Российская Академия Наук

Автоматика и телемеханика

Журнал основан в 1936 году
Выходит 12 раз в год

6
июнь
Москва 2022

Учредители журнала:
Отделение энергетики, машиностроения, механики и процессов управления РАН,
Институт проблем управления им. В.А. Трапезникова РАН (ИПУ РАН),
Институт проблем передачи информации им. А.А. Харкевича РАН (ИППИ РАН)

Адрес редакции: 117997, Москва, Профсоюзная ул., 65
Тел./факс: (495) 334-87-70
Электронная почта: redacsia@ipu.ru

Зав. редакцией *Е.А. Мартехина*

Москва
ООО «Тематическая редакция»

Тематический выпуск

ПРЕДИСЛОВИЕ

DOI: 10.31857/S0005231022060010, **EDN:** ACEKQG

В настоящем тематическом выпуске представлены статьи по различным вопросам искусственного интеллекта. Все представленные рукописи прошли не менее двух рецензий. Принятые к публикации статьи были существенно переработаны с учетом замечаний рецензентов.

Ведущим трендом исследований по искусственному интеллекту в последнем десятилетии является машинное обучение. Выпуск открывается статьей Д.В. Виноградова, представляющей собой обзор исследований по алгебраическому машинному обучению. Основной акцент сделан на вопросах вычислительной сложности, а также на использовании теории решеток и вероятностных алгоритмов, основанных на цепях Маркова. Еще три статьи посвящены применению методов машинного обучения в различных областях. В статье С.А. Шумского рассматривается подход к созданию «сильного» искусственного интеллекта. В ней описана иерархическая архитектура модели искусственной психики, использующая глубокое обучение с подкреплением. В статье А.О. Исхаковой, Д.А. Вольфа, Р.В. Мещерякова рассмотрено применение методов обучения на основе сверточных нейронных сетей для оценки эмоционального состояния человека по его речи. Предложен подход к предобработке данных, повышающий эффективность обучения. В статье А.И. Панова ставится и исследуется задача совместного планирования поведения и обучения когнитивного агента принятию решений в динамической среде. Предложен новый алгоритм обучения действиям в частично наблюдаемой внешней среде. Описан модельный пример и продемонстрирован принцип работы агента при управлении беспилотным автомобилем.

Две статьи посвящены проблемам анализа данных. В статье Т.В. Афанасьевой предложен подход к грануляции многомерных временных рядов, который применен для дескриптивного анализа социально-экономических показателей регионов Российской Федерации. В статье Д.А. Егурнова и Д.И. Игнатова задача трикластеризации трехмерных данных рассмотрена в терминах анализа формальных понятий. Предложены два метода решения этой задачи и проведены численные эксперименты, подтверждающие преимущества предложенных методов.

В статье К.С. Яковлева, А.А. Андрейчука, Ю.С. Белинской, Д.А. Макарова рассмотрена задача управления движением колесного робота в среде со статическими и динамическими препятствиями. Разработан метод быстрого планирования траектории движения робота, избегающей столкновения

с препятствиями. Проведены экспериментальные исследования, показавшие эффективность предложенного метода.

В статье О.П. Кузнецова исследован эффект затухания сигнала в цепи нейроподобных пороговых элементов при определенных соотношениях переходных процессов включения и отключения элементов. Найдены условия прохождения сигнала по такой цепи.

Статья И.С. Проскуркина и В.К. Ванага продолжает исследования по построению сетей из импульсно связанных спайковых микроосцилляторов, реализованных на основе колебательной химической реакции Белоусова–Жаботинского. Построена и экспериментально исследована сеть из микроосцилляторов, способная случайным образом реагировать на внешний сигнал.

В статье А.Я. Фридмана описан опыт применения методов искусственного интеллекта при создании прикладных систем ситуационного моделирования и управления.

Как видно, тематика статей номера довольно разнообразна как по методам исследования, так и по областям приложений. Она демонстрирует, что искусственный интеллект не сводится лишь к машинному обучению.

О.П. Кузнецов, д-р техн.наук

© 2022 г. Д.В. ВИНОГРАДОВ, д-р физ.-мат. наук
(vinogradov.d.w@gmail.com)
(Вычислительный центр им. А.А. Дородницына
Федерального исследовательского центра
“Информатика и управление” РАН, Москва)

АЛГЕБРАИЧЕСКОЕ МАШИННОЕ ОБУЧЕНИЕ: УПОР НА ЭФФЕКТИВНОСТЬ[1]

Представлен обзор современного состояния исследований по алгебраическому машинному обучению. Основной упор сделан на вопросы вычислительной сложности. Ключевыми идеями являются использование методов теории решеток и вероятностных алгоритмов, основанных на цепях Маркова.

Ключевые слова: вероятностные алгоритмы, сложность вычислений, решетки, машинное обучение, переобучение.

DOI: 10.31857/S0005231022060022, **EDN:** ACGDEV

1. Введение

В последнее время на волне успехов обучения нейронных сетей разных видов (рекуррентных, сверточных, LSTM и др.) наблюдается тенденция отказа классическим (“old-school symbolical”) методам извлечения знаний (например, ДСМ-методу [1] порождения гипотез) в праве считаться методами искусственного интеллекта. В этой ситуации отчасти несут ответственность создатели таких методов, не уделяющие достаточного внимания вопросам сложности вычислений.

Цель настоящей статьи — попытаться исправить это ложное мнение, объяснив, как использование вероятностных алгоритмов может победить “проклятие размерности”. Сосредоточимся на варианте алгебраического машинного обучения, основанного на бинарной операции сходства. Описанный подход назовем “алгебраическим”, потому что он пытается приближенно восстановить зависимость как набор элементов конечно-порожденной алгебры — решетки кандидатов (в гипотезы о причинах). Будем использовать идеи и результаты из Анализа формальных понятий (Formal Concept Analysis, FCA) [2] — современного раздела теории решеток [3].

Для понимания статьи требуется минимальное знакомство с теорией вероятностей (в объеме базового курса) и цепей Маркова (теорема о невозвратных состояниях). План работы следующий.

[1] Работа выполнена при частичной финансовой поддержке Российского фонда фундаментальных исследований (проект № 18-29-03063мк).

Раздел 2 будет посвящен парадигме Л. Вэльянта [4] — В.Н. Вапника–А.Я. Червоненкиса [5] “вероятно приближенно-корректного” (ВПК) обучения. Мы продемонстрируем этот подход в случае конъюнктивных понятий.

В разделе 3 будут обсуждаться проблемы классического подхода к извлечению знаний с помощью бинарной операции сходства (ДСМ-метода). Для этого предварительно напомним некоторые понятия FCA [2]. Затем обсудим случай булевой алгебры, который ставит барьер для прямого подхода к поиску гипотез с точки зрения теории сложности вычислений. Наконец, будет представлен результат автора о переобучении — возникновении “фантомных” кандидатов, который обосновывает вероятность допущения ошибок при предсказании.

Раздел 4 содержит описание вероятностных алгоритмов поиска сходств и исследование их алгоритмических свойств.

В разделе 5 будут описаны процедуры извлечения знаний с помощью вероятностного подхода. Будет выведена улучшенная оценка на число гипотез (кандидатов, не имеющих контрпримеров), чтобы с заданной надежностью правильно предсказать все достаточно важные тестовые примеры, предъявленные на прогноз наличия целевого свойства для оценки качества порожденных гипотез.

Наконец, в разделе 6 позволим себе сформулировать критерий дискриминации данных от знаний с точки зрения специалиста по теории сложности вычислений. Приводятся примеры задач Эйлера о мостах и коммивояжера, в котором такое различие видится наиболее ярко.

2. Вероятно приближенно-корректное обучение

Для пропедевтических целей начнем с антропоморфного примера вероятно приближенно-корректного обучения по Вэльянту–Вапнику–Червоненкису.

В некоем племени начинают готовить преемника стареющему вождю. Тестируют способность обучаться новому. Для этого кандидат в вожди должен отправиться с шаманом в джунгли с целью обучиться правилу, какие растения лекарственные, а какие нет. Перед выходом ученик должен сказать, сколько растений m он хочет изучить. Если это число окажется неразумно большим, то его дисквалифицируют.

Для очередного растения $\vec{x}_t \in X_n$ шаман говорит, лекарственное ли это растение (положителен ли $c(\vec{x}_t) = 1$ этот пример) или нет (контрпример для искомого правила — отрицательный пример $c(\vec{x}_t) = 0$). Все примеры $\vec{x}_t \in X_n$ появляются независимо и с одинаковой вероятностью (Это допущение не слишком реалистично, но именно оно позволяет работать усиленному закону больших чисел.). Когда они увидят объявленное учеником число m обучающих примеров $S = \{\langle \vec{x}_1, c(\vec{x}_1)\rangle, \ldots, \langle \vec{x}_m, c(\vec{x}_m)\rangle\}$, шаман с учеником возвращаются домой.

Затем ученик должен найти правило $h = h_S : X_n \to \{0, 1\}$, как распозна-

вать лекарственные растения. Эффективность этого шага иногда игнорируется, но это очень важно!

Наконец, ученик с шаманом отправляются в (те же самые) джунгли. Ученик обязан классифицировать первое встретившееся растение (тестовый пример) $\vec{x} \in X_n$, который появляется независимо от обучающей выборки и с распределением, совпадающим с обучающим. Если классификация ученика не совпадет с классификацией шамана (= неверна) $h_S(\vec{x}) \neq c(\vec{x})$, то ученика дисквалифицируют.

Задачами ученика являются:

1) выбрать язык описания объектов (задать область X_n обучающих примеров);

2) научиться оценивать снизу число m случайных обучающих примеров $S = \{\langle \vec{x}_1, c(\vec{x}_1)\rangle, \ldots, \langle \vec{x}_m, c(\vec{x}_m)\rangle\}$, чтобы из них можно с заранее выбранной вероятностью $\geqslant 1 - \delta$ получать правило $h : X_n \to \{0,1\}$, ошибающееся в не более чем заданной доле $\varepsilon > 0$ случаев (для любого $c : X_n \to \{0,1\}$);

3) придумать алгоритм $A(S) = h_S$ нахождения такого правила за ограниченное (полиномом от n, $\frac{1}{\varepsilon}$ и $-\log \delta = \log\left(\frac{1}{\delta}\right)$) число шагов;

4) реализовать предсказание тестовых примеров $\vec{x} \in X_n$ с помощью найденного правила h_S.

