эти величины выражаются через временные рассогласования между импульсами нейронов на предыдущем такте ξ_i^1, $i = 2, 3, \ldots, N$. Применяя эти рассуждения для второго и третьего тактов, мы получим, что временные рассогласования на третьем такте прохождения волны возбуждения по кольцу выражаются через соответствующие рассогласования на втором такте. Возникает итерационный процесс. Можно доказать, что он сходится.

Обозначим предельные значения рассогласований между спайками нейронов как $\xi_1^0, \xi_2^0, \ldots, \xi_N^0$. Полный период системы равен $\overline{T} = \sum_{i=1}^{N} \xi_i^0$.

При этом необходимо учесть следующее. Во-первых, возбуждение от любого элемента кольца, возвратившись к нему, должно попадать на период времени, когда рассматриваемый нейрон вышел из состояния рефрактерности, т.е. $\overline{T} - \xi_i^0 > T_R$ для любого $i = 1, 2, \ldots, N$. Во-вторых, все спайки должны носить индуцированный характер, т.е. являться результатом воздействия. Нейроны обладают собственной авторитмичностью с периодом $T_2 + o(1)$, поэтому последнее требование выражается с помощью неравенства $\overline{T} < T_2$. Если указанные условия соблюдаются, то по кольцу будет распространяться волна возбуждения в сторону возрастания номеров элементов (за N-ым нейроном следует первый). Подчеркнем, что нейрон после генерации спайка на промежутке T_R абсолютно невосприимчив к внешнему воздействию, поэтому

спайк i-го элемента системы не оказывает влияния на $i-1$-ый нейрон. Таким образом, волна возбуждения будет распространяться от первого нейрона ко второму и далее по направлению увеличения номеров элементов сети.

Параметры волны зависят от синаптических коэффициентов воздействия нейронов. Их можно подобрать таким образом, чтобы промежутки времени между спайками соседних элементов принимали заранее заданные значения.

Обозначим через $q_{i,i+1}$ - синаптический вес воздействия i-го элемента на $i+1$-ый нейрон ($i = 1, 2, 3, \ldots, N$). Синаптический вес воздействия i-го нейрона на $i-1$-ый обозначим $q_{i,i-1}$ для $i = 1, 2, \ldots, N$.

Зададим начальное состояние сети следующим образом. Пусть t_i^1 - последовательные моменты спайков нейронов (i — номер соответствующего нейрона) на первом такте прохождения волны возбуждения по кольцу ($i = 1, 2, \ldots, N$).

Рассмотрим первый такт прохождения волны возбуждения по кольцу, т.е. функционирование системы на промежутке $[t_1^1, t_N^1]$. Моменты времени t_i^1 ($i = 1, 2, \ldots, N$) выберем так, чтобы они удовлетворяли следующим условиям.

Во-первых, волна возбуждения распространяется от первого нейрона ко второму и т.д. в порядке возрастания номеров:

$$t_i^1 > t_{i-1}^1, i = 1, \ldots, N.$$

Во-вторых, спайк i-го нейрона происходит раньше, чем распался медиатор, появившийся в синапсах под действием спайка $i-1$-го элемента сети:

$$t_i^1 < t_{i-1}^1 + 2T_1, i = 1, \ldots, N.$$

В-третьих, волна возбуждения, порожденная спайком первого нейрона, пробегая по кольцу, индуцирует спайк последнего N-го нейрона уже после того, как первый элемент вышел из рефрактерного состояния, но раньше, чем тот успел спонтанно сгенерировать импульс:

$$T_R < t_N^1 < T_2.$$

Обозначим через t_i^k, $k = 2, 3, \ldots$, $i = 1, 2, \ldots, N$ - соответствующие последовательности моментов спайков на произвольном κ-ом такте, который открывается в момент времени t_1^k спайком первого нейрона. Покажем, что существует устойчивый периодический режим работы кольцевой структуры, в котором спайки нейронов сети следуют в порядке возрастания номеров. Для таких режимов распределение моментов спайков нейронов на каждом такте прохождения волны должно удовлетворять ряду условий.

Условие 1. Импульсы нейронов на каждом такте волны должны следовать в порядке возрастания номеров, т.е.

$$t_N^{k-1} < t_1^k < t_2^k < \ldots < t_N^k < t_1^{k+1}, k = 1, 2, \ldots$$

Условие 2. Спайк i-го нейрона происходит раньше, чем распался медиатор, появившийся в синапсах под действием спайка $i-1$-го элемента сети, т.е.

$$t_1^{k+1} < t_N^k + 2T_1, t_i^k < t_{i-1}^k + 2T_1; i = 2, \ldots, N; k = 1, 2, \ldots.$$

Условие 3. Волна возбуждения, порожденная импульсом i-го нейрона $i = 1, 2, \ldots, N$, пробегая по кольцу, индуцирует спайк этого нейрона уже после того, как он вышел из рефрактерного состояния. Неравенства, задающие это условие, выглядят так:

$$t_1^k + T_R < t_N^k, t_i^k + T_R < t_{i-1}^{k+1}, i = 2, \ldots, N, k = 1, 2, \ldots.$$

Условие 4. Все спайки носят индуцированный характер. Поскольку нейроны обладают собственной авторитмичностью, то волна, порожденная спайком i-го элемента, должна возвратиться к нему раньше, чем он успеет спонтанно сгенерировать следующий импульс. Этому условию соответствуют следующие неравенства:

$$t_{i-1}^{k+1} < t_i^k + T_2, i = 2, \ldots, N; t_N^k < t_1^k + T_2; k = 1, 2, \ldots.$$

Введем обозначения для временных рассогласований между спайками i-го и $i-1$-го нейронов кольца ($i = 1, 2, \ldots, N$) на k-ом такте прохождения волны. Определим:

$$\xi_i^k = t_i^k - t_{i-1}^k; i = 2, \ldots, N; k = 1, 2, \ldots$$

Временное отставание момента начала t_1^k импульса первого элемента сети относительно спайка N-го нейрона, т.е. начало k-го такта прохождения волны относительно импульса последнего элемента на $k-1$-ом такте обозначим как:

$$\xi_1^k = t_1^k - t_N^{k-1}; (k = 2, 3, \ldots)$$

Условия 2—4 позволяют записать неравенства, которым должны удовлетворять рассматриваемые временные рассогласования.

Из условия 2 следует:

$$0 < \xi_i^k < 2T_1; i = 1, \ldots, N; k = 1, 2, \ldots. \tag{5}$$

Согласно условию 3, возбуждение от любого нейрона, возвращаясь к нему, попадает на период восприимчивости:

$$\sum_{i=1}^{N} \xi_i^k - \xi_j^k > T_R; j = 1, 2, \ldots, N; k = 1, 2, \ldots. \tag{6}$$

Условие 4 означает, что продолжительность каждого такта меньше собственного периода отдельного нейрона:

$$\sum_{i=1}^{N} \xi_i^k < T_2 . k = 1, 2, \ldots \tag{7}$$

Таким образом, при прохождении волны возбуждения по кольцу нейронов, для временных рассогласований между началами импульсов соседних элементов сети возникает итерационный процесс. Ниже будет доказана его сходимость.

Обозначим через ξ_i^0 - ожидаемые значения промежутков времени между спайками $i-1$-го и i-го нейронов $(i = 1, 2, \ldots, N)$.

Теорема. *Пусть числа ξ_i^0 $(i = 1, 2, \ldots, N)$ удовлетворяют неравенствам:*

$$0 < \xi_i^0 < 2T_1; \sum_{i=1}^{N} \xi_i^0 < T_2; \sum_{j=1}^{N} \xi_j^0 - \xi_i^0 > T_R, (i = 1, 2, \ldots, N), \tag{8}$$

и синаптические веса определены формулами с точностью до $o(1)$:

$$q_{k-1,k} = \frac{T_2 - \sum_{j=1}^{N} \xi_j^0}{\xi_k^0}, \tag{9}$$

$$k = 1, 2, 3, \ldots, N.$$

Тогда существует устойчивый режим работы кольца, при котором импульсы нейронов следуют в порядке возрастания номеров, и временные рассогласования между спайками $(i-1)$-го и i-го нейронов $(i = 1, 2, \ldots, N)$ принимают значения ξ_i^0.

Поясним идею доказательства теоремы. Выберем моменты начала импульсов на первом такте волны, так, чтобы были выполнены условия 1 — 4. Исследуем динамику системы на основе сделанных предположений. Обозначим мембранный потенциал i-ого нейрона через $u_i(t)$. Рассмотрим второй такт прохождения волны по сети. Для каждого нейрона найдем соответствующий момент начала импульса.

Согласно начальным условиям при $t \in [t_1^1, t_N^1]$ в синапсах, прилежащих к первому нейрону, медиатор отсутствует (он появится только для $t > t_N^1$). С момента времени t_N^1 первый нейрон будет испытывать воздействие N-ого нейрона, значит, уравнение для $u_1(t)$ примет вид согласно (1):

$$\dot{u}_1(t) = \lambda [-1 - f_{Na}(u_1(t)) + f_K(u_1(t-1)) + \alpha q_{N,1}] u_1(t), \tag{10}$$

соответствующее начальное условие:

$$u_1(t_N^1) = \exp -\lambda\alpha(T_2 - (t_N^1 - t_1^1) + o(1)).$$

На интервале $[t_N^1 - 1, t_N^1]$ мембранный потенциал первого нейрона почти равен нулю. Поэтому, с учетом формулы (2):

$$-1 - f_{Na}(u_1(t)) + f_K(u_1(t-1)) = -1 - f_{Na}(0) + f_K(0) + o(1) = \alpha + o(1).$$

Соответственно уравнение (10) приобретает вид:

$$\dot{u}_1(t) = \lambda\left[\alpha + \alpha q_{N,1} + o(1)\right] u_1(t).$$

Следовательно, динамика изменения $u_1(t)$ соответствует выражению:

$$u_1(t) = \exp -\lambda\alpha(T_2 - t_N^1 + t_1^1)e^{\lambda\alpha((1+q_{N,1})(t-t_N^1)+o(1))}$$

Значение мембранного потенциала в момент начала импульса равно единице. Используя это условие, из последнего равенства можно определить t_1^2 следующим образом:

$$t_1^2 = \frac{T_2 + t_1^1 + q_{N,1}t_N^1}{(1+q_{N,1}}. \tag{11}$$

Аналогично вычисляем моменты начала спайков для остальных нейронов на втором такте прохождения волны:

$$\dot{u}_i(t) = \lambda\left[-1 - f_{Na}(u_i(t)) + f_K(u_i(t-1)) + \alpha q_{i-1,1}\right] u_i(t),$$
$$u_i(t_{i-1}^2) = \exp -\lambda\alpha(T_2 - (t_{i-1}^2 - t_i^1) + o(1))$$
$$i = 2, 3, \ldots, N.$$

Соответствующее решение:

$$u_i(t) = \exp -\lambda\alpha(T_2 - t_{i-1}^1 + t_i^1)e^{\lambda\alpha((1+q_{i-1,i})(t-t_{i-1}^1)+o(1))}$$
$$i = 2, 3, \ldots, N.$$

Отсюда:

$$t_i^2 = \frac{T_2 + t_i^1 + q_{i-1,i}t_{i-1}^2}{1+q_{i-1,i}} + o(1) \tag{12}$$
$$i = 2, 3, \ldots, N.$$

Эти формулы переносятся на любой такт прохождения волны (если выполнены условия 1—4). Перепишем формулы (11), (12), используя ранее введенные обозначения для временных рассогласований между спайками на k-ом такте:

$$\xi_i^k = \frac{T_2 - (\sum_{j=1}^{i-1}\xi_j^k + \sum_{j=i+1}^{N}\xi_j^{k-1}}{1+q_{i-1,i}} + o(1)$$

Таким образом, получим соотношения:

$$(1+q_{N,1})\xi_1^k + \sum_{j=2}^{N}\xi_j^{k-1} + o(1) = T_2, \tag{13}$$

$$\sum_{j=1}^{i-1}\xi_j^k + (1+q_{i,i-1})\xi_i^k + \sum_{j=i+1}^{N}\xi_j^{k-1} + o(1) = T_2, \tag{14}$$
$$i = 2, 3, \ldots, N-1$$

$$\sum_{j=1}^{N-1}\xi_j^k + (1+q_{N-1,N})\xi_N^k + o(1) = T_2. \tag{15}$$

Формулы (13–15) описывают итерационный процесс, отражающий временные соотношения между началом импульсов соседних нейронов на последовательных тактах прохождения волны по сети. Этот процесс сходится, поскольку итерационные соотношения удовлетворяют условиям метода Зейделя для решения системы линейных уравнений с положительно определенной и симметрической матрицей.

При выборе синаптических весов в соответствии с формулами (9), как можно убедиться с помощью прямой подстановки, отображение (13– 15) имеет неподвижную точку $\xi^0 = (\xi_1^0, \ldots, \xi_N^0)$. В силу неравенств (8) условия 1.—4. выполнены. Тем самым, доказано существование интересующего нас режима. Стандартными рассуждениями можно показать, что условия 1.—4. будут выполнены на любом такте

прохождения волны, если начальная точка ξ^1 (вектор рассогласований на начальном такте волны), лежит в малой окрестности неподвижной точки ξ^0.

Для кольца из N нейронов мы можем указать значения синаптических коэффициентов так, что по нему в сторону возрастания номеров элементов сети будут распространяться волны возбуждения. Промежутки времени между спайками соседних по номеру нейронов принимают заранее заданные значения. Таким образом, построена модель системы, способной бесконечно долго хранить заранее заданную периодическую последовательность.

Если рассмотреть синаптические веса воздействия элемента кольца на предыдущий нейрон, т.е. коэффициенты $q_{i+1,i}$, где $i = 1, 2, \ldots, N$, аналогичными рассуждениями можно убедиться в существовании еще одного периодического режима работы сети. Он может быть организован таким образом, что волны возбуждения будут распространяться в сторону убывания номеров элементов сети и имеют заданные временные промежутки между началом импульсов соседних нейронов.

3. Заключение.

Решенная задача о синтезе системы, хранящей заданную последовательность импульсов, имеет самостоятельный интерес. Кроме того, если принять точку зрения относительно волновой природы памяти, эту конструкцию можно рассматривать как модель нейронной популяции, хранящей след памяти. Итоги исследования согласуются с результатами работ [5], [6], где элементами сетей служили импульсные генераторы (нейроны), описываемые уравнениями с запаздыванием. Аналогичные результаты получены для сетей из нейронных клеточных автоматов [7].

Литература

[1] Майоров В.В., Мышкин И.Ю. Об одной модели функционирования нейронной сети // Модел. динам. попул. Н.Новгород, 1990. С. 70—78.

[2] Майоров В.В., Мышкин И.Ю. Математическое моделирование нейронов сети на основе уравнений с запаздыванием // Математическое моделирование. 1990. Т.2, №11. С. 64—76.

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

[4] Кащенко С.А., Майоров В.В., Мышкин И.Ю. Волновые образования в кольцевых нейронных системах // Математическое моделирование. 1997. Т. 9. №3. С. 29—39.

[5] Кащенко С.А.,Майоров В.В., Мышкин И.Ю. Исследование колебаний в кольцевых нейронных системах // Докл. Российской АН. 1993. Т. 333. №5. С. 594.

[6] Кащенко С.А., Майоров В.В., Мышкин И.Ю. Распространение волн в простейших кольцевых нейронных структурах // Математическое моделирование. 1995. Т. 7. №12. С. 3—18.

[7] Шабаршина Г.В. Проведение возбуждения по кольцевой структуре нейронных клеточных автоматов // Моделирование и анализ информационных систем. Ярославль. 1994. №2. С. 116—121.

Модел. и анализ информ. систем. Т.11, №1 (2004) 24–27

УДК 517.5

Асимптотическое поведение наилучших кусочно-полиномиальных приближений в пространствах L_p $(0 < p < 1)$

Морозов А.Н.
Ярославский государственный университет
150 000, Ярославль, Советская, 14

получена 26 июня 2003

В статье получена точная асимптотическая формула для наилучших кусочно-полиномиальных порядка k приближений функций из W_1^k в тех случаях, когда приближение осуществляется в метрике пространства L_p с $p < 1$. Предложенный метод рассуждения покрывает фактически и случаи приближения в L_p с $p \geq 1$, для которых соответствующие формулы были получены ранее несколько иными рассуждениями.

1. Основные обозначения и результаты

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

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

(при $p < 1$ данный функционал не является нормой, но порождает некоторую метрику). Когда неясность возникнуть не может, сокращаем обозначение до $\|f\|_p$. Из пространств гладких функций рассматриваются пространства ($p \geq 1$)

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

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 полуинтервалов $\{[t_{j-1}, t_j)\}_{j=1}^m$ назовем разбиением полуинтервала $I = [a, b)$. Через u_m будем обозначать разбиение полуинтервала I на m равных по длине полуинтервалов.

Обозначим

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

- наилучшее приближение кусочно-полиномиальными функциями порядка k, подчиненными разбиению u_m полуинтервала I, в $L_p[I]$, 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 функции $f \in W_p^k[I]$ выполняется:

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

с явно определяемой постоянной $c_{k,p}$.
В данной статье аналогичная формула доказана для функции $f \in W_1^k[I]$ при $0 < p < 1$.

2. Приближения в L_p $(0 < p < 1)$

Как известно, наилучшие полиномиальные приближения при $p \geq 1$ обладают свойствами полунормы:

1. $E_k(f; I)_p = 0 \Longleftrightarrow f \in P_k$,
2. $E_k(cf; I)_p = |c|\ E_k(f; I)_p$,
3. $E_k(f_1 + f_2; I)_p \leq E_k(f_1; I)_p + E_k(f_2; I)_p$.

При $0 < p < 1$ "неравенство треугольника" уже не выполняется, однако, справедливо:

$E_k(f_1 + f_2; I)_p^p \leq E_k(f_1; I)_p^p + E_k(f_2; I)_p^p$,
т.е.
$|E_k(f_2; I)_p^p - E_k(f_1; I)_p^p| \leq E_k(f_2 - f_1; I)_p^p$.

Важную роль в дальнейшем будет играть следующее утверждение.

Лемма. Для каждого полуинтервала I при всех $0 < p < \infty$ выполняется:

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

где постоянная $c_{k,p}$ определяется равенством:

$$c_{k,p} = E_k\Big(\frac{t^k}{k!}; [0,1]\Big)_p.$$

Доказательство. Основой доказательства леммы является известное рассуждение с использованием свойств проекции.
Зафиксируем некоторые $0 < p < \infty$ и $I = [a, b)$. Определим оператор $T : L_p[I] \to L_p[0,1]$, действующий по формуле: $(Tf)(t) = f((b-a)t + a)$. Ясно, что есть соотношение:

$$\|f\|_{L_p[I]} = |I|^{\frac{1}{p}}\ \|Tf\|_{L_p[0,1]}.$$

Поскольку, кроме того, $T : P_k \to P_k$, то из свойств наилучшего приближения следует, что

$$E_k(f; I)_p = |I|^{\frac{1}{p}}\ E_k(Tf; [0,1])_p.$$

Пусть $f(t) = t^k$. Получаем:

$$E_k(t^k; I)_p = |I|^{\frac{1}{p}}\ E_k\Big(\big((b-a)t + a\big)^k; [0,1]\Big)_p =$$

$$= |I|^{\frac{1}{p}}\ E_k\big((b-a)^k t^k + \pi_k(t); [0,1]\big)_p =$$

$$= |I|^{k+\frac{1}{p}}\ E_k(t^k; [0,1])_p,$$

здесь π_k - некоторый многочлен порядка k.

Лемма доказана.

Теорема. Пусть $0 < p \leq 1$, а функция $f \in W_1^k[I]$, тогда имеет место соотношение:

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

где постоянная $c_{k,p}$ определяется указанным в лемме образом.

Доказательство. Заметим, что

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

где$\{J\}$ - набор полуинтервалов равномерного разбиения полуинтервала I на m частей ($|J| = \frac{|I|}{m}$).
Определим ступенчатую функцию $\varphi_m(f)_{k,p}$ подчиненную разбиению u_m, соотношениями: на каждом интервале $J \in u_m$

$$\varphi_m(f)_{k,p}\ |J = c_{k,p}^{-1}\ \frac{E_k(f;J)_p}{|J|^{k+\frac{1}{p}}}.$$

Отметим, что при $p \geq 1$ функция $\varphi_m(\cdot)_{k,p}$ удовлетворяет "неравенству треугольника", а при $0 < p < 1$ выполняется:

$$|\varphi_m(f_2)_{k,p}^p - \varphi_m(f_1)_{k,p}^p| \leq \varphi_m(f_2 - f_1)_{k,p}^p.$$

Перепишем

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

$$= c_{k,p}|I|^k\ \|\varphi_m(f)_{k,p}\|_{L_p[I]}.$$

Таким образом, утверждение теоремы равносильно условию:

$$\|\varphi_m(f)_{k,p}\|_p - \|f^{(k)}\|_p \to 0, \text{при } m \to \infty.$$

Рассмотрим произвольный полуинтервал J из разбиения u_m. Поскольку $f \in W_1^k[I]$, то п.в. на J существует $f^{(k)}$. Пусть $t_0 \in J$ - некоторая точка, в которой определена k-тая производная. Тогда функцию f можно разложить в этой точке в ряд Тейлора с остаточным членом в форме Пеано. Для полуинтервала J получим соотношение:

$$\left|\left(c_{k,p}^{-1}\frac{E_k(f;J)_p}{|J|^{k+\frac{1}{p}}}\right)^p - |f^{(k)}(t_0)|^p\right| = c_{k,p}^{-p}\ \frac{\left|E_k(f;J)_p^p - E_k\left(\frac{f^{(k)}(t_0)}{k!}t^k;J\right)_p^p\right|}{\left(|J|^{k+\frac{1}{p}}\right)^p} \leq$$

$$\leq c_{k,p}^{-p}\left(\frac{E_k\left(f - \frac{f^{(k)}(t_0)}{k!}t^k;J\right)_p}{|J|^{k+\frac{1}{p}}}\right)^p = c_{k,p}^{-p}\left(\frac{E_k\big(\varepsilon(t);J\big)_p}{|J|^{k+\frac{1}{p}}}\right)^p,$$

где $\varepsilon(t) = o\big(|t-t_0|^k\big)$ -остаточное слагаемое в тейлоровском разложении функции f. Очевидно,

$$E_k(\varepsilon;J)_p \leq \max_{t\in J}\big|\varepsilon(t)\big|\cdot|J|^{\frac{1}{p}} = o\big(|J|^{k+\frac{1}{p}}\big) = o\Big(\Big(\frac{|I|}{m}\Big)^{k+\frac{1}{p}}\Big).$$

Значит, последовательность функций $\big(\varphi_m(f)_{k,p}^p\big)$ сходится п.в. к функции $|f^{(k)}|^p$при $m \to \infty$.
Докажем, что $\big(\varphi_m(f)_{k,p}\big)$ сходится к $|f^{(k)}|$ в L_p, для чего воспользуемся утверждением, близким к классической теореме Лебега (см.[2], лемма 1):
"Пусть последовательность неотрицательных функций (φ_m) сходится п.в. на I к функции g, и при каждом m выполняется неравенство $\varphi_m \leq \psi_m$. Тогда если последовательность (ψ_m) сходится в среднем на I, то (φ_m) сходится в среднем на I к g."
Как отмечено в [1], для $f \in W_1^k$ последовательность $\big(\varphi_m(f)_{k,1}\big)$ сходится в среднем к $|f^{(k)}|$, следовательно, $\big(\varphi_m(f)_{k,p}^p\big)$ сходится в среднем к $|f^{(k)}|^p$ при $0 < p < 1$. Аналогично, из свойств наилучшего приближения при помощи неравенства Гельдера с показателями $\frac{1}{p}$ и $\frac{1}{1-p}$ следует, что

$$E_k(f;J)_p \leq\ |J|^{\frac{1}{p}-1}E_k(f;J)_1,$$

т.е.

$$\frac{E_k(f;J)_p}{|J|^{k+\frac{1}{p}}} \leq \frac{E_k(f;J)_1}{|J|^{k+1}}.$$

Откуда,

$$\varphi_m(f)_{k,p} \leq \frac{c_{k,1}}{c_{k,p}}\varphi_m(f)_{k,1}.$$

Применяя лемму 1 из [2] к функциям $\varphi_m = \varphi_m(f)_{k,p}$, $\psi_m = \frac{c_{k,1}}{c_{k,p}}\varphi_m(f)_{k,1}$ и $g = |f^{(k)}|$, получаем требуемое утверждение.

Теорема доказана.