Продемонстрируем успех парадигмы ВПК-обучения на фольклорном примере обучения конъюнктивным понятиям.

Определение 1. Пусть каждый объект x описывается множеством бинарных признаков $f_1, \ldots, f_n$, т.е. множество объектов $X \subseteq \{0,1\}^n$, где n – число признаков. Чтобы узнать значение признака на объекте $\vec{x}$, введем булевские переменные $p_1, \ldots, p_n : \{0,1\}^n \to \{0,1\}$, соответствующие проецированию на компоненты. Переменные вместе с их отрицаниями назовем ***литералами****.*

Понятие *— подмножество объектов $\{\vec{x} \in X : c(\vec{x}) = 1\}$, задаваемое булевой функцией $c : X \to \{0,1\}$, называемой* ***индикатором****. Если $c(\vec{x}) = 1$, то объект $\vec{x} \in X$ называется положительным примером понятия, в противном случае — отрицательным примером. Если индикатор представим конъюнкцией литералов, то такое понятие назовем* ***конъюнктивным****.*

Гипотеза *— список литералов $h = \{l_{i_1}, \ldots, l_{i_k}\}$, конъюнкция которых представляет кандидата на обучаемое понятие. Предсказание с помощью гипотезы осуществляется по булевой функции $h = l_{i_1} \wedge \ldots \wedge l_{i_k}$ в качестве индикатора.*

Легко проверить, что, отождествляя объект $\vec{x} \in \{0,1\}^n$ с максимально непротиворечивым множеством литералов $\{l_1, \ldots, l_n\}$ ($l_j \in \{\neg f_j, f_j\}$), гипотеза h доопределяет пример $\vec{x}$ положительно, если и только если $h \subseteq \vec{x}$.

В задаче ВПК-обучения конъюнктивным понятиям предполагается, что искомое понятие c задается конъюнкцией $c = \{l'_{i_1}, \ldots, l'_{i_r}\}$ литералов.

Определение 2. Объем m выборки $S = \{\langle \vec{x}_1, c(\vec{x}_1)\rangle, \ldots, \langle \vec{x}_m, c(\vec{x}_m)\rangle\}$ назовем ***большим****, если $m \geqslant \frac{2n \cdot [\ln(2n) - \ln(\delta)]}{\varepsilon}$.*

Алгоритм 1 (пересечения).

1. Перечислим элементы $S = \{\langle \vec{x}_1, c(\vec{x}_1)\rangle, \ldots, \langle \vec{x}_m, c(\vec{x}_m)\rangle\}$ в некотором порядке.

2. Положим $h_0 = L = \{\neg p_1, p_1, \ldots, \neg p_n, p_n\}$ (множество всех литералов).

3. По порядку проверяем $h_j(\vec{x}_{j+1}) \neq c(\vec{x}_{j+1})$. Если ошибка не происходит, то гипотеза не изменяется. Когда случается ошибка, из гипотезы удаляются те литералы, которых нет в неправильно предсказанном примере:

$$h_j(\vec{x}_{j+1}) \neq c(\vec{x}_{j+1}) \Rightarrow h_{j+1} = h_j \setminus (L \setminus \vec{x}_{j+1}).$$

Легко проверить, что $h_j \setminus (L \setminus \vec{x}_{j+1}) = h_j \cap \vec{x}_{j+1}$ как множества литералов, что и дает название алгоритму 1.

Сформулируем несколько лемм, чтобы доказать, что выборка большого объема достаточна, чтобы с заранее выбранной вероятностью $\geqslant 1 - \delta$ алгоритм 1 находил правило $h : X_n \to \{0, 1\}$, ошибающееся в не более чем заданной доле $\varepsilon > 0$ случаев.

Лемма 1. Никакой литерал из обучаемого конъюнкта $c = \{l'_{i_1}, \ldots, l'_{i_k}\}$ не удаляется алгоритмом пересечения, т.е. $c \subseteq h_j$ для всех j.

Лемма 2. Алгоритм 1 может ошибаться только на положительных примерах, т.е. случай $h_j(\vec{x}_j) = 1 \neq 0 = c(\vec{x}_j)$ не возможен.

Определение 3. Литерал l назовем ***плохим****, если $\mathbb{P}[c(\vec{x}) = 1 \& l \notin \vec{x}] > > \frac{\varepsilon}{2n}$.*

Лемма 3. Если h не содержит плохих литералов, то $\mathbb{P}[h(\vec{x}) \neq c(\vec{x})] \leqslant \varepsilon$.

Лемма 4. Если $m \geqslant \frac{2n \cdot [\ln(2n) - \ln(\delta)]}{\varepsilon}$, то $2n \cdot \left(1 - \frac{\varepsilon}{2n}\right)^m \leqslant \delta$.

Теорема 1. Алгоритм пересечения доказывает эффективную ВПК-обучаемость конъюнктивным понятиям.

Доказательство. Докажем, что при большом объеме выборки m вероятность того, что в гипотезе h останется хоть один плохой литерал, не превзойдет δ.

Пусть l – плохой литерал. Тогда $\mathbb{P}[l$ сохранится после одного примера$] = = 1 - \mathbb{P}[l$ удалится первым примером $] \leqslant 1 - \frac{\varepsilon}{2n}$.

Поэтому $\mathbb{P}[l$ сохранится после m примеров$] \leqslant \left(1 - \frac{\varepsilon}{2n}\right)^m$.

Так как всего литералов ровно $2n$, то утверждение теоремы следует из неравенства Буля $\mathbb{P}\left[\cup_{j=1}^k A_j\right] \leqslant \sum\limits_{j=1}^k \mathbb{P}[A_j]$ и леммы 4.

Ясно, что $\frac{2n \cdot [\ln(2n) - \ln(\delta)]}{\varepsilon}$ мажорируется полиномом от n, $\frac{1}{\varepsilon}$ и $-\log \delta$, и число шагов алгоритма пересечения для обработки одного примера линейно зависит от n.

3. Проблемы извлечения знаний с помощью операции сходства

Если изучаемое понятие $c : \{0,1\}^n \to \{0,1\}$ не является конъюнктивным, то алгоритм 1 не работает, так как выдает единственный (чаще всего пустой) конъюнкт. Более широким классом булевых функций, для которого возможно эффективное ВПК-обучение, являются списки решений [6]. Для общего случая дизъюнктивных нормальных формул вопрос остается открытым. Если ограничиться конъюнкцией хорновских дизъюнктов, то в [7, 8] предложены алгоритмы ВПК-обучения с использованием эмуляции запросов о принадлежности и эквивалентности с помощью случайных оценок.

Хотя индуктивные методы извлечения знаний восходят к трудам английского логика XIX в. Д.С. Милля [9], интерес к ним возродился с возникновением искусственного интеллекта на Западе в 1950-е гг. Некоторые эксперименты с индуктивным образованием понятий в логике высказываний описаны в книге [10], изданной в 1966 г. Важным был доклад С.А. Вере [11] на Четвертой Объединенной международной конференции по ИИ, проходившей в Тбилиси в 1975 г., где операция сходства в логике предикатов была реализована через унификацию.

В пионерских работах В.К. Финна [1, 12] был предложен метод извлечения знаний из обучающих примеров, описываемых бинарными признаками, с учетом контрпримеров, описываемых таким же образом. При этом допускалась множественность целевых свойств (тоже как набор бинарных переменных). В.К. Финн рассматривал симметричную ситуацию: нахождение сходств как положительных примеров, не допускающих включение ни в какой отрицательный контрпример, так и отрицательных примеров с запретом положительных контрпримеров. Это привело к красивой теории логики аргументации [13, 14], но дополнительно увеличило вычислительную сложность. Но более существенным недостатком первоначального подхода было ограничение булевой алгеброй как носителя универсума примеров и их сходств. Это привело к проблемам формализации методов различия [15] и остатков [16] Д.С. Милля, так как они использовали операцию относительного дополнения. Для булевых алгебр она определяется однозначно, что не так для недистрибутивных решеток.

Ранее в теории решеток [3] возникло новое направление — анализ формальных понятий (FCA) [2]. Уже в 1982 г. Рудольф Вилле доказал свою Фундаментальную теорему, которая позволила эффективно работать с общими конечными решетками.

Следующие шаги в развитии отечественного подхода к извлечению знаний с помощью бинарной операции сходства были сделаны в трудах С.О. Кузнецова. Сначала им был предложен алгоритм [17] “Замыкай-по-одному” для эффективного построения остовного дерева в решетке всех кандидатов.

Затем С.О. Кузнецов [18, 19] показал, как для извлечения знаний из обучающих примеров, описываемых бинарными признаками, использовать тех-

Таблица 1. Минимальная выборка для булевой алгебры

I	f_1	f_2	$\ldots$	f_n
o_1	0	1	$\ldots$	1
o_2	1	0	$\ldots$	1
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
o_n	1	1	$\ldots$	0

нику FCA, расширяя теорию с булевых алгебр обучающих примеров на произвольные конечные (полу-)решетки.

Более того, Фундаментальная теорема Р. Вилле ответила на вопрос о достаточности описания примеров бинарными признаками: любая конечная решетка может быть с точностью до изоморфизма порождена как решетка кандидатов некоторой обучающей выборки из примеров, описываемых битовыми строками.

С помощью техники FCA Д.В. Виноградову [20] удалось создать алгоритм и доказать его корректность для представления объектов, описываемых признаками, чьи значения образуют произвольную конечную нижнюю полурешетку, с помощью битовых строк.

Собирая вместе битовые строки положительных (обучающих) примеров O, получаем "обучающую выборку" (или, в терминологии FCA, формальный контекст) $I \subseteq O \times F$, где $F = \{f_1, \ldots, f_n\}$ – множество бинарных признаков.

Определение 4. Для подмножества $A \subseteq O$ объектов его общим ***фрагментом*** *называется подмножество признаков $A' = \{f \in F : \forall o \in A [oIf]\} \subseteq F$. Ясно, что $\emptyset' = F$. A' вычисляется побитовым умножением строк, соответствующих отобранным во множество A объектам.*

Для подмножества $B \subseteq F$ признаков его ***родителями*** *называется подмножество объектов $B' = \{o \in O : \forall f \in B\, [oIf]\} \subseteq O$. Очевидно, что $\emptyset' = O$. B' можно вычислить побитовым умножением столбцов, соответствующих отобранным во множество B признакам.*

Пару $\langle A, B \rangle$ назовем ***кандидатом****, если $A = B' \subseteq O$ и $B = A' \subseteq F$.*

В FCA [2] кандидаты называются "формальными понятиями", но предпочтем сменить имя, так как "кандидаты" из нашего определения играют эту роль в ДСМ-методе [1], где они превращаются в "гипотезы о причинах" после фальсификации некоторых из них дополнительными проверками.