3. Заключение

Фактически в доказательстве теоремы было получено несколько больше, чем утверждалось в формулировке, а именно: $||\varphi_m(f)_{k,p} - |f^{(k)}|\,||_p \to 0$, при $m \to \infty$. Из этого следует, что при $0 < p < 1$ утверждение теоремы можно распространить на более широкий класс функций, чем W_1^k. Что не удивительно, ведь в заданном контексте приближения пространство W_1^k соответствует $p = 1$.

Литература

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.

Модел. и анализ информ. систем. Т.11, №1 (2004) 28–31

УДК 517.928

Анализ поведения решения одного дифференциального уравнения с запаздывающим аргументом

Кубышкин Е. П.
Ярославский государственный университет
150 000, Ярославль, Советская, 14

получена 30 октября 2003

В работе изучается поведение решений линейного дифференциального уравнения с запаздывающим аргументом типа уравнения Матье. Это уравнение рассматривалось в работе [1], где построено первое приближение области неустойчивости. Использовался метод функционалов Ляпунова. В настоящей заметке, используя результаты работы [2], дан полный анализ поведения решений, построена вторая область неустойчивости, показана аналитическая зависимость областей неустойчивости от малого параметра.

Рассмотрим следующее дифференциальное уравнение с запаздывающим аргументом

$$\ddot{x}(t) + \omega^2 \left(x(t) - \mu \cos(2t) x(t-h)\right) = 0, \tag{1}$$

$0 < \mu \ll 1$, $h > 0$. При $h = 0$ уравнение (1) превращается в известное уравнение Матье [3]. Будем интересоваться поведением решений уравнения (1) при $t > 0$. Под решением уравнения (1) будем понимать функцию $x(t+\tau)$ $(t > 0,\ -h \le \tau \le 0)$, дважды непрерывно дифференцируемую при $t > 0$, обращающую (1) в тождество и удовлетворяющую начальным условиям

$$x(t+\tau)|_{t=0} = x_0(\tau) \in C(-h, 0), \quad \dot{x}(0) = x_1. \tag{2}$$

Положим $x_1(t) = x(t)$, $x_2(t) = \dot{x}(t)$. В дальнейшем обозначим $x(t) = \mathbf{col}\,(x_1(t), x_2(t))$ и перейдем от уравнения (1) к эквивалентной системе уравнений

$$\dot{x}(t) = A_0(\omega) x(t) + \mu A_1(t) x(t-h), \tag{3}$$

где соответственно

$$A_0(\omega) = \begin{pmatrix} 0 & 1 \\ -\omega^2 & 0 \end{pmatrix}, \qquad A_1(t) = \begin{pmatrix} 0 & 0 \\ \omega^2 \cos(2t) & 0 \end{pmatrix}.$$

Пусть сначала $\omega \neq 1, 2$. Отметим, что вектор-функции $e(t) = \exp(i\omega t)\,\mathbf{col}\,(1, i\omega)$ $(i = \sqrt{-1})$ $\bar{e}(t)$ и $h(t) = 2^{-1} exp(i\omega t)\,\mathbf{col}\,(1, i\omega^{-1})$, $\bar{h}(t)$, $(e(t), h(t)) = 1$, $(e(t), \bar{h}(t)) = 0$ $((\cdot,\cdot)$ – скалярное произведение в $C^2)$ соответственно являются решениями уравнений

$$\dot{x}(t) = A_0(\omega) x(t), \quad \dot{Y}(t) = A_0^*(\omega) Y(t),$$

где $A_0^*(\omega)$ – сопряженная с $A_0(\omega)$ матрица.

В соответствии с [2, 4] введем в рассмотрение матричный ряд

$$x(t;\mu) = (V_0(t) + \mu V_1(t) + ...) \exp\left((\mu D_1 + \mu^2 D_2 + ...)t\right) \tag{4}$$

Здесь $V_0(t) = (e(t), \bar{e}(t))$, матрицы-функции (2×2) $V_j(t) = (v_j(t), \bar{v}_j(t))$, $v_j(t) = v_{j0}(t) \exp(i\omega t)$, $v_{j0}(t) = v_{j0}(t + \pi)$ – тригонометрические многочлены, постоянные матрицы

$$D_j = \begin{pmatrix} d_{j1} & \bar{d}_{j2} \\ d_{j2} & \bar{d}_{j1} \end{pmatrix}. \tag{5}$$

Ниже будет показано, что ряд (4) при малых μ сходится, является матричным решением уравнения (3) и своим поведением при $t \to \infty$ полностью определяет характер устойчивости решений уравнения (3). Таким образом, устойчивость решений уравнения (3) определяется характером расположения собственных значений матрицы $D_1 + \mu D_2 ...$

Подставим ряд (4) в уравнение (3) и приравняем коэффициенты при одинаковых степенях μ. В результате получим рекуррентную последовательность матричных дифференциальных уравнений вида

$$\dot{V}_j(t) = A_0(\omega)V_j(t) + F_j(t) - V_0(t)D_j, \tag{6}$$

где $F_j(t) = (f(t), \bar{f}(t))$ – известные матрицы-функции, определяемые $V_k(t) \quad (k = 0, 1, ..., j-1)$, структура которых аналогична $V_j(t)$. Условия разрешимости уравнения (6)

$$M\left(f_j(t) - e(t)d_{j1} - \bar{e}(t)d_{j2},\ \{h(t), \bar{h}(t)\}\right) = 0 \tag{7}$$

в классе матриц вида (4) однозначно определяют $d_{j1},\ d_{j2}$. Здесь $M\left(f(t)\right) = \lim_{T\to\infty}(2T)^{-1}\int_{-T}^{T} f(t)\,dt$, выражение в фигурных скобках обозначает множество переменных векторов.

При этом матрицы $V_j(t) = (v_j(t), \bar{v}_j(t))$ следует выбирать удовлетворяющими условиям

$$M\left(v_j(t),\ \{h(t), \bar{h}(t)\}\right) = 0. \tag{8}$$

С необходимостью имеем $f_1(t) = \mathbf{col}\,(0, 2^{-1}\omega^2\exp(-(\omega h)\,(\exp(i(2+\omega)t + \exp(-i(2-\omega)t)$. Согласно (7) $d_{11} = d_{12} = 0$. Определяя теперь $v_1(t) = \mathbf{col}\ (v_{11}(t), v_{12}(t))$ удовлетворяющим условию (8), имеем

$$v_{11}(t) = -\omega^2\exp(-i\omega h)\left[(1+\omega)^{-1}\exp(i(\omega+2)t) + (1-\omega)^{-1}\exp(i(\omega-2)t)\right]/8.$$

Вид $v_{12}(t)$ для дальнейшего существенного значения не имеет. Отсюда

$$f_2(t) = \mathbf{col}\,(0, -\frac{\omega^4}{16}\exp(i\omega t)((1+\omega)^{-1}\exp(-i(2+2\omega)h) + (1-\omega)^{-1}\exp(i(2-2\omega)h)) + \ldots).$$

Здесь точками обозначены слагаемые, не имеющие существенного значения для дальнейших вычислений. Согласно (7) будем иметь

$$d_{21} = \frac{i\omega^3}{32}\left[(1+\omega)^{-1}\exp(-i(2+2\omega)h) + (1-\omega)^{-1}\exp(i(2-2\omega)h)\right].$$

Таким образом, устойчивость (неустойчивость) решений уравнения (3) определяется условием

$$Re\,d_{21} < 0\,(>0),$$

что эквивалентно условию

$$\frac{\sin((2+2\omega)h)}{1+\omega} - \frac{\sin((2-2\omega)h)}{1-\omega} < 0\,(>0). \tag{9}$$

Рассмотрим случай $\omega \approx 1$. Положим $\omega = 1+\delta,\ |\delta| \ll 1$. Аналогом (4) здесь будет ряд

$$x(t;\mu,\delta) = \left(V_0(t) + \mu V_1(t) + \delta V^1(t) + ...\right)\exp\left((\mu D_1 + \delta D^1 + ...)t\right) \tag{10}$$

Структура матриц $D_j,\ D^j$ и $V_j(t)$ аналогична (4). Подставляя (10) в (3) и приравнивая коэффициенты при одинаковых степенях μ и δ, получим уравнения для определения $V_1(t),\ V^1(t),\ D_1,\ D^1$. Несложные вычисления дают $d_{11} = 0,\ d_{12} = 2^{-1}\exp(ih),\ d_1^1 = i,\ d_2^1 = 0$. С учетом этого и предельного перехода при $\omega \to 1$ в (9), условия устойчивости (неустойчивости) решений уравнения (3) в рассматриваемом случае имеют вид $\delta^2 - \mu^2/4 > 0\ (<0)$. Таким образом, при

$$\delta < \pm 2^{-1}\mu \tag{11}$$

решения неустойчивы. Условия (11) определяют в плоскости $(\omega,\ \mu)$ область неустойчивости решений – область параметрического резонанса.

Рассмотрим теперь случай $\omega \approx 2$. Положим $\omega = 2+\delta, \quad |\delta| \ll 1$. Поступая аналогично предыдущему случаю, имеем

$$d_{11} = d_{12} = 0, \quad d_1^1 = i, \quad d_2^1 = 0, \quad v_{11}(t) = 2^{-1}\exp(-i2h)\,(1 - \exp(i4t)/3)\,.$$

Отсюда

$$f_{21}(t) = \mathbf{col}\ (0, exp(-i2h) - \exp(i6h)/3\exp(i2t))$$

и согласно (3)

$$d_{21} = -i\left(3\exp(-i2h) - \exp(-i6h)\right)/12, \quad d_{22} = i\exp(-i2h)/4.$$

Таким образом, устойчивость (неустойчивость) решений уравнения (3) здесь определяют корни уравнения

$$\lambda^2 + a(\mu)\lambda + b(\mu,\delta) = 0, \tag{12}$$

где

$$\begin{aligned} a(\mu) &= -\mu^2\left(\sin(6h) - 3\sin(2h)\right)/6, \\ b(\mu,\delta) &= \left\{-9\mu^4 + \left[\mu^2(\cos(6h) - 3\cos(2h)) + 12\delta\right]^2 + \mu^4(\sin(6h) - 3\sin(2h))^2\right\}/144. \end{aligned}$$

Как известно, при $a(\mu),\ b(\mu,\delta) > 0$ корни уравнения (12) лежат в левой комплексной полуплоскости. Первое условие выполняется при $0 < h < \pi/2$. Пусть h удовлетворяет этому условию. Нарушение второго условия приводит к неравенству $b(\mu,\delta) < 0$. Граница области неустойчивости

$$\delta = \delta(\mu) = \gamma_1\mu + \gamma_2\mu^2 + \ldots \tag{13}$$

определяется из уравнения $b(\mu,\delta) = 0$. В результате имеем

$$\gamma_1 = 0, \quad \gamma_2 = \left\{-\cos(6h) + 3\cos(2h) + \left[9 - (\sin(6h) - 3\sin(2h))^2\right]^{1/2}\right\}/12. \tag{14}$$

Таким образом, в рассматриваемом случае при $0 \le h < \pi/2$ в плоскости (ω,μ) имеется область неустойчивости решений уравнения (3) (область параметрического резонанса), определяемая (12)–(14). При $\pi/2 \le h < \pi$ решения уравнения (3) неустойчивы.

Обоснование результатов следует из работы (2). Однако остановимся кратко на схеме доказательства. Покажем сначала сходимость рядов в представлении (4).

Введем пространство T функций вида $f(t) = g_1(t)\exp(i\omega t) + g_2(t)\exp(i\omega t)$ $(-\infty < t < \infty)$ со значениями в C^2, где $g_j(t) = g_j(t+\pi)$ непрерывно дифференцируемые функции $|\dot g_j(t)| < \infty$ ($|*|^2 = (*,*)$, $M\left((f(t),\ \{h(t),\bar h(t)\})\right) = 0$. Норму в T определим как $\|f\| = \sup_t |\dot f(t)|$. Через T^2 обозначим пространство матриц-функций (2×2) вида $F = (f,\bar f)$, нормированных естественным образом.

Отметим, что дифференциальное уравнение

$$B(\omega)x(t) \equiv \dot x(t) - A_0(\omega)x(t) = f(t)$$

в T однозначно разрешимо, $x(t) = B^{-1}(\omega)f(t)$. Оператор $B^{-1}(\omega)$ вполне непрерывен.

Рассмотрим матричное выражение $(V_0(t) + \mu V)\cdot\exp(\mu D t)$, где $V_0(t)$ определена в (4), $V \in T^2$, матрица D имеет вид (5). Подставим это выражение в уравнение (3). В результате имеем следующее равенство

$$V(t) = \mu B^{-1}(\omega)\left[A_1(t)\left(V_0(t-h) + \mu V(t-h)\right) - \left(V_0(t) + \mu V_1(t)\right)D\right] \tag{15}$$

для определения $V(t;\mu)$ и $D(\mu)$. Условия принадлежности правой части (15) пространству T^2 позволяют при малых μ и $\|V\|_{T^2}$ определить $D = D(V;\mu)$. В результате имеем в T^2 операторное уравнение для определения $V(t;\mu)$, которое при малых μ однозначно разрешимо. При этом имеем аналитическую зависимость решения от параметра μ.

Рассмотрим поведение остальных решений уравнения (3). Обозначим через E фазовое пространство уравнения (3), т.е. в соответствии с (2) совокупность векторов вида $x(\tau) = \mathbf{col}\,(x_0(\tau), x_1)$, $x_0(\tau) \in C(-h,0)$, $x_1 \in R$. Выражение (4) определяет двумерное инвариантное подпространство решений уравнения (3), определяемое выражением $x(t;\mu)x^{-1}(0,\mu)c_0$, $c_0 = \mathbf{col}(c_{01},\bar c_{01})$. Это подпространство определяет множество начальных условий $E_+ \subset E$. Обозначим через E_- ортогональное дополнение к E_+ до E $(E = E_+ \oplus E_-)$, введенное посредством естественным образом определенного скалярного произведения для уравнения с запаздывающим аргументом (3). Пусть $x_-(\tau) \in E_-$. Тогда с необходимостью получаем, что $x_0(0) = 0$, $x_1 = 0$.Возьмем $x_-(\tau) \in E_-$ в качестве начального условия для решения $x_-(t+\tau)$ и построим его методом шагов. В результате получаем, что $x_-(t+\tau) \to 0$ при $t \to \infty$.

Таким образом, поведение решений уравнения (3) при $t \to \infty$ полностью определяется поведением решений с начальными условиями из E_+.

Случаи $\omega \sim 1, 2$ рассматриваются аналогично.

Литература

[1] Анашкин О.В. Параметрический резонанс в линейной системе с запаздыванием//Известия РАЕН. Сер. МММИУ. Т. 1, №1. 1997. С. 3–17.

[2] Кубышкин Е.П. Параметрический резонанс в линейных системах с последействием// Исследования по устойчивости и теории колебаний. Ярославль, 1978. С. 43–76.

[3] Малкин И.Г. Некоторые задачи теории нелинейных колебаний. М : ГИТТЛ, 1956.

[4] Колесов Ю.С. Обзор результатов по теории устойчивости решений дифференциально-разностных уравнений с почти периодическими коэффициентами // Исследования по устойчивости и теории колебаний. Ярославль, 1977. С. 82–142.

Модел. и анализ информ. систем. Т.**11**, №**1** (2004) 32–38

УДК 519.22

Оценка дисперсии и робастность обобщенного γ-критерия

Янкевич А.П.
Ярославский государственный университет
150 000, Ярославль, Советская, 14

получена 15 марта 2004

В статье рассматривается обобщенный критерий однородности А.Ю. Левина. Отдельно изучаются дисперсия критериальной статистики при выполнении гипотезы однородности и робастность критерия. Получена оценка дисперсии при гипотезе однородности и показано, что с ростом параметра k, входящего в определение статистики, устойчивость критерия к ошибкам в статистических данных возрастает.

1. Введение

В работах [1, 2] А.Ю. Левиным был предложен новый мощный многомерный непараметрический критерий однородности — γ-критерий, основанный на отношении "быть ближайшим". Там же была доказана полная состоятельность критерия и показаны его положительные свойства, такие как нечувствительность к размерности, гибкость при получении инвариантных версий, робастность и др.

В работе [3] было рассмотрено одно из возможных обобщений γ-критерия, связанное с "расширением кругозора"статистики. Цель обобщения — при сохранении всех положительных свойств оригинальной идеи добиться большей устойчивости критерия к ошибкам в статистических данных, иными словами, в усилении свойства робастности.

Для обобщенного γ-критерия получено значение мат. ожидания критериальной статистики γ_n^{2k-1} при гипотезе однородности и подробно исследовано поведение мат. ожидания и дисперсии γ_n^{2k-1} в общем случае [3].

В данной работе после краткого описания обобщенного γ-критерия и определения величин, используемых в дальнейших рассуждениях, будет показано (§ 3.), что в случае однородности $\forall \varepsilon > 0$ $\mathbf{D}\gamma_n^{2k-1} \le Cn^{\varepsilon}$, где C — некоторая константа, не зависящая от n. В § 4. будет введена количественная характеристика робастности — $\delta_l^{(k)}$, отражающая силу влияния на статистику одного произвольного наблюдения, и показано, что с ростом параметра k, входящего в определение γ_n^{2k-1}, $\delta_l^{(k)}$ убывает.

2. Обобщенный γ-критерий

Напомним определение обобщенного γ-критерия [3].

Итак, пусть в $\mathbf{R}^N$ рассматриваются две независимые выборки

$$S_i = \{x_{i1}, x_{i2}, \ldots, x_{in_i}\}, \quad i = 1, 2,$$

с непрерывными плотностями p_1, p_2 соответственно. Проверке подлежит гипотеза однородности

$$H_0 : p_1(x) = p_2(x) \text{ всюду в } \mathbf{R}^N. \tag{1}$$

Обозначим через S объединенную выборку объема $n = n_1 + n_2$. Как обычно предполагается, что выборки равномощны, то есть с ростом n величины $n_1 n^{-1}, n_2 n^{-1}$ стремятся к каким-либо ненулевым пределам:

$$\frac{n_i}{n} \to s_i (\neq 0), \quad i = 1, 2 \quad (n \to \infty). \tag{2}$$

Зададим натуральный параметр k. Пусть $\theta_i \quad (0 \le \theta_i \le n_i)$ — число элементов выборки S_i, для которых из остальных $n-1$ элементов объединенной выборки S среди $2k-1$ ближайших k или более принадлежат той же выборке $S_i \quad (i = 1, 2)$.

Обобщенный γ-критерий основан на статистике

$$\gamma_n^{2k-1} = \sqrt{n}\left(\frac{\theta_1}{n_1} + \frac{\theta_2}{n_2} - 1\right). \tag{3}$$

Критерий гласит, что при заданном уровне значимости ε гипотеза H_0 отвергается, если посчитанное по экспериментальным данным значение γ_n^{2k-1} оказывается больше $\gamma_{cr} = \sqrt{C\varepsilon^{-1}}$. Здесь C — оценка $\mathbf{D}\gamma_n^{2k-1}$ свехру при гипотезе H_0.

3. Оценка $\mathbf{D}\gamma_n^{2k-1}$ при гипотезе H_0

Теорема 1. *Если справедлива гипотеза H_0, то для любого $\varepsilon > 0$ существует константа C, зависящая от ε и k, такая, что при любом n*

$$\mathbf{D}\gamma_n^{2k-1} \le Cn^{\varepsilon}. \tag{4}$$

Доказательство. Итак, пусть выполнена гипотеза H_0, т.е. все n элементов различных выборок S_l имеют одну и ту же плотность $p(x)$.

Через I_{li} обозначим индикатор события A_{li}, состоящего в том, что среди $2k-1$ ближайших к x_{li} элементов объединенной выборки S k или более принадлежат выборке S_l $(l = 1, 2)$.

Поскольку все x_{li} одинаково распределены и независимы, их совместное распределение перестановочно (т.е. симметрично). Поэтому из определения статистики γ_n^{2k-1} непосредственно следует, что

$$\mathbf{D}\gamma_n^{2k-1} = n\mathbf{D}\left(\frac{1}{n_1}\sum_{i=1}^{n_1} I_{1i} + \frac{1}{n_2}\sum_{i=1}^{n_2} I_{2i} - 1\right) =$$
$$\frac{\mathbf{D}I_{11}}{s_1} + \frac{\mathbf{D}I_{21}}{s_2} + \frac{1}{s_1}\sum_{i=2}^{n_1} Cov\left[I_{11}, I_{1i}\right] + \frac{1}{s_2}\sum_{i=2}^{n_2} Cov\left[I_{21}, I_{2i}\right] + \frac{2}{s_2}\sum_{i=1}^{n_2} Cov\left[I_{11}, I_{2i}\right]. \tag{5}$$

Оценим величины, входящие в выражение (5).

1. Предварительно докажем несколько вспомогательных утверждений.

1.1. Пусть $B(z, r)$ — шар радиуса r с центром в $z(\in \mathbf{R}^N)$, υ_N — объем единичного шара в $\mathbf{R}^N$, т.е. $\upsilon_N r^N$ — объем шара радиуса r.

Если x_{li} обладает плотностью распределения p, то для функции распределения $F(t, z)$ скалярной величины $\|x_{li} - z\|$ имеем (при $t > 0$)

$$F(t, z) = \int\limits_{B(z,t)} p(x)dx = p(z)\upsilon_N t^N + o(t^N) \quad (t \to 0).$$

Поэтому, очевидно, что существуют константы C_1, C_2 такие, что

$$C_1 t^N \le F(t, z) \le C_2 t^N \quad \forall t > 0. \tag{6}$$

В силу непрерывности плотность $p(x)$ ограничена сверху, поэтому $\forall z \in \mathbf{R}^N$ $0 \le C_1, C_2 \le C_{max}$.

1.2. Сформулируем простую лемму.

Лемма 1. *Пусть функция $F(t) = F(t, z)$ удовлетворяет (6), тогда при $\varepsilon_n = (\tau ln(n)/(C_1 n))^{1/N}$*

$$\int\limits_{\varepsilon_n}^{\infty} F(t)^{2k-2}(1 - F(t))^n dF(t) \le C_5 n^{-\tau}. \tag{7}$$

Доказательство леммы приведено в [4].

1.3. Положим

$$\rho_{li}^{(2k-1)} = \min_{m,j,h:m\neq l, j\neq i}{}^{(2k-1)}\{\|x_{lj} - x_{li}\|, \|x_{mh} - x_{li}\|\}, \tag{8}$$

где $min^{(k)}\{x_1, \ldots, x_n\} = x_k$, если $x_1 \le x_2 \le \cdots \le x_n$.

Через $\varphi_{n,2k-1}^{l}(u, y)$ обозначим плотность распределения случайной величины $\rho_{li}^{(2k-1)}$ при условии, что $x_{li} = y$ и в числе $2k-1$ ближайших к x_{li} элементов выборки S k или более принадлежат выборке S_l. (ясно, что распределение не зависит от i). В силу независимости элементов выборки S имеем

$$\varphi_{n,2k-1}^{l}(u, y) = \Psi_{n,2k-1}^{l} F(u, y)^{2k-2}(1 - F(u, y))^{n-2k}\frac{\partial F(u, y)}{\partial u}, \tag{9}$$

где

$$\Psi^1_{n,2k-1} = (n_1-1)\sum_{j=k-1}^{2k-2}\binom{n_1-2}{j}\binom{n_2}{2k-2-j} + n_2\sum_{j=k}^{2k-2}\binom{n_1-1}{j}\binom{n_2-1}{2k-2-j}, \quad (10)$$

$$\Psi^2_{n,2k-1} = n_1\sum_{j=0}^{k-2}\binom{n_1-1}{j}\binom{n_2-1}{2k-2-j} + (n_2-1)\sum_{j=0}^{k-1}\binom{n_1}{j}\binom{n_2-2}{2k-2-j}. \quad (11)$$

Понятно, что

$$\mathbf{E}\,[I_{li}|x_{li}=y] = \mathbf{P}\{A_{li}|x_{li}=y\} = \int_0^\infty \varphi^l_{n,2k-1}(u,y)du. \quad (12)$$

2. Оценим ковариации $Cov\,[I_{li}, I_{mj}]$.

Лемма 2. *Пусть функции $F(t,y)$, $F(t,z)$ удовлетворяют условию (6) с общими константами C_1, C_2, и пусть $|y-z| > 2n^{-(1-\varepsilon)/N}$, $\varepsilon > 0$. Тогда*

$$Cov\,[I_{li}, I_{mj}|x_{li}=y, x_{mj}=z] = \mathbf{E}\,[I_{li}I_{mj}|x_{li}=y, x_{mj}=z] - \mathbf{E}\,[I_{li}|x_{li}=y]\,\mathbf{E}\,[I_{mj}|x_{mj}=z] \le C_{lm}n^{-1}, \quad (13)$$

где константа C_{lm} зависит только от k (l,m=1,2).