Случаю булевой алгебры соответствует выборка (см. таблицу 1).

Здесь множество обучающих примеров $O = \{o_1, o_2, \ldots, o_n\}$ – коатомы, где $o_i I f_j$ (т.е. пример o_i имеет признак f_j), если и только если $i \neq j$.

Ясно, что любая пара $\langle O \setminus \{o_{j_1}, \ldots, o_{j_k}\}, \{f_{j_1}, \ldots, f_{j_k}\} \rangle$ будет кандидатом, поэтому имеем булеву алгебру всех 2^n подмножеств признаков (= битовых строк).

При $n = 32$ обучающая выборка занимает $n \cdot n = 2^{10}$ бит = 128 байт. Но чтобы записать результат, требуется $n \cdot 2^n = 2^{37}$ бит, т.е. ровно 16 Гбайт!

Этот пример указывает на вычислительную трудность реализации первоначальной идеи В.К. Финна о порождении всех кандидатов, а затем фальсификации тех из них, которые вкладываются (как битовые строки) в отрицательные примеры (так называемый "запрет контрпримеров").

Но проблема не ограничивается сложностью вычислений по памяти. Д.В. Виноградов [21] обнаружил эффект "переобучения" — возникновение "фантомных" кандидатов, не устранимых сколь угодно большим количеством контрпримеров.

Мы предполагаем, что имеется по крайней мере 2 различных набора бинарных признаков (= 2 причины), включение любого из которых в описание объекта делает его положительным. Допустим, имеется n не входящих ни в одну из этих причин признаков, называемых **сопутствующими**.

Рассмотрим следующую вероятностную модель:

1) в обучающей выборке имеется четыре обучающих примера, два из которых в точности совпадают с различными причинами;

2) остальные два обучающих примера содержат по одной из причин, а сопутствующие бинарные признаки образуют последовательности Бернулли с вероятностью успеха p;

3) имеется m контрпримеров, каждый из которых не имеет ни одной из причин, а сопутствующие признаки образуют последовательности Бернулли с вероятностью успеха p, все случайные биты предполагаются независимыми.

Нетривиальное сходство второй пары обучающих примеров назовем **фантомным**. Заметим, что возникновение такого сходства мешает правильному предсказанию целевого свойства у тестовых примеров, так как его включение объявит этот пример истинным, хотя он может не содержать ни одной из исходных (правильных) причин. Это сходство возникает из-за попытки извлечь максимум информации из обучающей выборки, но объясняется случайным совпадением нескольких сопутствующих признаков у обучающих примеров второй пары, каждый из которых имеет свою причину, отличную от таковой у другого примера.

Довольно легко доказать лемму 5.

Лемма 5. С вероятностью $\sum\limits_{j=0}^{m} \binom{m}{j} \cdot (-1)^j \cdot (1 - p^2 + p^{2+j})^n$ *возникнет фантомное сходство, которое не устранится ни одним из* m *случайных контрпримеров.*

С помощью леммы 5 доказывается теорема 2.

Теорема 2 [21]. *При числе сопутствующих признаков* $n \to \infty$ *и вероятности появления этих признаков у контрпримеров и обучающих примеров, равной* $p = \sqrt{\frac{a}{n}}$ $(a \leqslant 1)$, *вероятность возникновения фантомного сходства двух обучающих примеров, не устраненного никаким из* $m = c \cdot \sqrt{n}$ *контрпримеров, будет стремиться к*

$$1 - e^{-a} - a \cdot e^{-a} \cdot \left[1 - e^{-c \cdot \sqrt{a}}\right] > 0.$$

Легко проверить, что вероятность $p \geqslant \sqrt{\frac{a}{n}}$ гарантирует, что возникнет нетривиальное фантомное сходство, т.е. фрагмент, который содержит хотя бы один сопутствующий признак; при $p < \sqrt{\frac{a}{n}}$ гарантии нет даже безотносительно наличия контрпримеров.

Левая часть последнего неравенства теоремы при $c \to \infty$ уменьшится до $1 - e^{-a} - a \cdot e^{-a}$, что совпадает с вероятностью того, что пуассоновская случайная величина со средним a примет значение больше единицы, что не может стать равным нулю. Это значит, что даже для сколь угодно большого числа контрпримеров вероятность возникновения фантомного сходства не может быть обнулена.

Другими словами, этот результат указывает на возможность некорректного предсказания тестовых примеров, т.е. приводит к аналогу ВПК-обучения. А в этом случае можно не ограничиваться детерминированными алгоритмами.

Л.А. Якимова [22] смогла экспериментально обнаружить фантомные гипотезы при работе алгоритма исчерпывающего порождения гипотез (ДСМ-метода) на массиве Mushrooms из репозитория данных для тестирования алгоритмов машинного обучения Университета Калифорнии в г. Ирвайн [23]. Особенно настораживающим был тот факт, что фантомные гипотезы смогли заставить ДСМ-систему неправильно объявить несколько ядовитых грибов съедобными.

4. Вероятностные алгоритмы поиска сходств

Отказавшись от детерминизма, переходим к идее порождения достаточно большого числа кандидатов (в гипотезы о причинах), каждая своей траекторией случайного блуждания по решетке кандидатов.

Базовые шаги этих блужданий и их корректность устанавливаются в следующей легко доказываемой лемме-определении.

Лемма 6. Операция ***замыкай-по-одному-вниз*** *на кандидате* $\langle A, B\rangle$ *и объекте* $o \in O$ *порождает кандидат*

$$CbODown(\langle A, B\rangle, o) = \big\langle (A \cup \{o\})'', B \cap \{o\}' \big\rangle.$$

Операция ***замыкай-по-одному-вверх*** *на кандидате* $\langle A, B\rangle$ *и признаке* $f \in F$ *порождает кандидат*

$$CbOUp(\langle A, B\rangle, f) = \big\langle A \cap \{f\}', (B \cup \{f\})'' \big\rangle.$$

Возможно небольшое ускорение вычислений, если заметить, что $o \in A \Rightarrow$ $\Rightarrow CbODown(\langle A, B\rangle, o) = \langle A, B\rangle$ и $f \in B \Rightarrow CbODown(\langle A, B\rangle, f) = \langle A, B\rangle$, так как проверка принадлежности значительно быстрее, чем вычисление операций *CbO*.

В случае булевой алгебры все еще сильнее ускоряется: если $o_j \notin A$, то $CbODown(\langle A, B\rangle, o_j) = \langle A \cup \{o_j\}, B \setminus \{f_j\}\rangle$, и $CbODown(\langle A, B\rangle, o_j) = \langle A, B\rangle$ в противном случае. Аналогично если $f_j \notin B$, то $CbOUp(\langle A, B\rangle, f_j) =$ $= \langle A \setminus \{o_j\}, B \cup \{f_j\}\rangle$, и, как обычно, $CbOUp(\langle A, B\rangle, f_j) = \langle A, B\rangle$ при $f_j \in B$.

Простейшее случайное блуждание порождается алгоритмом 2.

Алгоритм 2 (немонотонный).

1. Сформируем обучающую выборку $I \subseteq O \times F$, где $O := (+)$-примеры, $F :=$ признаки.

2. Начинаем с наименьшего кандидата: положим $A := O$; $B := O'$.

3. До некоторого шага T делаем цикл из пунктов 4 и 5.

4. Формируем множество свободных примеров и признаков: $R := (O \setminus A) \cup \cup (F \setminus B)$.

5. Выбираем случайный элемент $r \in R$.

Если выбран объект $r \in O \setminus A$, то применяем операцию CbODown: $\langle A, B\rangle := CbODown(\langle A, B\rangle, r)$.

Если выбран признак $r \in F \setminus B$, то применяем операцию CbOUp: $\langle A, B\rangle := := CbOUp(\langle A, B\rangle, r)$.

6. После цикла выдается результат $\langle A, B\rangle$ – кандидат, на котором алгоритм 2 остановился на шаге T.

Очевидно, что в случае булевой алгебры алгоритм 2 с равной $\frac{1}{n}$ вероятностью переходит с текущего подмножества на одного из n его соседей, т.е. представляет собой случайное блуждание по соответствующему гиперкубу.

Алгоритм 3 (монотонный).

1. Сформируем обучающую выборку $I \subseteq O \times F$, где $O := (+)$-примеры, $F :=$ признаки.

Формируем дизъюнктное объединение множеств примеров и признаков: $R := O \cup F$.

2. Начинаем с наименьшего кандидата: положим $A := O$; $B := O'$.

3. До некоторого шага T делаем цикл.

4. Выбираем случайный элемент $r \in R$.

Если выбран объект $r \in O$, то применяем операцию CbODown: $\langle A, B\rangle := := CbODown(\langle A, B\rangle, r)$.

Если выбран признак $r \in F$, то применяем операцию CbOUp: $\langle A, B\rangle := := CbOUp(\langle A, B\rangle, r)$.

5. После цикла выдается результат $\langle A, B\rangle$ – кандидат, на котором алгоритм 3 остановился на шаге T.

В случае булевой алгебры алгоритм 3 с вероятностью $\frac{1}{2}$ никуда не сдвигается, а с равной $\frac{1}{2 \cdot n}$ вероятностью переходит с текущего подмножества на одного из n его соседей, т.е. представляет собой ленивое случайное блуждание по соответствующему гиперкубу.

Заметим, что корректность алгоритмов 2 и 3 (т.е. то, что в результате их работы обязательно получим кандидата) следует из леммы 6.

Основная проблема с алгоритмами 2 и 3, применяемыми к произвольным обучающим выборкам, состоит в том, что неизвестна никакая полиномиальная оценка на "время остановки" T, которая обеспечивала бы хорошую "пе-

ремешиваемость" соответствующей цепи Маркова. Таким образом, вопрос о выборе параметра T (фактически, о длине цикла) в этих алгоритмах остается открытым.

По этой причине и после экспериментов с алгоритмами 2 и 3 [24] было решено отказаться от них в пользу семейства спаривающих цепей Маркова [25]. Простейший вариант задается алгоритмом 4.

Алгоритм 4 (спаривающий).