Доказательство. Рассмотрим случай, когда шары радиусов u, v с центрами в точках y и z не пересекаются. Тогда плотность $\varphi^{l,m}_{n,2k-1}(u,v,y,z)$ совместного распределения случайных величин $\rho^{(2k-1)}_{li}$, $\rho^{(2k-1)}_{mj}$ при условии, что $x_{li}=y$, $x_{mj}=z$, в числе $2k-1$ ближайших к x_{li} элементов выборки S k или более принадлежат выборке S_l, а в числе ближайших к x_{mj} — выборке S_m, равна

$$\varphi^{l,m}_{n,2k-1}(u,v,y,z) = \Psi^{l,m}_{n,2k-1}F(u,y)^{2k-2}F(v,z)^{2k-2}(1-F(u,y)-F(v,z))^{n-4k}\frac{\partial F(u,y)}{\partial u}\frac{\partial F(v,z)}{\partial v}, \quad (14)$$

где $\Psi^{l,m}_{n,2k-1}$ — это число возможных вариантов расположения точек выборок S_1, S_2, при которых на границе первого шара лежит точка, на границе второго шара лежит точка, в первом шаре, включая его границу находится k или более точек выборки S_l, во втором шаре, включая границу, расположено k или более точек выборки S_m. Для случая $l=m=1$, в частности, имеем

$$\begin{aligned}\Psi^{1,1}_{n,2k-1} = {} & (n_1-2)(n_1-3)\sum_{j=k-1}^{2k-2}\binom{n_1-4}{j}\binom{n_2}{2k-2-j}\sum_{i=k-1}^{2k-2}\binom{n_1-4-j}{i}\binom{n_2-2k+2+j}{2k-2-i} + \\ & (n_1-2)n_2\sum_{j=k-1}^{2k-2}\binom{n_1-3}{j}\binom{n_2-1}{2k-2-j}\sum_{i=k}^{2k-2}\binom{n_1-3-j}{i}\binom{n_2-2k+1+j}{2k-2-i} + \\ & (n_1-2)n_2\sum_{j=k}^{2k-2}\binom{n_1-3}{j}\binom{n_2-1}{2k-2-j}\sum_{i=k-1}^{2k-2}\binom{n_1-3-j}{i}\binom{n_2-2k+1+j}{2k-2-i} + \\ & n_2(n_2-1)\sum_{j=k}^{2k-2}\binom{n_1-2}{j}\binom{n_2-2}{2k-2-j}\sum_{i=k}^{2k-2}\binom{n_1-2-j}{i}\binom{n_2-2k+j}{2k-2-i}. \quad (15)\end{aligned}$$

Если шары пересекаются, то справедлива следующая оценка:

$$\int_{u>\varepsilon_n}\int_{v>\varepsilon_n}\varphi^{l,m}_{n,2k-1}(u,v,y,z) = \mathcal{O}(n^{-n^c}),$$

где $\varepsilon_n = n^{-(1-\varepsilon)/N}$. Для ее получения воспользуемся неравенством $\varphi^{l,m}_{n,2k-1}(u,v,y,z) \le \min\{\varphi^l_{n,2k-1}(u,y), \varphi^m_{n,2k-1}(v,z)\}$ и применим лемму 1. Нетрудно видеть, что для параметра τ из этой леммы при $\varepsilon_n = n^{-(1-\varepsilon)/N}$ имеем $\tau = \mathcal{O}(n^c)$, где $0 < c < \varepsilon$.

Следовательно, при нахождении условных вероятностей $\mathbf{P}\{A_{li}A_{mj}|x_{li}=y, x_{mj}=z\}$ достаточно рассматривать шары с радиусом $< n^{-(1-\varepsilon)/N}$, которые не пересекаются.

Применяя для оценки распределения (14) то, что

$$(1-F(u,y)-F(v,z))^n \le (1-F(u,y))^n(1-F(v,z))^n,$$

получаем таким образом

$$\mathbf{P}\{A_{li}A_{mj}|x_{li}=y,x_{mj}=z\}=$$
$$\Psi^{l,m}_{n,2k-1}\int\limits_0^{\varepsilon_n}\int\limits_0^{\varepsilon_n}F(u,y)^{2k-2}F(v,z)^{2k-2}(1-F(u,y)-F(v,z))^{n-4k}d_uF(u,y)d_vF(v,z)+\mathcal{O}(n^{-n^c})\le$$
$$\Psi^{l,m}_{n,2k-1}\int\limits_0^{\infty}F(u,y)^{2k-2}(1-F(u,y))^{n-4k}d_uF(u,y)\int\limits_0^{\infty}F(v,z)^{2k-2}(1-F(v,z))^{n-4k}d_vF(v,z)+\mathcal{O}(n^{-n^c}). \quad (16)$$

Интегралы справа заменами $t=F(u,y)$ и $t=F(v,z)$ соответственно сводятся к бета-функции:

$$B(k,n)=\int\limits_0^1 t^{k-1}(1-t)^{n-1}dt=\frac{\Gamma(k)\Gamma(n)}{\Gamma(n+k)}=\frac{1}{k\binom{n+k-1}{k}}=B(k,n+p)+\mathcal{O}(n^{-k-1})=(k-1)!n^{-k}+\mathcal{O}(n^{-k-1}), \quad (17)$$

$0<k$, $p=o(n)$.

Используя то, что при $p=o(n)$ $\binom{n}{p}=\frac{n^p}{p!}+\mathcal{O}(n^{p-1})$, и делая замены $n_1=s_1n$, $n_2=s_2n$, нетрудно показать, что

$$\Psi^{l,m}_{n,2k-1}=\Psi^{l}_{n,2k-1}\Psi^{m}_{n,2k-1}+\mathcal{O}(n^{4k-3}). \quad (18)$$

При $l=m=1$ из (10), (15), в частности, имеем

$$\Psi^{1}_{n,2k-1}=\frac{n^{2k-1}}{(2k-2)!}\left(s_1\sum_{i=k-1}^{2k-2}\binom{2k-2}{i}s_1^is_2^{2k-2-i}+s_2\sum_{i=k}^{2k-2}\binom{2k-2}{i}s_1^is_2^{2k-2-i}\right)+\mathcal{O}(n^{2k-2}), \quad (19)$$

$$\Psi^{1,1}_{n,2k-1}=\frac{n^{4k-2}}{(2k-2)!^2}\left(s_1^2\sum_{i=k-1}^{2k-2}\binom{2k-2}{i}s_1^is_2^{2k-2-i}\sum_{j=k-1}^{2k-2}\binom{2k-2}{j}s_1^js_2^{2k-2-j}+\right.$$
$$2s_1s_2\sum_{i=k-1}^{2k-2}\binom{2k-2}{i}s_1^is_2^{2k-2-i}\sum_{j=k}^{2k-2}\binom{2k-2}{j}s_1^js_2^{2k-2-j}+$$
$$\left.s_2^2\sum_{i=k}^{2k-2}\binom{2k-2}{i}s_1^is_2^{2k-2-i}\sum_{j=k}^{2k-2}\binom{2k-2}{j}s_1^js_2^{2k-2-j}\right)+\mathcal{O}(n^{4k-3}). \quad (20)$$

Для других $\Psi^{l,m}_{n,2k-1}$ выражения повторяются с точностью до соответствующих замен пределов суммирования.

Объединяя (16), (17) и (18), получаем утверждение (13). Лемма 2 доказана. □

3. Теперь оценим дисперсию $\mathbf{D}\gamma_n^{2k-1}$. Вернемся к выражению (5). Так как случайные величины I_{li} — индикаторы, их мат. ожидания, дисперсии и ковариации ограничены:

$$\mathbf{E}I_{li}\le 1,\quad \mathbf{D}I_{li}\le 1/4,\quad Cov\,[I_{li},I_{lj}]\le 1/4,\quad Cov\,[I_{1i},I_{2j}]\le 1. \quad (21)$$

Для оценки сумм ковариаций в (5) выберем $\varepsilon>0$ и разобьем каждую сумму по i на две части:

$$I_1^1=\{i:\|x_{1i}-x_{11}\|\le n^{-(1-\varepsilon)/N}\},\quad I_1^2=\{i:\|x_{1i}-x_{11}\|>n^{-(1-\varepsilon)/N}\},$$
$$I_2^1=\{i:\|x_{2i}-x_{21}\|\le n^{-(1-\varepsilon)/N}\},\quad I_2^2=\{i:\|x_{2i}-x_{21}\|>n^{-(1-\varepsilon)/N}\},$$
$$I_3^1=\{i:\|x_{2i}-x_{11}\|\le n^{-(1-\varepsilon)/N}\},\quad I_3^2=\{i:\|x_{2i}-x_{11}\|>n^{-(1-\varepsilon)/N}\},$$

Пусть ν_j — число слагаемых в сумме I_j^1, то есть ν_j — число успехов в n_1-1, n_2-1, n_2 испытаниях соответственно с вероятностью успеха в одном испытании равной $F(n^{-(1-\varepsilon)/N})$. В силу (6) $\mathbf{E}\nu_j,\mathbf{D}\nu_j=\mathcal{O}(n^{\varepsilon})$, поэтому

$$\mathbf{P}\{\nu_j>C_3n^{2\varepsilon}\}\to 0\quad (n\to\infty),$$

где C_3 — некоторая константа, не зависящая от n.

Для оценки ковариаций в суммах I_1^1, I_2^1, I_3^1 применим (21), а для оценки ковариаций в суммах I_1^2, I_2^2, I_3^2 используем лемму 2:

$$\mathbf{D}\gamma_n^{2k-1} \le \frac{1}{4}\left(\frac{1}{s_1}+\frac{1}{s_2}\right) + \frac{1}{s_1}\sum_{i\in I_1^1}\frac{1}{4} + \frac{1}{s_1}\sum_{i\in I_1^2}\frac{C_{11}}{n} + \frac{1}{s_2}\sum_{i\in I_2^1}\frac{1}{4} + \frac{1}{s_2}\sum_{i\in I_2^2}\frac{C_{22}}{n} + \frac{2}{s_2}\sum_{i\in I_3^1}1 + \frac{2}{s_2}\sum_{i\in I_3^3}\frac{C_{12}}{n} \le$$
$$\frac{1}{4}\left(\frac{1}{s_1}+\frac{1}{s_2}\right) + C_{11} + 2C_{12} + C_{22} + C_3 n^{2\varepsilon}\left(\frac{1}{4s_1}+\frac{1}{4s_2}+\frac{2}{s_2}\right).$$

Так как ε может быть выбрано произвольным,отсюда очевидно вытекает (4). Теорема 1 доказана. □

4. Робастность обобщенного γ-критерия

Идея обобщения γ-критерия направлена на то, чтобы уменьшить влияние на статистику ошибок, содержащихся в статистических данных. Не секрет, что любые реальные (не модельные) экспериментальные данные содержат ошибки того или иного рода. Ошибки эти могут быть, в частности, следствием некоторых нарушений необходимых предположений о независимости и одинаковой распределенности элементов внутри выборок. При вычислении статистики γ_n^{2k-1} рассмотрение для каждого x_{li} более одного ближайшего элемента позволяет уменьшить влияние на вероятность события A_{li} любого из оставшихся $n-1$ элементов объединенной выборки S (событие A_{li} состоит в том, что среди $2k-1$ ближайших к x_{li} элементов выборки S k или более принадлежат выборке S_l $(l=1,2)$). С ростом k событие A_{li} становится суммарным результатом все большего и большего числа наблюдений, поэтому влияние каждого из них в отдельности уменьшается. Для строгости приведем ряд рассуждений, подтверждающих этот интуитивно понятный факт.

Будем придерживаться *инфинитезимального* подхода к количественной оценке робастности статистической процедуры, предложенного Ф. Хампелем [5] и связанного с оценкой введенной им функции влияния.

Предлагается в качестве оценки силы влияния одного произвольного наблюдения x_{mj} на вероятность наступления события A_{li} рассмотреть величину

$$\delta_l^{(k)} = \max_m R_{lm}^{(k)} - \min_m r_{lm}^{(k)}, \tag{22}$$

где

$$R_{lm}^{(k)} = \max_{z\in\mathbf{R}^N}\mathbf{P}\{A_{li}|x_{mj}=z\}, \quad r_{lm}^{(k)} = \min_{z\in\mathbf{R}^N}\mathbf{P}\{A_{li}|x_{mj}=z\}, \quad (x_{li}\ne x_{mj}).$$

Величине $R_{lm}^{(k)}$ $(r_{lm}^{(k)})$ при $l=m$ будет соответствовать вероятность наступления события A_{li} при условии, что x_{mj} достоверно содержится (не содержится) в числе $2k-1$ ближайших к x_{li}, и наоборот для случая $l\ne m$.

Теорема 2. *Если справедлива гипотеза H_0, то функция влияния $\delta_l^{(k)}$ с ростом k убывает.*

Доказательство. Предположим, что гипотеза H_0 верна. Через $p_l^{(k)}$ обозначим $\mathbf{P}\{A_{li}\}$ $(l=1,2)$, значения которых согласно [3] равны

$$p_1^{(k)} = \frac{\sum\limits_{j=k}^{2k-1}\binom{n_1-1}{j}\binom{n_2}{2k-1-j}}{\binom{n-1}{2k-1}}, \quad p_2^{(k)} = \frac{\sum\limits_{j=0}^{k-1}\binom{n_1}{j}\binom{n_2-1}{2k-1-j}}{\binom{n-1}{2k-1}}. \tag{23}$$

Для любого элемента первой выборки $(l=1)$

$$R_{11}^{(k)} = \frac{\sum\limits_{j=k}^{2k-1}\binom{n_1-2}{j-1}\binom{n_2}{2k-1-j}}{\binom{n-2}{2k-2}}, \quad R_{12}^{(k)} = \frac{\sum\limits_{j=k}^{2k-1}\binom{n_1-1}{j}\binom{n_2-1}{2k-1-j}}{\binom{n-2}{2k-1}}, \tag{24}$$

$$r_{11}^{(k)} = \frac{\sum\limits_{j=k}^{2k-1}\binom{n_1-2}{j}\binom{n_2}{2k-1-j}}{\binom{n-2}{2k-1}}, \quad r_{12}^{(k)} = \frac{\sum\limits_{j=k}^{2k-2}\binom{n_1-1}{j}\binom{n_2-1}{2k-2-j}}{\binom{n-2}{2k-2}}. \tag{25}$$

Для оценки $\delta_1^{(k)}$ рассмотрим по отдельности величины $R_{1m}^{(k)} - p_1^{(k)}$ и $p_1^{(k)} - r_{1m}^{(k)}$ $(m=1,2)$.

$$R_{11}^{(k)} - p_1^{(k)} = \frac{\sum\limits_{j=k}^{2k-1} \binom{n_1-2}{j-1}\binom{n_2}{2k-1-j}}{\binom{n-2}{2k-2}} - \frac{\sum\limits_{j=k}^{2k-1} \binom{n_1-1}{j}\binom{n_2}{2k-1-j}}{\binom{n-1}{2k-1}}.$$

Воспользуемся тем, что $\binom{n-1}{m-1} = \frac{m}{n}\binom{n}{m}$, приведем слагаемые к одному знаменателю и объединим суммы:

$$R_{11}^{(k)} - p_1^{(k)} = \frac{\sum\limits_{j=k}^{2k-1} \left(\frac{n-1}{2k-1}\cdot\frac{j}{n_1-1} - 1\right)\binom{n_1-1}{j}\binom{n_2}{2k-1-j}}{\binom{n-1}{2k-1}}.$$

Поскольку при $m \ll n$ $\binom{n}{m} \approx \frac{n^m}{m!}$, делая замены $s_l = n_l/n$ (не забываем, что $s_1+s_2=1$) и затем $i = 2k-1-j$, имеем

$$R_{11}^{(k)} - p_1^{(k)} \approx \frac{\sum\limits_{j=k}^{2k-1} \left(\frac{1}{s_1}\cdot\frac{j}{2k-1} - 1\right)\frac{n_1^j}{j!}\frac{n_2^{2k-1-j}}{(2k-1-j)!}}{n^{2k-1}/(2k-1)!} = \sum_{j=k}^{2k-1}\left(\frac{1}{s_1}\cdot\frac{j}{2k-1} - 1\right)\binom{2k-1}{j}s_1^j s_2^{2k-1-j} =$$

$$= \frac{s_1^{k-1}}{2k-1}\sum_{i=0}^{k-1}((2k-1)s_2 - i)\binom{2k-1}{i}s_1^{k-1-i}s_2^i.$$

Далее, под знаком суммы сделаем замену $s_1^{k-1-i} = \sum_{j=0}^{k-1-i}(-1)^j\binom{k-1-i}{j}s_2^j$ и сгруппируем слагаемые при одинаковых степенях s_2:

$$R_{11}^{(k)} - p_1^{(k)} \approx \frac{s_1^{k-1}}{2k-1}\times\left(\sum_{m=1}^{k}s_2^m\sum_{i=0}^{m-1}(2k-1)\binom{2k-1}{i}(-1)^{m-1-i}\binom{k-1-i}{m-1-i} - \right.$$

$$\left. - \sum_{m=1}^{k-1}s_2^m\sum_{i=1}^{m}i\binom{2k-1}{i}(-1)^{m-i}\binom{k-1-i}{m-i}\right). \quad (26)$$

В выражении, стоящем в скобках, рассмотрим отдельно коэффициенты при s_2 в некоторой фиксированной степени m меньше k, умножив их для удобства на $(-1)^{2i-m+1}$. Затем используем свойство сочетаний $\binom{n+1}{m+1} = \binom{n}{m} + \binom{n}{m+1}$ и преобразуем первую сумму, а во второй сумме выполним замену индексов $j = i-1$:

$$\sum_{i=0}^{m-1}(-1)^i(2k-1)\binom{2k-1}{i}\binom{k-1-i}{m-1-i} + \sum_{i=1}^{m}(-1)^i i\binom{2k-1}{i}\binom{k-1-i}{m-i} =$$

$$\sum_{i=0}^{m-1}(-1)^i(2k-1)\left(\binom{2k-2}{i} + \binom{2k-2}{i-1}\right)\binom{k-1-i}{m-1-i} - \sum_{j=0}^{m-1}(-1)^j(j+1)\binom{2k-1}{j+1}\binom{k-2-j}{m-1-j} =$$

$$\sum_{i=0}^{m-1}(-1)^i(2k-1)\binom{2k-2}{i}\left(\binom{k-1-i}{m-1-i} - \binom{k-2-i}{m-2-i}\right) -$$

$$- \sum_{j=0}^{m-1}(-1)^j(2k-1)\binom{2k-2}{j}\binom{k-2-j}{m-1-j} = 0.$$

Таким образом, в (26) слагаемые при всех $m < k$ сокращаются и остается единственное слагаемое из первой суммы при $m = k$. Используя тот факт, что $(-1)^n\binom{2n}{n} = \sum_{j=0}^{n}(-1)^j\binom{2n+1}{j}$, который легко подтвердить, представив каждое слагаемое в выражении справа в виде суммы $\binom{2n}{j} + \binom{2n}{j-1}$, получаем в итоге

$$R_{11}^{(k)} - p_1^{(k)} \approx s_1^{k-1}s_2^k\sum_{i=0}^{k-1}(-1)^{k-1-i}\binom{2k-1}{i} = \binom{2k-2}{k-1}s_1^{k-1}s_2^k. \quad (27)$$

Далее, оценим величину $p_1^{(k)} - r_{12}^{(k)}$. В выражении для $r_{12}^{(k)}$ из (25) сделаем замену индексов $i = 2k-2-j$ и в силу того, что $\sum\limits_{j=0}^{k} \binom{n}{j}\binom{m}{k-j} = \binom{n+m}{k}$, получаем

$$r_{12}^{(k)} = \frac{\sum\limits_{i=0}^{k-2} \binom{n_1-1}{2k-2-i}\binom{n_2-1}{i}}{\binom{n-2}{2k-2}} = \frac{\binom{n-2}{2k-2} - \sum\limits_{i=k-1}^{2k-2} \binom{n_1-1}{2k-2-i}\binom{n_2-1}{i}}{\binom{n-2}{2k-2}} = 1 - \frac{\sum\limits_{j=k}^{2k-1} \binom{n_1-1}{2k-1-j}\binom{n_2-1}{j-1}}{\binom{n-2}{2k-2}}.$$

С выражением для $p_1^{(k)}$ поступим аналогичным образом, сделав замену индексов $i = 2k-1-j$. Отсюда

$$p_1^{(k)} - r_{12}^{(k)} = \frac{\sum\limits_{j=k}^{2k-1} \binom{n_1-1}{2k-1-j}\binom{n_2-1}{j-1}}{\binom{n-2}{2k-2}} - \frac{\sum\limits_{i=k}^{2k-1} \binom{n_1-1}{2k-1-i}\binom{n_2}{i}}{\binom{n-1}{2k-1}}.$$

Очевидно, что оценка величины $p_1^{(k)} - r_{12}^{(k)}$ сводится к предыдущей задаче, и поэтому из (27) в силу симметрии получаем $p_1^{(k)} - r_{12}^{(k)} \approx \binom{2k-2}{k-1} s_1^k s_2^{k-1}$.

В свете изложенного нетрудно показать, что величины $R_{12}^{(k)} - p_1^{(k)}$ и $p_1^{(k)} - r_{11}^{(k)}$ близки к нулю при $n \gg 1$. Таким образом

$$\delta_l^{(k)} = R_{11}^{(k)} - r_{12}^{(k)} \approx \binom{2k-2}{k-1} s_1^{k-1} s_2^{k-1}.$$

Поскольку $\binom{2k}{k} = 2\binom{2k-1}{k-1} = \frac{2(2k-1)}{k}\binom{2k-1}{k-1}$, $\delta_l^{(k+1)}/\delta_l^{(k)} \approx (4 - \frac{2}{k}) s_1 s_2 < 1$, то есть $\delta_l^{(k)}$ с ростом k убывает. Теорема доказана. □

Поясним полученный результат. Итак, показано, что если одно из n наблюдений содержит ошибку, то ее влияние на события A_{li} и, следовательно, на статистику γ_n^{2k-1} в целом с ростом k будет убывать. Данное утверждение может быть обобщено на случай m (> 1) ошибок. Таким образом, при фиксированном n увеличение k (при сохранении соотношения $k \ll n$) обеспечивает усиление устойчивости статистики к небольшим отклонениям от исходной статистической модели, либо к малому числу грубых ошибок в статистических данных, или попросту к усилению робастности.

На практике рекомендуется наблюдать изменения значений статистики с ростом параметра k. Эти изменения должны оказываться незначительными. В противном случае следует отнестись к полученным результатам очень настороженно, и прежде чем принимать решение об отклонении или принятии гипотезы H_0, стоит постараться найти причины наблюдаемой тенденции.

Литература

1. Левин А.Ю. О состоятельном многомерном непараметрическом критерии однородности // Усп. матем. наук. 1993. Т. 48, N 6. С. 155–156.

2. Левин А.Ю. О состоятельном многомерном непараметрическом критерии однородности // Модел. и анализ информ. систем. 2002. Т. 9, N 2. С. 32–44.

3. Янкевич А.П. Об обобщении критерия однородности А.Ю. Левина // Модел. и анализ информ. систем. 2003. Т. 10, N 2. С. 50–59.

4. Майоров В.В., Тимофеев Е.А. Статистическая оценка обобщенных размерностей // Мат. заметки. 2002. Т. 71, N 5. С. 697–712.

5. Хампель Ф. [и др.] Робастность в статистике. Подход на основе функций влияния: Пер. с англ. М., 1989.

Модел. и анализ информ. систем. Т.11, №1 (2004) 39–47

УДК 541.1+612.82

Разработка диспетчера подключений к базам данных в .NET Framework

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

получена 21 февраля 2004

1. Постановка задачи

Большинство современных прикладных программ опирается в своей работе на базы данных. Чаще всего база одна, но нередки и приложения, использующие несколько баз. При этом множество используемых баз данных может быть либо известно уже на этапе разработки приложения, либо динамически формироваться во время выполнения.

Для таких приложений актуальна задача разработки программного компонента, который предоставлял бы доступ к объектам подключения к БД по логическим именам. Компонент должен корректно вести себя в многопоточной среде, а также предоставлять возможность безопасно открывать и автоматически закрывать подключения.