1. Сформируем обучающую выборку $I \subseteq O \times F$, где $O :=$ (+)-примеры, $F :=$ признаки.

2. Положим $R := O \cup F$; $\mathrm{Min} := \langle O, O' \rangle$ – наименьший кандидат; $\mathrm{Max} :=$ $:= \langle F', F \rangle$ – наибольший кандидат.

3. Пока $\mathrm{Min} \neq \mathrm{Max}$, т.е. оба кандидата не станут равны, делать цикл.

4. Выбираем случайный элемент $r \in R$.

Если выбран объект $r \in O$, то одновременно вычисляются

$$\mathrm{Min} := CbODown(\mathrm{Min}, r); \quad \mathrm{Max} := CbODown(\mathrm{Max}, r).$$

Если выбран признак $r \in F$, то одновременно вычисляются

$$\mathrm{Min} := CbOUp(\mathrm{Min}, r); \quad \mathrm{Max} := CbOUp(\mathrm{Max}, r).$$

5. После цикла выдается результат – любой (из совпадающих) кандидатов, например, Min.

Для доказательства останавливаемости алгоритма 4 с вероятностью единица, нам нужно определение 5.

Определение 5. ***Порядок*** *на кандидатах:* $\langle A_1, B_1 \rangle \leqslant \langle A_2, B_2 \rangle$, *если* $B_1 \subseteq B_2$.

Порядок, задаваемый этим определением, двойственен порядку, рассматриваемому в FCA [2]: $\langle A_1, B_1 \rangle \leqslant \langle A_2, B_2 \rangle$, если $A_1 \subseteq A_2$.

Теперь легко доказать лемму 7.

Лемма 7. Для всякой упорядоченной пары кандидатов $\langle A_1, B_1 \rangle \leqslant \langle A_2, B_2 \rangle$ *и любого* $o \in O$ *имеем* $CbODown(\langle A_1, B_1 \rangle, o) \leqslant CbODown(\langle A_2, B_2 \rangle, o)$.

Для всякой упорядоченной пары кандидатов $\langle A_1, B_1 \rangle \leqslant \langle A_2, B_2 \rangle$ *и любого* $f \in F$ *имеем* $CbOUp(\langle A_1, B_1 \rangle, f) \leqslant CbOUp(\langle A_2, B_2 \rangle, f)$.

С помощью леммы 7 легко доказать теорему 3.

Теорема 3 [25]. *Алгоритм 4 соответствует цепи Маркова.*

Эргодические состояния спаривающей цепи Маркова имеют вид $\langle A, B \rangle =$ $= \langle A, B \rangle$ (соответствуют совпадающим парам кандидатов). Невозвратные состояния имеют вид $\langle A_1, B_1 \rangle < \langle A_2, B_2 \rangle$ (склеивание кандидатов еще не произошло).

Теперь имеем результат о конечности траекторий с вероятностью единица как следствие классической теоремы о невозвратных состояниях [26].

Следствие 1. Вероятность того, что состояние $\langle A_1(t), B_1(t) \rangle \leqslant$ $\leqslant \langle A_2(t), B_2(t) \rangle$ *спаривающей цепи Маркова окажется невозвратным, стре-*

мится к нулю, когда $t \to \infty$. Следовательно, алгоритм 4 остановится с вероятностью единица.

Хотя вопрос о среднем времени работы алгоритма 4 остается открытым, Д.В. Виноградовым получены теорема о среднем времени остановки этого алгоритма и о сильной концентрации около этого среднего для частного случая булевой алгебры [27]. Здесь на вход алгоритма 4 подается выборка, определенная на странице 5, но мы имеем вероятностный алгоритм, следующий шаг которого определяется датчиком случайных чисел (для выбора объекта или признака для замыкания).

Теорема 4. Среднее время работы алгоритма 4 для n-мерной булевой алгебры равно

$$\mathbb{E}T = \sum_{j=1}^{n} \frac{n}{j} \approx n \cdot \ln(n) + n \cdot \gamma + \frac{1}{2}.$$

$\mathbb{P}\left[T \geqslant (1+\varepsilon) \cdot n \cdot \ln(n)\right] \to 0$ *при* $n \to \infty$ *для любого* $\varepsilon > 0$.

Эта теорема позволяет понять, насколько эффективно работает алгоритм спаривающей цепи Маркова. В случае 32-мерной булевой алгебры количество ее элементов составляет 4 294 967 296. Одна траектория авторского алгоритма спаренного случайного блуждания имеет среднюю длину $\mathbb{E}T \leqslant 130$. Из-за сильной концентрации все траектории имеют примерно ту же самую длину. При каждом шаге порождается (не всегда, так как возможны возвращения или неподвижность одного или обоих кандидатов из пары) не более двух кандидатов (один снизу и один сверху). Таким образом, при порождении, например, 1000 случайных кандидатов с помощью 1000 траекторий суммарно будет порождено около 260 000 элементов булевой алгебры.

Эксперименты Л.А. Якимовой [22] показали, что этот эффект (быстрая остановка и, как следствие, порождение малого числа кандидатов) наблюдается и в случае общих обучающих выборок.

В качестве практического средства для устранения наиболее длинных траекторий возможно применение следующей техники остановки алгоритма 4 (или его ленивого варианта в алгоритме 5 далее) и запуска его заново.

Определение 6. Если $T_1, \ldots, T_r$ – независимые целочисленные случайные величины, имеющие распределение времени склеивания T, то ***верхняя граница склеивания*** *по r испытаниям определяется как $\hat{T} = T_1 + \ldots + T_r$.*

На практике предлагается сделать r прогонов спаривающей цепи Маркова и взять оценку $t_1 + \ldots + t_r$ верхней границы склеивания.

Оценим, как изменяются вероятности попадания в эргодические состояния при остановке спаривающей цепи Маркова по r прогонам.

Определение 7. Для целочисленной случайной величины $\hat{T}$, независимой от целочисленной случайной величины T, ***условное распределение*** *со-*

стояний относительно события $B = \{T \leqslant \hat{T}\}$ *есть распределение*

$$\nu_i = \mu_{\hat{T},i} = \frac{\mathbb{P}[X_T = i, T \leqslant \hat{T}]}{\mathbb{P}[T \leqslant \hat{T}]}$$

для любого эргодического состояния i.

Используем расстояние тотального изменения (вариации) в качестве метрики между распределениями вероятности на конечном множестве.

Определение 8. Расстояние ***тотального изменения*** *между распределениями вероятностей* $\mu = (\mu_i)_{i \in U}$ *и* $\nu = (\nu_i)_{i \in U}$ *на конечном пространстве* U *определяется формулой:* $\|\mu - \nu\|_{TV} = \frac{1}{2} \cdot \sum\limits_{i \in U} |\mu_i - \nu_i|$.

Приведем ключевые леммы, необходимые для доказательства ключевой теоремы.

Лемма 8. $\|\mu - \nu\|_{TV} = \max_{R \subseteq U} |\mu(R) - \nu(R)|$.

Здесь подмножество R, на котором достигается максимум, можно задать, например, формулой: $R = \{i \in U \mid \mu_i > \nu_i\}$.

Лемма 9. $\|\mu - \mu_{\hat{T}}\|_{TV} \leqslant \frac{\mathbb{P}[T > \hat{T}]}{1 - \mathbb{P}[T > \hat{T}]}$ *, где* $\mu_{\hat{T}}$ *– распределение остановленной на верхней границе* $\hat{T}$ *склеивания по* $r > 1$ *испытаниям, а* μ *– распределение выдачи неостановленной цепи.*

Лемма 10. $\|\mu - \mu_{\hat{T}}\|_{TV} \leqslant \frac{1}{2^r - 1}$ *, где* $\mu_{\hat{T}}$ *– распределение остановленной на верхней границе склеивания по* $r > 1$ *испытаниям, а* μ *– распределение выдачи неостановленной цепи.*

Соединяя результаты лемм 8 и 10, получим теорему 5.

Теорема 5 [25]. *Для любого подмножества* R *решетки кандидатов если вероятность алгоритма 4 выбора какого-то элемента* R *равна* $\mu(R) = \sum\limits_{j \in R} \mathbb{P}[X_T = i]$, *то вероятность* $\mu_{\hat{T}}(R)$ *выбора какого-то элемента этого подмножества остановленным вариантом алгоритма 4 ограничена снизу* $\mu_{\hat{T}}(R) \geqslant \mu(R) - \frac{1}{2^r - 1}$, *если остановка совершается по верхней границе* $\hat{T}$ *склеивания для* $r > 1$ *предварительных траекторий.*

Впрочем, этот результат может не иметь практического смысла. Как показали эксперименты Л.А. Якимовой [22], обычно длинные траектории так редки (хотя и наблюдались), что проверка критерия остановки в цикле приводит к замедлению вычислений, не компенсируемым отбрасыванием длинных траекторий. Этот эмпирический факт находится в хорошем согласии со свойствами малой средней длины и сильной концентрации длин траекторий около среднего, которые для случая булевой алгебры были доказаны в теореме 4 выше.

Но бесспорным стало преимущество использования ленивого варианта спаривающей цепи Маркова. Чтобы понять, откуда здесь возникают возможности ускорения вычислений, необходимо подробнее рассмотреть базовые шаги случайных блужданий и их реализацию на современных компьютерах.

Фундаментальная теорема Р. Вилле оказывается полезной не только с идеологической (достаточно ограничиться описанием обучающих объектов и их сходств битовыми строками, т.е. наша постановка задачи максимально общая), но и с вычислительной точки зрения. Дело в том, что сходство двух битовых строк реализуется побитовым умножением, а эта операция на современных компьютерах реализована максимально эффективно. На CPU она занимает один такт (если битовая строка влезает в регистр), а их около 5 млрд. в секунду. На GPGPU нужны четыре волны (четыре такта с частотой более 1,5 ГГц), но зато современные видеокарты могут в параллель обрабатывать до (а сейчас уже и более) 4096 потоков. При этом алгоритмы, основанные на цепях Маркова, допускают максимальное распараллеливание: траектория определяется только датчиками случайных чисел.

Более того, современные языки программирования имеют специальные структуры данных (например, C++ использует boost::dynamic_bitset для работы с битовыми строками, причем даже переменной длины). Современные компиляторы умеют дополнительно оптимизировать побитовое умножение коротких строк, размещая несколько штук в длинные регистры и выполняя за один такт сразу несколько умножений (так называемый векторный параллелизм).