В статье предлагается и обосновывается архитектура такого компонента - диспетчера подключений (разделы 2-6). В частности, предлагается несложный способ его конфигурирования (раздел 5). Далее обсуждаются преимущества и недостатки, связанные с повторным использованием объектов подключений и команд. Озвучиваются дополнительные требования, которые это накладывает на функционирование диспетчера (разделы 7-9).

Рассмотрены проблемы, возникающие при использовании диспетчера в многопоточных приложениях и дан один из вариантов их решения (разделы 10-11).

В конце статьи описывается удобный для разработчика способ открывать и закрывать подключения к базе данных, который позволяет без дополнительных усилий переключаться между режимами функционирования диспетчера (раздел 12).

Необходимо отметить, что подобные компоненты присутствуют во многих программных платформах (см., например, [2], [3]), и, скорее всего, некое аналогичное решение появится в следующих версиях .NET Framework.

2. Место диспетчера в приложении

В данной статье мы не будем рассматривать, что именно приложение делает с базой данных [1]. Для нас важен сам факт подключения для выполнения каких бы то ни было операций. Рассмотрим традиционный для ADO.NET сценарий работы с базой.

1. Должна быть известна строка подключения к искомой базе (connection string). Эта строка содержит всю информацию, необходимую инфраструктуре ADO.NET для того, чтобы успешно подключиться.

2. Мы должны создать объект подключения соответствующего типа. Объекты разных типов обслуживают различные СУБД, так что знать нужный тип объекта не менее важно, чем строку подключения.

3. Созданный объект подключения нужно проинициализировать имеющейся у нас строкой подключения.

4. Теперь мы можем установить подключение ("открыть" его) и начать общение с СУБД.

5. По завершении сессии объект подключения следует уничтожить.

Выглядит это примерно так:

```
string conString = "...";
SqlConnection con = new SqlConnection();
con.ConnectionString = conString;
using( con ){
```

```
con.Open();
...
} // В этой точке подключение будет автоматически закрыто и уничтожено
```

Код прост и логичен, но в реальном приложении с ним могут возникнуть проблемы:

- Почти всегда разработка приложения ведется не с теми строками подключения, которые будут использоваться при эксплуатации. Соответственно, мы не можем жестко зашить строку подключения в код, а должны как-то узнать ее во время выполнения.
- В ряде приложений разработчик не знает конкретного типа объекта подключения и работает с базовым интерфейсом IDbConnection. Для таких случаев код типа "new SqlConnection" не годится. В то же время разработчик обычно четко представляет, какое подключение ему нужно открыть, и может логически обозначить их "база А" или "база Б", что бы это ни означало в среде конечного пользователя. В случае приложения с одной базой можно подключаться и к некоторой базе по умолчанию, никак ее не именуя.

Очевидно, что удобным решением могло бы стать использование некоторого механизма, позволяющего получать объект подключения к базе данных по его логическому имени. Конечному пользователю этот механизм должен предоставлять возможность быстро и просто ассоциировать логическое название с реальной строкой подключения. Например, это можно делать в файле конфигурации. Подобный механизм мы и назовем "диспетчер подключений к базам данных".

3. Основная функция диспетчера

Основная функция диспетчера - по заданному логическому имени вернуть объект подключения нужного типа, проинициализированный нужной строкой подключения. В использовании это может выглядеть так:

```
SqlConnection c1 = (SqlConnection)dbmgr["beta"];
IDbConnection c2 = dbmgr.Default;
```

Приведение типа в первой строке обусловлено тем, что наше приложение может работать с базами разных типов, и, следовательно, диспетчер не может возвращать объект подключения какого-то определенного типа. Так как любой объект подключения должен реализовывать интерфейс IDbConnection, диспетчеру наиболее логично давать доступ к объектам именно через этот интерфейс. Очевидно, что эта функция диспетчера примерно соответствует шаблону проектирования Factory Method [4].

4. Перечисление подключений

Мы уже говорили о приложениях с неопределенным на этапе разработки количеством подключений. В практике применения диспетчера подключений это может выглядеть, например, так: программа должна последовательно получить информацию из каждой базы, зарегистрированной в диспетчере. При этом на момент написания программы мы не знаем, какие базы и в каком количестве будут нужны конечному пользователю. Очевидно, что в этом случае логические имена баз нас не очень-то интересуют. Гораздо больше нам нужна возможность перебора всех баз в диспетчере. Например:

```
foreach( IDbConnection con in dbmgr ) {
// Получаем информацию
...
}
```

Для того чтобы эта языковая конструкция работала, да и вообще для перебора всех имеющихся подключений, наш класс DbManager должен реализовывать интерфейс IEnumerable.

5. Конфигурирование диспетчера

Настройка диспетчера заключается в установлении соответствия между логическим именем подключения и информацией, достаточной для создания объекта подключения. Несложные размышления показывают,

что достаточно знать тип объекта подключения, строку подключения, а также указание, является ли данное подключение подключением по умолчанию или нет.

Как уже указывалось, проще всего эту настройку делать через конфигурационный файл приложения. При этом и разработчик имеет единое место для описания подключений, и конечный пользователь может быстро произвести адаптацию приложения к своим условиям. Нужно отметить, впрочем, что при этом подходе конечный пользователь должен знать стандартный формат строк подключения ADO.NET.

Примерный код, инструктирующий диспетчера произвести чтение настроечных данных, даже если он уже был сконфигурирован:

dbmgr.Configure(true); // forceReload = true

При этом формат секции конфигурационного файла может быть таким:

```
<Database>
<connection name="alfa"
connectionString="..."
default=true" />
<connection name="beta"
connectionString="..."
type="OleDbConnection" />
</Database>
```

Подобные конфигурационные секции являются вполне обычным решением и широко применяются в самых разных приложениях (см. [5], [6]).

Здесь декларируется, что приложение использует две базы данных. Первая из них называется alfa, обслуживается объектом типа SqlConnection (ибо ничего другого не указано), и является подключением по умолчанию. Вторая носит логическое имя beta и обслуживается объектом типа OleDbConnection. Безусловно, для обеих баз указаны и корректные строки подключений.

Имея простой и удобный способ описания подключений через конфигурационный файл, мы, тем не менее, не должны забывать, что бывают ситуации, когда все это должно быть сделано программным путем. Например, так:

```
Configuration config = new Configuration();
// Настраиваем объект config
...
// Назначаем конфигурацию диспетчеру
DbManager.Configure( config );
```

В данном случае объект типа Configuration предоставляет нам те же возможности настройки, что и файл конфигурации.

Очень тяжело представить приложение, в котором существовало бы несколько отдельных наборов подключений к базам данных. Я говорю, например, о ситуации, когда в двух разных местах приложения мы используем два разных подключения с именем beta. Какие выводы из этого следуют?

Во-первых, это значит, что все экземпляры диспетчера подключений, используемые в приложении, должны быть сконфигурированы одинаково. Соответственно, методы Configure(:) мы смело можем делать статическими.

Во-вторых, напрашивается вывод, что мы вполне можем обойтись одним экземпляром диспетчера на все приложение. В некоторых случаях, о которых мы поговорим позже, нам понадобится большее, но все же ограниченное количество экземпляров. Из этого следует, что экземпляр диспетчера мы должны получать не при помощи оператора new, а посредством некоего статического метода класса. Пример:

DbManager dbmgr = DbManager.Get();

Подобный подход напоминает о шаблоне проектирования Singleton [4], но, в отличие от классической трактовки, у нас может быть не один экземпляр, а несколько. Более подробно это освещено в разделе 10.

6. Структура класса

Продумав сценарий использования диспетчера, мы можем спроектировать структуру класса. Вот она:

```
public class DbManager : IEnumerable
{
```

```
    public static DbManager Get() {...}
    public IDbConnection this[string name]
    {
      get  {...}
    }
    public IDbConnection Default
    {
      get  {...}
    }
    public static void Configure( bool forceReload ) {...}
    public static void Configure( Configuration config ) {...}
    public IEnumerator GetEnumerator() {...}
    // Непубличные методы и члены класса
...
  }
```

Приведем краткое описание методов.

- **Get** - возвращает диспетчер подключений. Если экземпляра диспетчера еще нет, создается новый.
- **this[string]** – возвращает объект подключения по данному логическому имени. В том случае, если имя не указано (равно null), возвращается объект подключения по умолчанию.
- **Default** – возвращает объект подключения по умолчанию.
- **Configure(bool)** – читает настроечную информацию из конфигурационного файла. Если мы пытаемся работать с еще не сконфигурированным диспетчером, он должен автоматически вызвать этот метод.
- **Configure(Configuration)** – настраивает диспетчер в соответствии с данным конфигурационным объектом.
- **GetEnumerator** – позволяет пробежаться по всем подключениям диспетчера циклом foreach.

7. Варианты работы с базой

Мы уже рассматривали кусок типового кода, работающего с базой. Более полный фрагмент выглядит так: мы сначала создаем подключение (например, SqlConnection), потом создаем команду (SqlCommand), добавляем к команде параметры, ассоциируем ее с подключением, открываем подключение, выполняем команду, закрываем подключение:

```
SqlConnection con = new SqlConnection();
con.ConnectionString = "...";
SqlCommand cmd = new SqlCommand();
cmd.CommandText = "...";
cmd.Connection = con;
cmd.Parameters.Add( new SqlParameter( ... ) );
using( con ){
con.Open();
cmd.Execute();
...
}
```

Мы делаем это при каждом обращении к базе, так что возникает вопрос: а не будет ли быстрее заранее создать и сохранить подключение и команду, а потом только использовать их? С точки зрения элементарной логики кажется очевидным, что должно быть быстрее. С другой стороны, известно, что создание объектов в .NET Framework происходит очень быстро, так что выигрыш вряд ли будет большим.

Проведем тест. В одном прогоне мы будем каждый раз создавать подключение и команду, а в другом - использовать готовые объекты. Команде определим три параметра. В двух прогонах по 100 000 итераций

удалось выяснить следующее: первый подход, при котором все создается заново, примерно на 5 процентов медленнее второго.

В абсолютном исчислении это замедление составляет всего 0.08 миллисекунды на каждую итерацию, т.е. оно очень мало. Если учесть, что само обращение к базе выполняется на несколько порядков медленнее создания любого объекта, то выигрыш получается и вовсе умозрительный.

Какие следуют выводы? Во-первых, логика восторжествовала - не создавать объекты оказалось быстрее, чем создавать. Во-вторых, это совершенно не важно. Разница в скорости между пересозданием объектов и использованием готовых настолько мала, что разработчик может смело выбирать тот или иной подход, руководствуясь только своим личным пониманием прекрасного.

Говоря же о практической экономии, можно сделать такую оценку: если у нас есть некая динамическая web-страница, которая делает одно обращение к базе, а к ней самой обращаются 10 раз в секунду, то сохранение объектов поможет нам выиграть целую секунду за полчаса.

Естественно, между этими двумя полярными вариантами есть большое число промежуточных состояний. Например, можно каждый раз создавать объект подключения, но хранить готовые команды. Этот подход, вероятно, весьма органично сочетается с визуальным дизайнером компонентов из Visual Studio. Набросав на компонент команд, мы получаем код для их инициализации, который выполняется при создании экземпляра компонента. Очевидно, что извлекать этот код из метода InitializeComponent неразумно, а лучше просто назначить нужной команде тот объект подключения, который мы собираемся открывать в данный момент.

8. Повторное использование подключений

В то время как с повторным использованием командных объектов (SqlCommand, OleDbCommand и т.п.) все, в общем, понятно, вопрос повторного использования объекта подключения остается открытым. Нужно ли это кому-нибудь, а если нужно, то зачем?

Под "повторным" мы здесь понимаем такое использование, когда один и тот же объект подключения используется снова и снова во всех частях приложения, где нужен доступ к соответствующей базе данных. При этом мы сознаем, что все стандартные для ASP.NET объекты подключения не являются безопасными для многопоточного использования (non thread safe), поэтому для начала будем считать, что наше приложение имеет только один поток.

Какие проблемы у нас могут возникнуть при подобном использовании объекта подключения? Во-первых, каждый раз, открывая подключение к базе данных, мы можем обнаружить, что оно уже было открыто раньше. Попытка открытия уже открытого подключения вызывает ошибку. Во-вторых, открытый объект DataReader блокирует свое подключение, так что до его закрытия выполнить еще какую-либо команду невозможно. Это может создать проблему в методе, вызванном во время чтения данных из базы.