В базовом шаге $CbODown(\langle A, B\rangle, o) = \langle (A \cup \{o\})'', B \cap \{o\}'\rangle$ вторая компонента (вычисление фрагмента) соответствует побитовому умножению фрагмента старого кандидата B и описания $\{o\}'$ дополнительного объекта-родителя, т.е. вычисляется очень быстро. Первая же компонента $(A \cup \{o\})''$, так называемое "замыкание", требует нахождения всех родителей у соответствующего фрагмента. Хорошо еще, что $(A \cup \{o\})'' = (B \cap \{o\}')'$, т.е. можно использовать побитовое умножение столбцов из матрицы обучающей выборки. Но даже в этом случае количество перемножаемых столбцов может быть велико.

Аналогичное замечание верно и для $CbOUp$, только там первая и вторая компоненты меняются местами относительно сложности их вычисления.

Ключевым наблюдением будет то, что объекты и признаки выпадают сериями, а внутри серии нужно вычислять только быстрые операции. Когда серия прерывается, нужно один раз сделать вычислительно сложную операцию замыкания (для $CbOUp$ используется равенство $(B \cup \{f\})'' = (A \cap \{f\}')'$).

Собирая все нужные нам идеи, получаем алгоритм 5.

Алгоритм 5 (ленивый).

1. Сформируем обучающую выборку $I \subseteq O \times F$, где $O :=$ (+)-примеры, $F :=$ признаки.

2. Положим $R := O \cup F$; $A_1 := O$, $B_1 := O'$ – наименьший кандидат; $A_2 := F'$, $B_2 := F$ – наибольший кандидат.

Выбираем направление движения $moveUp := true$.

3. Пока $\mathrm{Min} \neq \mathrm{Max}$, т.е. оба кандидата не станут равны, делать цикл.

4. Выбираем случайный элемент $r \in R$.

5. Если выбран объект $r \in O$, то проверяем истинность $moveUp$.

Если это так, то сначала применяем замыкание $B_1 := A'_1$; $B_2 := A'_2$, и переопределяем $moveUp := false$.

Затем (независимо от $moveUp$) одновременно вычисляются сходства $B_1 := := B_1 \cap \{r\}'$; $B_1 := B_1 \cap \{r\}'$.

6. Если выбран признак $r \in F$, то проверяем истинность $moveUp$.

Если это не так, то сначала применяем замыкание $A_1 := B'_1$; $A_2 := B'_2$, и переопределяем $moveUp := true$;

Затем (независимо от $moveUp$) одновременно вычисляются сходства $A_1 := := A_1 \cap \{r\}'$; $A_1 := A_1 \cap \{r\}'$;

7. После цикла выдается результат — любой (из совпадающих) кандидатов, например, Min.

Возникает вопрос о степени экономии, достигаемой такой процедурой. Рассмотрим последовательность типов (объект или признак) элементов, выбираемых в ходе работы алгоритма 5. Ясно, что это – последовательность (вообще говоря, бесконечная) испытаний Бернулли $\langle \sigma_1, \ldots, \sigma_j, \ldots \rangle$ с вероятностью успеха (например, выбора признака), равной $p = \frac{n}{n+k}$, где n – число признаков, а k – число обучающих примеров.

Прежде всего зафиксируем два события $\{\sigma_1 = 0\}$ и $\{\sigma_1 = 1\}$.

В случае $\sigma_1 = 0$ нас интересует длина события $\{\sigma_1 = \ldots = \sigma_i = 0, \sigma_{i+1} = = \ldots = \sigma_j = 1, \sigma_{j+1} = 0\}$, а в случае $\sigma_1 = 1$ интересна длина $\{\sigma_1 = \ldots = \sigma_i = = 1, \sigma_{i+1} = \ldots = \sigma_j = 0, \sigma_{j+1} = 1\}$.

Рассмотрим случайную величину T суммы длин двух переходов от объектов к признакам и снова к объектам (при $\sigma = 0$) и суммы длин двух переходов от признаков к объектам (при $\sigma = 1$).

Теорема 6 [25]. *В ленивой схеме вычислений на каждую пару применений операции замыкания, одной для CbOUp и одной для CbODown, в среднем в классической схеме мы будем делать* $\mathbb{E}T = \frac{(n+k)^2}{k \cdot n}$ *операций замыкания.*

Ясно, что выигрыш тем больше, чем больше разница между k и n, где k – число обучающих примеров, а n – число признаков, используемых для описания объектов, так как $\frac{(n+k)^2}{k \cdot n} = 4 + \frac{(n-k)^2}{k \cdot n}$. Даже в худшем случае $k = n$ это сокращение вызовов трудоемкой операции не меньше двух раз, потому что $\frac{(2k)^2}{k \cdot k} = 4$.

Л.А. Якимова в ее выпускной квалификационной работе бакалавра [28], выполненной под руководством Д.В. Виноградова, продемонстрировала, что выигрыш от такого перехода на реальных данных может достигать очень большой величины, близкой к теоретическому предсказанию.

Сравнение алгоритмов 4 и 5 проводилось на массиве Mushrooms из репозитория данных для тестирования алгоритмов машинного обучения Университета Калифорнии в г. Ирвайн [23]. В этом массиве содержится описание 8124 грибов, из которых $k = 4208$ являются съедобными, а 3916 ядовитыми. Грибы кодировались битовыми строками длины $n = 124$ бита.

По теореме 6 средний выигрыш от применения ленивой схемы вычислений достигает на операциях замыкания $\frac{1}{2} \cdot \frac{(k+n)^2}{k \cdot n} \approx 18$ раз.

Алгоритмы спаривающих цепей Маркова в стандартном (алгоритм 4) и ленивом вариантах (алгоритм 5) сравнивались по времени вычисления заданного числа кандидатов. Соотношение этих промежутков времени (чуть более 17 раз) оказалось замечательно согласованным с теоретическим результатом, вычисленным выше. То, что выигрыш оказался несколько меньше, объясняется тем, что, кроме операций замыкания, в этих алгоритмах имеются еще операции побитового умножения, которые хотя и очень быстрые, тем не менее остаются в неизменном количестве. За подробностями читатель отсылается к [28].

5. Алгебраическое машинное обучение

Будем использовать классическую схему машинного обучения: сначала разделение выборки на обучающую и тестовую, потом извлечение знаний (индуктивное порождение гипотез), наконец, тестирование качества с помощью предсказания целевого свойства у тестовых примеров.

В классическом ДСМ-методе [1] имеется дополнительная процедура — абдуктивное принятие гипотез, но в нашем вероятностном случае она в значительной степени теряет свой смысл, так как некоторые гипотезы могут не возникнуть. Более того, ВПК-парадигма имеет свой способ обеспечения надежности результата, который в нашем случае будет составлять утверждение теоремы 7.

Индуктивное обобщение обучающих примеров осуществляется по алгоритму 6.

Алгоритм 6 (Индукция).

1. Зафиксируем число N порождаемых гипотез (например, из теоремы 7).

2. Сформируем обучающую выборку $I \subseteq O \times F$, где $O :=$ (+)-примеры, $F :=$ признаки. Сформируем список контрпримеров $C :=$ (–)-примеры.

3. Создадим несколько нитей для порождения гипотез, применяя алгоритм 5.

4. Пока число гипотез не достигнет заданного числа N, вычислять в различных нитях (threads) вычислений п. 5.

5. Если в нити возник кандидат $\langle A, B \rangle$, проверять наличие контрпримера: Если для некоторого $c \in C$ выполняется $B \subseteq c'$, то запустить заново алгоритм 5. Если контрпримеров не нашлось, то выдать $\langle A, B \rangle$ в качестве еще одной гипотезы.

После индуктивного обобщения обучающих примеров можно использовать тестовые примеры для проверки качества порожденных гипотез.

Алгоритм 7 (Аналогия).

1. Сформируем список тестовых примеров $X :=$ (τ)-примеры.

2. Во внешнем цикле выбираем текущий тестовый пример $o \in X$. Временно объявляем его отрицательным примером.

3. Во внутреннем цикле проверяем вложение фрагмента B гипотезы $\langle A, B\rangle$, порожденной алгоритмом 6, в описание тестового примера $B \subseteq o'$. Если $B \subseteq o'$, объявить пример o положительным; внутренний цикл прервать.

4. Если вложения не нашлось, то объект o останется отрицательным.

Алгоритм 7 называется Аналогией потому, что объявленный положительным тестовый пример o содержит в своем описании фрагмент B гипотезы $\langle A, B\rangle$, который также содержится в описаниях всех обучающих примеров из списка родителей $A \subseteq O$. Если верно, что была выявлена причина целевого свойства как сходство всех обучающих примеров из A, то и тестовый пример доопределен правильно по аналогии с ними.

Определение 9. ***Нижнее полупространство*** $H^{\downarrow}_{1/2}(o)$, *определяемое объектом* o *с фрагментом* $o' \subseteq F$, *задается линейным неравенством* $x_{j_1} + \ldots + x_{j_k} < \frac{1}{2}$, *где* $F \setminus o' = \{f_{j_1}, \ldots, f_{j_k}\}$.

Здесь переменные $x_1, \ldots, x_n \in \mathbb{R}$ находятся во взаимно-однозначном соответствии с признаками $F = \{f_1, \ldots, f_n\}$.

Так как обучающие и тестовые примеры и фрагменты лежат в $\{0,1\}^n$, то можно ограничиться булевым кубом.

Лемма 11. Пример o *предсказывается положительным с помощью гипотезы* $\langle A, B\rangle$ $(B \subseteq o')$ *тогда и только тогда, когда в его нижнем полупространстве* $H^{\downarrow}_{1/2}(o)$ *содержится точка, соответствующая* B.

Лемма 11 фактически переворачивает ВПК-парадигму: если в исходной постановке Вапника–Червоненкиса гипотезы были фиксированы и соответствовали подмножествам пространства обучающих примеров, а примеры выбирались независимо и случайно (что может быть невыполнимо на практике), то у нас тестовые примеры соответствуют нижним полупространствам булева куба, а фрагменты случайно и независимо порождаемых гипотез — точкам в этих полупространствах.

Зафиксируем $\varepsilon > 0$ – точность предсказания.

Определение 10. Объект o *назовем* ε-***важным***, *если суммарная вероятность появления таких гипотез* $\langle A, B\rangle$, *что* $B \in H^{\downarrow}_{1/2}(o)$, *будет больше* ε.

Теорема 7. Для n *признаков и любых* $\varepsilon > 0$ *и* $1 > \delta > 0$ *достаточно породить*

$$N \geqslant \frac{n \cdot \ln 2 - \ln \delta}{\varepsilon}$$

гипотез, чтобы с вероятностью, большей $1 - \delta$, *все* ε*-важные объекты были предсказаны положительно.*

Доказательство. Докажем, что при $N \geqslant \frac{n \cdot \ln 2 - \ln \delta}{\varepsilon}$ гипотезах вероятность того, что хотя бы один ε-важный пример будет пропущен, не превзойдет δ.

Пусть o будет ε-важным примером, а $H^{\downarrow}_{1/2}(o)$ — соответствующим нижним полупространством. Тогда $\mathbb{P}[o$ не доопределится одной гипотезой$] = 1 - \mathbb{P}[o$ доопределится первой гипотезой$] \leqslant 1 - \varepsilon$.

Поэтому $\mathbb{P}[o$ не доопределится N гипотезами$] \leqslant (1-\varepsilon)^N$ из-за независимости гипотез.

Так как различных нижних полупространств ровно 2^n, то по неравенству Буля $\mathbb{P}[$хотя бы один ε-важный o не доопределится N гипотезами$] \leqslant 2^n \cdot (1-\varepsilon)^N$.

Но $2^n \cdot (1-\varepsilon)^N \leqslant 2^n \cdot e^{-N\cdot\varepsilon} \leqslant 2^n \cdot e^{\ln\delta - n\cdot\ln 2} = \delta$, что завершает доказательство.

В публикации [25] использовалось прямое сведение к технике Вапника–Червоненкиса [5] "метод повторной выборки", что дало более грубую оценку. К тому же в ее формулировке там имеется опечатка.

Значение этой теоремы — оценка на число требуемых гипотез в алгоритме 6. В принципе, эту оценку можно немного улучшить, если заметить, что можно рассматривать только минимальные по включению примеры: они образуют антицепь в булевой алгебре, по теореме Шпернера их число не превосходит $\binom{n}{\lfloor \frac{n}{2} \rfloor}$, а не 2^n. Но это рассуждение дает дополнительное слагаемое $\left(-\frac{1}{2}\right) \cdot \ln\left(\frac{n\cdot\pi}{2}\right)$ в числителе, что не меняет порядок величины.

6. Заключение

Сформулируем критерий того, когда процедура приобретает знания, и обсудим, что такое знание в противовес простой информации. Эти вопросы стали актуальными в связи с широкими философскими дискуссиями об определении интеллектуальных систем.

Представляется правильным посмотреть на это с точки зрения теории сложности вычислений: знания — это решение NP-трудных задач, а процедура способна приобретать знания, если она способна вычислительно эффективно решать класс NP-трудных задач (хотя бы приближенно).

Чтобы продемонстрировать отличие знаний от информации, рассмотрим две известные классические проблемы: обобщенную задачу Эйлера о Кенигсбергских мостах и задачу коммивояжера. В случае задачи Эйлера вопрос о построении требуемого маршрута решается жадным алгоритмом (а возможность проверяется за линейное время). В случае коммивояжера даже для существования маршрута необходимо решить NP-полную задачу. Решение такой задачи, если оно известно, следует хранить и передавать другим, так как в случае утери его нельзя будет быстро восстановить из исходных данных.

ДСМ-метод [1] может приобретать знания, но ценой экспоненциального времени работы. С точки зрения эффективности ДСМ-метод может работать на выборках очень малого объема. Напротив, описанный в этой статье метод алгебраического машинного обучения может считаться технологией приоб-

ретения знаний для выборок среднего объема и даже “BigData”, но ключевой проблемой в этом случае становится кодирование объектов со сложной структурой битовыми строками так, чтобы пересечение (побитовое умножение) выявляло бы общие фрагменты.

Альтернативой описанному подходу может служить метод последовательного расширения решетки кандидатов [29], развиваемый в последнее время С.О. Кузнецовым с коллегами.

Автор благодарен О.П. Кузнецову и С.О. Кузнецову за полезные обсуждения и поддержку и рецензентам настоящей статьи, которые способствовали существенному улучшению изложения. Особо хочу поблагодарить свою аспирантку Л.А. Якимову за совместную работу, однако за все неточности и ошибки, как обычно, несет ответственность исключительно автор.

СПИСОК ЛИТЕРАТУРЫ

1. *Финн В.К., Аншаков О.М.* (ред.) ДСМ-метод автоматического порождения гипотез: Логические и эпистемологические основания. М.: Эдиториал УРСС, 2009.
2. *Ganter B., Wille R.* Formal Concept Analysis. Transl. from German. Berlin: Springer-Verlag, 1999.
3. *Davey B.A., Priestley H.A.* Introduction to Lattices and Order. 2nd eds. Cambridge: Cambridge University Press, 2002.
4. *Valiant L.G.* A Theory of Learnable // Communications of the ACM. 1984. V. 27. No. 11. P. 1134–1142.
5. *Вапник В.Н., Червоненкис А.Я.* Теория распознавания образов (статистические проблемы обучения). М.: Наука, 1974.
6. *Rivest R.L.* Learning Decision Lists // Machine Learning. Springer. 1987. V. 2. No. 3. P. 229–246.
7. *Borchmann D., Hanika T., Obiedkov S.* Probably approximately correct learning of Horn envelopes from queries // Discrete. Appl. Math. 2020. V. 273. No. 1. P. 30–42.
8. *Yarullin R., Obiedkov S.* From Equivalence Queries to PAC Learning: The Case of Implication Theories // Int. J. Approx. Reason. 2020. V. 127. P. 1–16.
9. *Милль Дж.Ст.* Система логики силлогистической и индуктивной: Изложение принципов доказательства в связи с методами научного исследования. Пер. с англ. Изд. 5. М.: Эдиториал УРСС, 2011.
10. *Hunt E.B., Marin J., Stone P.J.* Experiments in Induction. N.Y.: Academic Press, 1966.
11. *Vere S.A.* Induction Of Concepts In The Predicate Calculus // Proc. 4th Int. Joint Conf. on Artificial Intelligence IJCAI-75. 1975. Tbilisi: JICAI. P. 281–287.
12. *Финн В.К.* О машинно-ориентированной формализации правдоподобных рассуждений в стиле Ф. Бэкона–Д.С. Милля // Семиотика и информатика. 1983. Вып. 20. С. 35–101.
13. *Финн В.К.* Об одном варианте логики аргументации // НТИ. Сер. 2. 1996. № 5–6. С. 3–19.
14. *Виноградов Д.В.* Метод семантических таблиц для логики аргументации // НТИ. Сер. 2. 2006. № 11. С. 17–20.

15. *Финн В.К.* Индуктивные методы Д.С. Милля в системах искусственного интеллекта. Ч. I // Искусственный интеллект и принятие решений. 2010. № 3. С. 3–21.

16. *Финн В.К.* Индуктивные методы Д.С. Милля в системах искусственного интеллекта. Ч. II // Искусственный интеллект и принятие решений. 2010. № 4. С. 14–40.

17. *Кузнецов С.О.* Быстрый алгоритм построения всех пересечений объектов из нижней полурешетки // НТИ. Сер. 2. 1993. № 1. С. 17–20.

18. *Кузнецов С.О., Финн В.К.* Об одной модели обучения и классификации, основанной на операции сходства // Обозрение Прикладной и Промышленной Математики. 1996. Т. 3. № 1. С. 66–90.

19. *Ganter B., Kuznetsov S.O.* Formalizing Hypotheses with Concepts // Proc. 8th Int. Conf. on Conceptual Structures ICCS–2000 (Mineau, G., Ganter, B. Eds.), Lecture Notes in Artifical Intelligence (Springer). 2000. V. 1867. P. 342–356.

20. *Виноградов Д.В.* О представлении объектов битовыми строками для ВКФ-метода // НТИ. Сер. 2. 2018. № 5. С. 1–4.

21. *Виноградов Д.В.* Скорость сходимости к пределу вероятности порождения случайного сходства при наличии контрпримеров // НТИ. Сер. 2. 2018. № 2. С. 21–24.

22. *Якимова Л.А.* Экспериментальное исследование поведения решателей, основанных на бинарной операции сходства. Магистерская диссертация по направлению подготовки 45.04.04 (Науч. руков. Виноградов Д.В.). М.: РГГУ, 2020.

23. *Dua D., Graff C.* UCI Machine Learning Repository. Irvine, CA:
[http://archive.ics.uci.edu/ml]. 2019.

24. *Сидорова Е.Ю.* Экспериментальная реализация вероятностных алгоритмов для поиска ДСМ-сходств. Дипломная работа по направлению подготовки 036000 (Науч. руков. Виноградов Д.В.). М.: РГГУ, 2013.

25. *Vinogradov D.V.* Machine Learning Based on Similarity Operation // Communications in Computer and Information Science, Springer. 2018. V. 934. P. 46–59.

26. *Кемени Дж., Снелл Дж.* Конечные цепи Маркова. Пер. с англ. М. Наука, 1970.

27. *Виноградов Д.В.* Сильная концентрация времени работы алгоритма поиска сходств // Матер. 8 Всеросс. мультиконф. по проблемам управления (МКПУ-2015) 2015. Т. 1. С. 42–45.

28. *Якимова Л.А.* Реализация ленивой схемы вычислений сходств в ВКФ-методе. Выпускная квалификационная работа бакалавра по направлению подготовки 45.03.04 (Науч. руков. Виноградов Д.В.). М.: РГГУ, 2018.

29. *Kuznetsov S., Napoli A., Makhalova T.* Gradual Discovery with Closure Structure of a Concept Lattice // Proc. 15th Int. Conf. CLA-2020. 2020. CEUR-WS. Iss. 2668. P. 145–157.

Статья представлена к публикации членом редколлегии О.П. Кузнецовым.

Поступила в редакцию 07.12.2021

После доработки 02.01.2022

Принята к публикации 26.01.2022

© 2022 г. С.А. ШУМСКИЙ, канд. физ.-мат. наук
(serge.shumsky@gmail.com)
(Московский физико-технический институт)

ADAM — МОДЕЛЬ ИСКУССТВЕННОЙ ПСИХИКИ[1]

Предложена модель искусственной психики ADAM, реализующая иерархическую архитектуру глубокого обучения с подкреплением. ADAM способен обучаться все более сложным и протяженным во времени поведенческим навыкам по мере увеличения количества управляющих уровней искусственной психики. Целенаправленное поведение формируется иерархической обучающейся системой с постепенным наращиванием числа уровней, где каждый иерархический уровень ответственен за свой временной масштаб поведения.