Первую проблему обойти несложно. Достаточно проверять состояние подключения перед открытием и пропускать этот шаг, если оно уже открыто. Здесь важно заметить, что метод, открывший подключение, обязательно должен его закрыть, так что, если проверка показала, что подключение нужно открывать, это значит и то, что его нужно закрыть после использования.

Обойти вторую проблему в том месте, где она дала о себе знать, невозможно. Действительно, данные уже читаются, подключение уже заблокировано, и сделать мы с этим ничего не можем. Единственное решение здесь - так проектировать блоки чтения, чтобы они даже потенциально не могли никого блокировать. Например, можно сначала прочитать все данные в массив, а уже потом проводить их дальнейшую обработку. Основные проблемы ясны, и они достаточно серьезны. Какие преимущества могут быть у данного подхода? Перевешивают ли они недостатки? Во-первых, мы можем существенно ускорить работу в тех приложениях, где свежеоткрытое подключение нужно специально готовить. Например, приложение может использовать application roles. Для входа в роль MS SQL Server требует выполнения хранимой процедуры `sp_setapprole`:

```
EXEC sp_setapprole 'SalesApprole', 'AsDeFXX'
```

Очевидно, что если обработка запроса состоит, к примеру, из пяти обращений к базе, то гораздо быстрее будет открыть подключение и выполнить эту команду один раз, нежели все пять. Сама операция открытия подключения требует очень мало времени - на это есть connection pooling. Лишнее же обращение к базе - это серьезный удар по быстродействию. Естественно, я говорю не о простейшем случае, когда все пять обращений к базе находятся в одном методе. В конце концов, мы живем во времена победившего объектно- ориентированного подхода, так что "макаронный" код почти почил в бозе. Все эти обращения совершаются разными компонентами, обслуживающими запрос. Как быть в этом случае? Предлагается

открыть подключение и выполнить эту команду в начале обработки запроса, а затем передать объект подключения в дальнейшее использование.

Представляется, что это преимущество выглядит достаточно серьезным. Конечно, для определенного класса приложений. Кстати, здесь мы должны обратить внимание еще на одну особенность: войдя в роль и рассчитывая на автоматический выход из нее по закрытию подключения, можно получить неприятный сюрприз в том случае, если подключение на самом деле закрыто не будет. Последующие обращения к базе, возможно, будут выполняться с несвойственными правами. С другой стороны, это может случиться только в приложении с весьма специфической архитектурой. Во-вторых, можно представить себе приложение, открывающее без меры много подключений. Большущая вложенность вызовов. Может быть, даже рекурсия. Все методы открывают подключения и, не закрывая, вызывают другие методы. В таком (совершенно гипотетическом) приложении мы можем столкнуться с тем, что свободные подключения закончатся, и в какой-то момент времени мы не сможем открыть подключение к базе. Использование одного объекта подключения могло бы нас здесь спасти. Впрочем, подобная проблема выглядит совершенно надуманной. Если она и имеет где-то место, то это, скорее, ошибка в проектировании приложения, и решать ее нужно другими способами.

9. Режимы функционирования диспетчера

Вернемся к диспетчеру подключений. Очевидно, что он мог бы функционировать в двух режимах: либо каждый раз создавать новый объект подключения, либо возвращать уже готовый экземпляр, соответствующий заданному логическому имени.

На практике диспетчер, осуществляющий кэширование объектов подключения, должен успешно проходить вот такой тест:

```
DbManager.Mode = DbManagerMode.CacheConnections;
DbManager dbmgr =DbManager.Get();
IDbConnection c1 = dbmgr["beta"];
IDbConnection c2 = dbmgr["beta"];
Assert.IsTrue( c1 == c2 ); // Диспетчер возвращает один и тот же экземпляр
```

"Простой" диспетчер, т.е. не осуществляющий кэширование, выглядит следующим образом:

```
DbManager.Mode = DbManagerMode.DoNotCacheConnections;
DbManager dbmgr =DbManager.Get();
IDbConnection c1 = dbmgr["beta"];
IDbConnection c2 = dbmgr["beta"];
Assert.IsTrue( c1 != c2 ); // Диспетчер возвращает разные экземпляры
```

Подобная функциональность позволила бы разработчику легко выбирать между двумя описанными выше вариантами работы с подключениями, и даже без больших сложностей перейти с одного подхода на другой в частично уже написанном приложении.

10. Многопоточность

Ранее мы рассматривали приложение, в котором есть только один поток (thread). Какие сложности могут преследовать нас, если потоков будет несколько? По сути, сложность здесь только одна - один объект подключения можно одновременно использовать только в одном потоке. Если мы на каждое обращение создаем новый объект подключения, то нас эта проблема совершенно не касается. Соответственно, диспетчер подключений, работающий в этом режиме, очень прост в реализации. Метод, возвращающий объект подключения по имени, можно написать так:

```
public override IDbConnection this[String name]
{
  get
  {
    ConnectionInfo info = (ConnectionInfo)_config.Connections[name];
    if( info != null )
    {
```

```
        return CreateConnection( info );
      }
      return null;
    }
  }
```

Несколько более сложно придется поступить при кэшировании объектов подключения. С одной стороны, мы должны организовать некий словарь готовых объектов, из которого мы будем отдавать объекты по запросу. С другой стороны, мы должны сделать так, чтобы каждый поток работал со своим экземпляром данного подключения. Пытаться реализовать такую функциональность в одном объекте, обслуживающем сразу все потоки, может быть слишком сложно. Поэтому предлагается сделать так, чтобы каждый экземпляр диспетчера обслуживал только один поток. Очевидно, что экземпляров тогда будет не один, а некоторое неопределенное количество, не большее, чем общее количество потоков в приложении. Неопределенность обусловлена тем, что для тех потоков, в которых диспетчер не требуется, создавать экземпляр не нужно.

Таким образом, нам нужно, во-первых, написать такой метод Get, который бы возвращал экземпляр, приписанный к вызывающему потоку, и, во-вторых, сделать словарь готовых объектов. Вот приблизительный фрагмент кода:

```
[ThreadStatic] private static DbManager _instance;
private ListDictionary _connections = new ListDictionary();
internal static new DbManager Get()
{
 // Если экземпляр уже есть, вернуть его
 if( _instance != null ) return _instance;
 // Создать новый экземпляр
 _instance = new CachingDbManager();
 _instance.Init();
 return _instance;
}
public override IDbConnection this[String name]
{
  get
  {
   // Пытаемся взять готовый объект из словаря
   IDbConnection result = (IDbConnection)_connections[name];
   if( result == null )
   {
    // мщем описание подключения в конфигурации
    ConnectionInfo info = (ConnectionInfo)_config.Connections[name];
    if( info != null )
    result = CreateConnection( info );
   }
   return result;
  }
}
```

11. Диспетчер и ASP.NET

Как известно, приложения ASP.NET весьма активно используют многопоточность. В то же время делают они это настолько неявно, что этот факт легко оставить без внимания и получить неожиданные ошибки.

Вспомним, в общих чертах, структуру обычного приложения ASP.NET. В домене приложения (AppDomain) есть несколько экземпляров класса HttpApplication. Каждый из этих экземпляров обладает набором сопутствующих ему модулей (HttpModule). Набор модулей у каждого приложения одинаков, да и сами приложения, по идее, не должны ничем отличаться. Далее, домен приложения имеет набор рабочих потоков (working threads), готовых обслуживать пользовательские запросы. Со всей очевидностью, потоков существует по крайней мере столько же, сколько объектов Http-приложения.

При обслуживании запроса, ASP.NET неким псевдослучайным образом выбирает рабочий поток и объект приложения, с которым этот поток будет работать. В связи с этим определенный объект приложения в разных запросах будет, скорее всего, работать с разными потоками. При большой загрузке сервера даже возможна ситуация, когда рабочий поток меняется в процессе обработки запроса.

Предположим теперь, что в нашем приложении мы создали командный объект (SqlCommand) и сохранили его для дальнейшего использования. Команда связана с определенным объектом подключения, а именно с тем объектом, который был возвращен диспетчером подключений в момент создания и первого выполнения команды. Не будем, однако, забывать, что данный объект HttpApplication при обслуживании следующего (в его хронологии) запроса, скорее всего, будет работать уже с другим рабочим потоком, а поэтому диспетчер подключений вернет не то подключение, с которым связана наша команда. Хуже того, с возвращенным подключением, вероятно, будет связана аналогичная команда в другом объекте приложения.

Выход из описанной проблемы достаточно прост. Необходимо сделать такой модуль (HttpModule), который в начале обработки запроса будет связывать диспетчер подключений, приписанный к данному объекту приложения, с контекстом обработки запроса (HttpContext). Очевидно, что в этом случае методы Get и Init в своей работе должны будут полагаться не на текущий поток, а на текущий контекст. Это искоренит все проблемы такого рода и позволит опять забыть про реальное положение дел с потоками в ASP.NET.

Код модуля предельно прост:

```
public class AspAdapter : IHttpModule
{
  private HttpApplication application;
  private DbManager manager;
  public void Init(System.Web.HttpApplication context)
  {
   application = context;
   manager = DbManager.Get();
   application.BeginRequest += new EventHandler( OnBeginRequest );
  }
  protected void OnBeginRequest( object sender, EventArgs e )
  {
    manager.Init();
  }
  public void Dispose()
  {
    application.BeginRequest -= new EventHandler( OnBeginRequest );
  }
}
```

В последних примерах можно заметить ранее не упоминавшийся нами метод Init. Он служит для привязки данного экземпляра диспетчера к вызывающему потоку.

12. Корректное закрытие подключений

Общеизвестно, что при использовании пула подключений (connection pool) основополагающий принцип работы с подключениями гласит: открывай поздно, закрывай рано. Иными словами, открывать нужно перед самым использованием, а закрывать сразу после оного. При этом нужно помнить, что в блоке использования подключения к базе может произойти какая-нибудь ошибка (exception), которая может помешать нам закрыть подключение.

Мы уже рассматривали обычный для ADO.NET способ работы с объектом подключения. Примерно такой:

```
using( connection ) {
connection.Open();
// Активнее используем подключение! мначе, зачем открывали?!
...
}
```

Он весьма прост и удачен, если нам не жалко уничтожить объект подключения в конце блока, т.е. если мы каждый раз создаем новый объект подключения. Если же мы хотим использовать один объект, то код становится гораздо менее красивым:

```
   bool wasOpened = false;
   if( connection.State == ConnectionState.Closed )
   {
connection.Open();
wasOpened = true;
   }
   try {
// используем подключение
...
   }
   finally
   {
if( wasOpened ) connection.Close();
   }
```

Помимо общей сложности он еще и затрудняет возможность переключения между двумя режимами работы диспетчера подключений. Мы же не хотим, перекинув флаг, еще и править все блоки работы с базой.

Используем для решения тот же механизм детерминированной деструкции - интерфейс IDisposable, который столь упрощает использование подключения по стандартной схеме. Нам достаточно создать класс, который при конструировании открывал бы подключение, если это необходимо, а при уничтожении - закрывал бы, если открывал его сам. При этом мы можем сделать класс достаточно умным, чтобы он понимал, в каком режиме работает диспетчер подключений. Если диспетчер не кэширует объекты, наш сервисный класс будет удалять их в конце блока using.

В использовании это будет выглядеть примерно так:

```
   using( new DbOpen( connection )) {
// мспользуем подключение, раз открывали!
...
   }
```

Отметим также, что при конструировании экземпляра класса подключение открывается автоматически. Таким образом, еще одна строка в стандартном сценарии становится ненужной.

Литература

[1] Майоров А.В. Организация объектно-ориентированного доступа к реляционным базам данных // Моделирование и анализ информационных систем. 1999. Т.6, 2. С.28-33

[2] WebSphere Connection Manager // http://www.ibm.com /software /webservers /appserv /doc /v20dcadv /doc/whatis/iccmgr.html

[3] Netscape Application Server Architecture, Database Access Services // http://developer.netscape.com /docs /manuals /appserv /2_1/inover4.htm

[4] Gamma E., et al. Design Pattern: Elements of Reusable Object-Oriented Software. N.-Y. : Addison-Wesley. 1995.

[5] G. Andrew Duthie, Caching Improvements in ASP.NET Whidbey // http://msdn.microsoft.com /library /en-us /dnaspp /html /aspnetwhidbey_caching.asp¿

[6] Configuring applications, Data source configuration // http://jakarta.apache.org /struts /userGuide /configuration.html