Ключевые слова: общий искусственный интеллект, глубокое обучение с подкреплением, иерархическая система управления.

DOI: 10.31857/S0005231022060034, **EDN:** ACIMYZ

1. Введение

Под искусственным интеллектом (ИИ) обычно понимают алгоритмы решения различных интеллектуальных (когнитивных) задач на уровне человека или лучше. В разные времена под "интеллектуальными" понимались разные типы задач. В 1950-е гг. таковыми считались "творческие" задачи, в которых невозможно предусмотреть заранее все варианты решений: игра в шахматы, доказательство теорем, машинный перевод. С течением времени область ИИ расширялась и пополнялась другими типами когнитивных задач, уже не связанными с логическим интеллектом, например задачи распознавания образов и моделирования целесообразного поведения животных [1]. Однако все современные системы машинного интеллекта имитируют каждая лишь какую-то одну очень узкую область человеческих способностей, т.е. являются *слабым ИИ*. Задача создания *сильного ИИ*, способного конкурировать с человеком во всех областях, до недавнего времени на практике даже и не ставилась. Считалось, что это проблема очень отдаленного будущего.

Перелом во взглядах ИИ-сообщества на сильный ИИ произошел в последние несколько лет после свершившейся в 2010-х *революции глубокого обучения*. В ходе этой (все еще продолжающейся) революции происходит смена

[1] Работа выполнена при частичной финансовой поддержке Центра компетенций Национальной технологической инициативы по направлению "Искусственный интеллект" при МФТИ.

основной парадигмы ИИ. Мейнстрим ИИ сместился в область обучения искусственных нейросетей, и место ИИ, основанного на человеческих знаниях, занимает теперь ИИ, основанный на машинном обучении, которому удается решать практически все задачи ИИ в единой методологии, причем с гораздо лучшим качеством, чем прежде [2]. Лидеры революции глубокого обучения сегодня предсказывают переход от моделирования систем бессознательного сенсорного интеллекта к по-настоящему разумным машинам, самостоятельно планирующим свое поведение и "понимающим", что и зачем они делают [3]. Появление таких разумных машин создаст новый массовый рынок автономных роботов, способных к обучению, в отличие от современных роботов с программируемым поведением.

Иными словами, главной задачей следующего этапа развития ИИ является синтез всех видов интеллекта — сенсорного, моторного, стратегического и других в единой *искусственной психике*, называемой в англоязычной литературе *общим ИИ — Artificial General Intelligence* (AGI). Именно такую цель — создание искусственной психики роботов, позволяющей им самостоятельно планировать достижение поставленных целей и осуществлять эти планы, адаптируясь к изменяющейся обстановке, — ставит перед собой лаборатория Когнитивных архитектур МФТИ.

2. Когнитивные архитектуры

Искусственная психика представляет собой целостную систему со своей *когнитивной архитектурой*, которая определяет все ее базовые свойства. Поэтому разработка искусственной психики, как и любой сложной системы, должна начинаться именно с проектирования ее архитектуры. Как и архитектура фон-Неймановских компьютеров, когнитивная архитектура подразумевает исполнение самых разных алгоритмов. В традиционных когнитивных архитектурах эти алгоритмы (включающие *знания* об устройстве мира и полезные *навыки* поведения в этом мире) были в основном рукотворными [4]. Мы ставим перед собой задачу, чтобы эти алгоритмы не закладывались извне в готовом виде, а возникали в процессе активного взаимодействия искусственной психики с внешней средой путем обучения с подкреплениями. Такая постановка задачи, по нашему мнению и согласно [5], наиболее близка к определению AGI. Отличительной чертой нашего подхода является попытка воспроизвести в нашей когнитивной архитектуре базовые черты вычислительной архитектуры человеческого мозга.

Несколько огрубляя, базовая схема современных когнитивных архитектур может быть суммирована в так называемой *стандартной модели интеллекта* [6], описывающей схему взаимодействия основных типов когнитивных модулей искусственной психики. Как видно из рис. 1, связующим звеном между всеми когнитивными модулями является оперативная рабочая память, соответствующая текущей активности в коре мозга. Содержимое рабочей памяти контролируется моделями поведения в базальных ганглиях, управляющих текущей активностью коры — операциями, хранящимися в долговременной

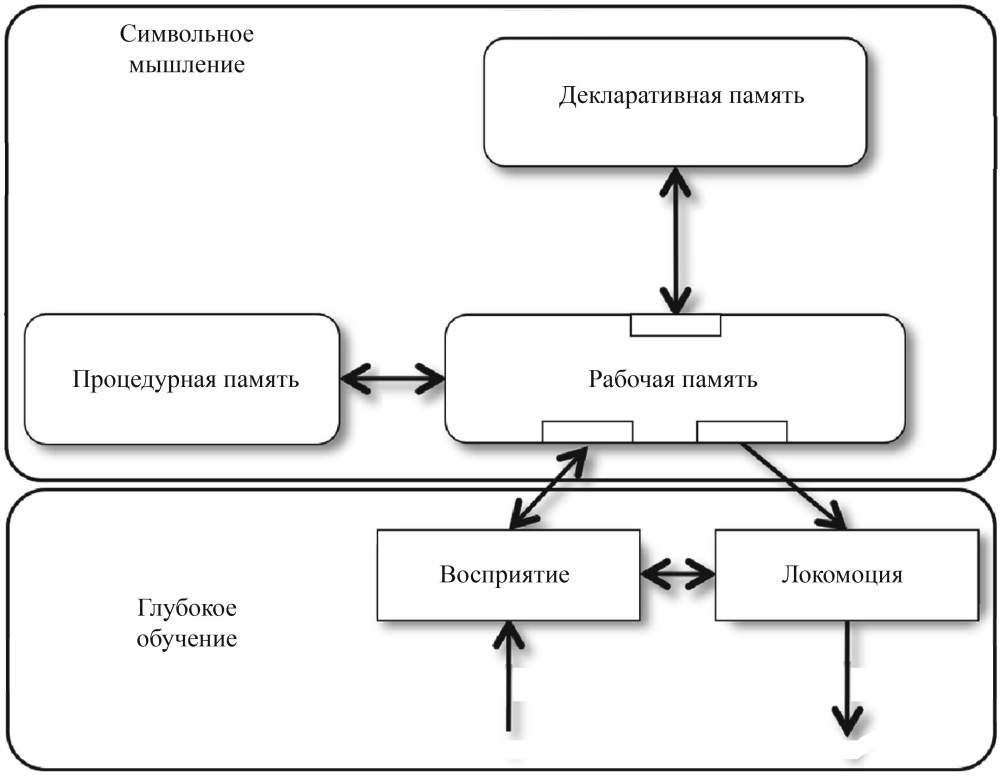


Рис. 1. Стандартная модель интеллекта (Standard Model for the Mind [6]).

процедурной памяти. Элементы долговременной декларативной памяти коры при активации поступают в рабочую память. Все когнитивные архитектуры, объединяемые стандартной моделью, используют знания, представленные в символьной форме (факты и правила). Интерфейс символьной рабочей памяти с векторным физическим пространством обеспечивается специальными кодирующими и декодирующими модулями — соответственно сенсорными и моторными, в качестве которых могут выступать современные глубокие нейросети.

Важнейшим элементом стандартной модели является идея *когнитивного акта*, стандартной операции выбора и исполнения одного из правил процедурной памяти. Любое сколь угодно сложное поведение состоит из таких элементарных когнитивных актов длительностью в десятые доли секунды. Вся сложность нашего мышления и поведения возникает в результате правильно подобранных цепочек элементарных когнитивных актов. Стандартная модель суммирует наши знания о механизмах работы мозга и структуре нашей психики. Но она не дает ответа, каким именно образом выстраиваются неимоверно длинные осмысленные цепочки когнитивных актов, как организовано планирование нашего поведения на больших масштабах времени — от минут до дней, месяцев и даже лет (см. рис. 2).

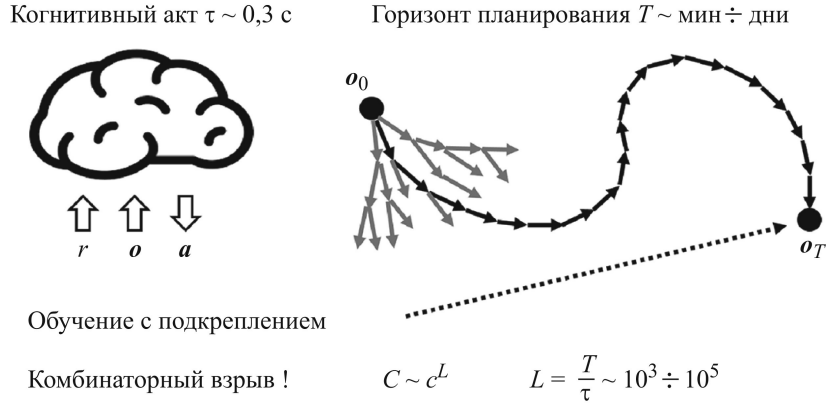


Рис. 2. Основная проблема машинного мышления — переход от единичного когнитивного акта к большим горизонтам планирования. Здесь: r – подкрепления, $\boldsymbol{o}$ – наблюдения, $\boldsymbol{a}$ – действия, C – разнообразие возможных цепочек действий, характеризующее сложность поиска оптимальной стратегии, c – разнообразие действий на каждом шаге, L – количество шагов на горизонт планирования T.

Планы в стандартной модели могут задаваться в виде иерархии правил процедурной памяти, где отдельные действия могут содержать в себе различные этапы. Но эти иерархии правил закладываются в них вручную, а не возникают автоматически, в отличие от иерархии признаков, автоматически возникающих в результате обучения глубоких нейросетей. Как пишет автор классического современного учебника по ИИ Стюарт Рассел: “В настоящее время все существующие методы иерархического планирования опираются на сгенерированные человеком иерархии абстрактных и конкретных действий. Мы еще не понимаем, как такие иерархии могут быть получены путем обучения” [7].

Действительно, хотя в последнее десятилетие ручное программирование правил поведения и уступает место глубокому обучению с подкреплением, обеспечив тем самым прорыв в уровне стратегического игрового интеллекта, иерархическое планирование в глубоком обучении до сих пор отсутствует. Так, успех известной программы AlphaZero обеспечивается потрясающей интуицией ее глубокой нейросети, обученной правильно оценивать любую игровую позицию и находить в ней наилучшие варианты ходов. Однако глубокая нейросеть AlphaZero способна генерировать варианты своих ходов лишь на один шаг вперед. Для выбора наилучшего варианта на каждом шаге AlphaZero производит просчет очень объемного дерева вариантов на десятки ходов вперед [8]. Это обеспечивает отличное качество игры, но очень дорогой ценой из-за комбинаторного взрыва числа возможных комбинаций, перебираемых методом грубой силы.

Человеческое мышление устроено по-другому. Мы не перебираем в уме все возможные варианты цепочек когнитивных актов, что было бы практи-

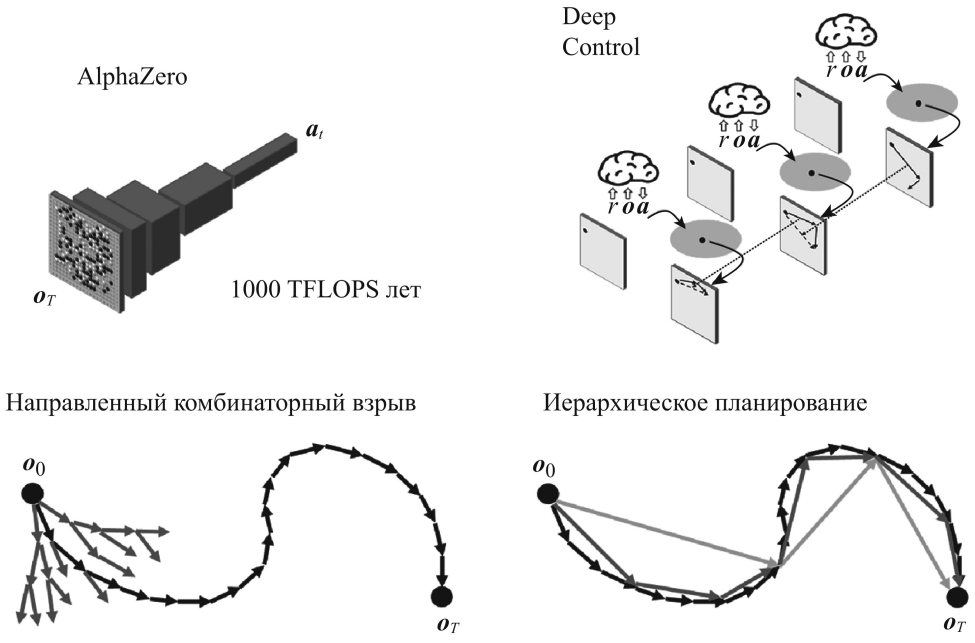


Рис. 3. Пошаговый просчет оптимальной траектории программой AlphaZero (слева) и иерархическое планирование поведения от общего замысла ко все более детальным планам в архитектуре Deep Control (справа).

чески невозможно. Вместо этого мы используем иерархии планов: от крупномасштабного замысла достижения цели — ко все более подробным планам его достижения. При этом разнообразие вариантов выбора на каждом уровне планирования относительно невелико, а детализируются лишь те этапы, которые реализуются в данный момент (см. рис. 3, справа внизу). Именно так планируют свое поведение люди, и именно так устроено планирование в предложенной автором когнитивной архитектуре Deep Control [9], где в процессе обучения автоматически формируется иерархия правил поведения по аналогии с глубоким обучением, автоматически формирующим иерархии признаков при распознавании образов. Тем самым решается проблема формирования разумного поведения с большим горизонтом планирования, т.е. перекидывается мостик от простейшей психики животных к сложно организованному символьному мышлению человека. В отличие от стандартной модели интеллекта в архитектуру Deep Control иерархичность встроена в явном виде, отражая иерархичность, присущую кортико-стриарной системе нашего мозга, управляющей нашим поведением и обучающейся с помощью дофаминовых подкреплений [10] (см. рис. 4).

Другим определяющим принципом архитектуры Deep Control является предиктивное управление, согласно которому наш мозг постоянно предсказывает будущее в контексте своих собственных управляющих воздействий

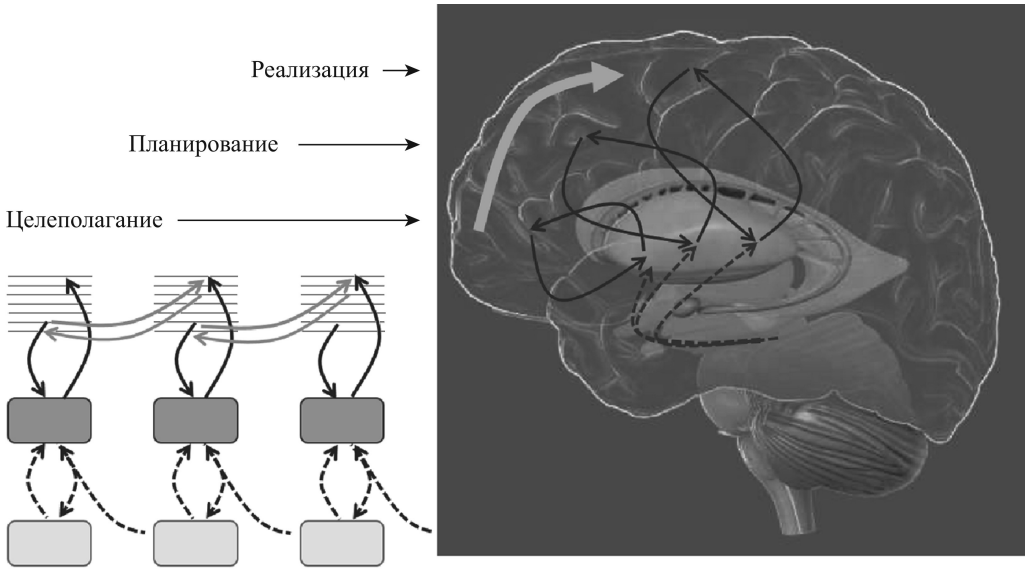


Рис. 4. Кортико-стриарная система мозга, управляющая целесообразным поведением, устроена иерархически. Программа поведения формируется в мозге от мотивации к планированию и далее — к реализации. Все уровни иерархии имеют одинаковый набор модулей, представленных в различных частях коры, базальных ганглий и дофаминовой системы среднего мозга.

(рис. 4, серые стрелки). Эта особенность нашего мозга хорошо изучена и положена в основу многих теоретических моделей мышления [11, 12], в отличие от которых нас интересует действующая модель искусственной психики, являющаяся результатом обратного инжиниринга архитектуры мозга.

3. Архитектура Deep Control: иерархическое планирование поведения

В архитектуре Deep Control проблема, о которой говорит Стюарт Рассел, — обучение роботов иерархическому планированию поведения — решается с использованием оригинальной технологии *глубокого структурного обучения* [9], а именно: управление поведением на разных временных масштабах осуществляется в разных вычислительных слоях. Чем выше слой, тем большим временным масштабом он оперирует, решая, по существу, одну и ту же типовую задачу, как показано на рис. 5. Каждый слой управляет взаимодействием с внешним миром, предсказывая свое очередное дискретное состояние, кодирующее на своем временном масштабе *сенсомоторную* информацию — как входящую (*наблюдения*), так и исходящую (*действия*), т.е. любой план действий сопровождается соответствующими предсказаниями наблюдений, которые постоянно сравниваются с реальностью, поставляя материал для обучения даже в отсутствие подкрепляющих сигналов, что выгодно отличает Deep Control от обычного глубокого обучения с подкреплением. На рис. 5

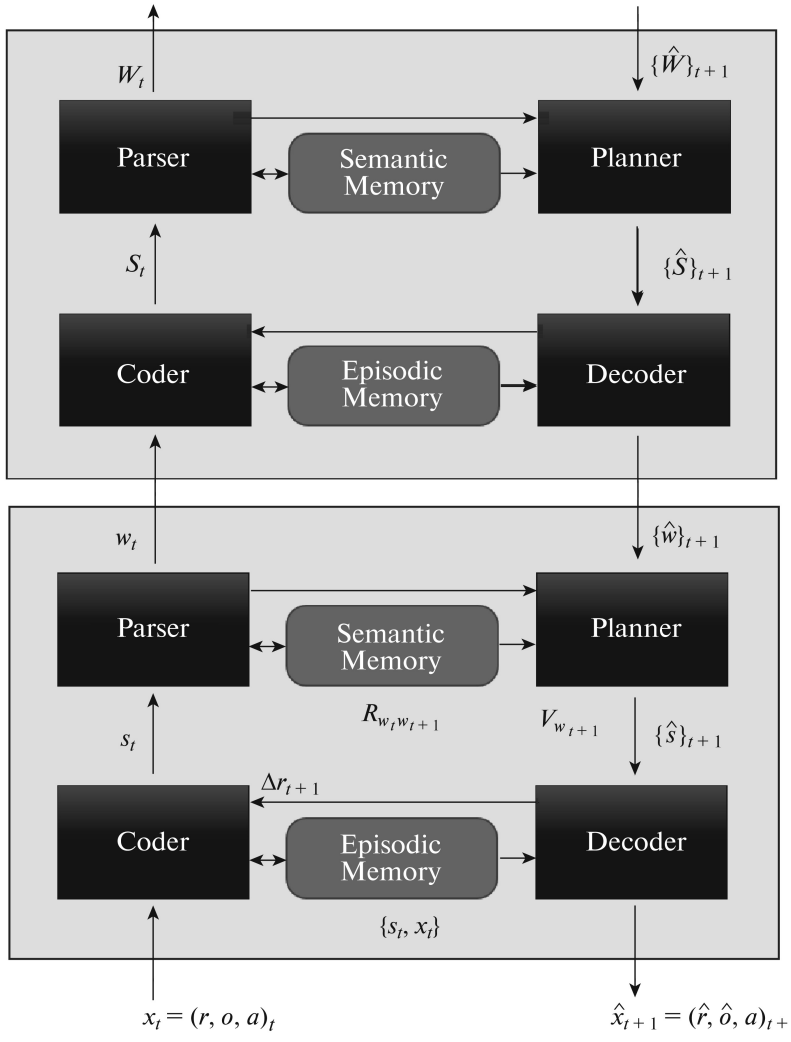


Рис. 5. Архитектура Deep Control представлена однотипными вычислительными слоями, осуществляющими управление поведением каждый на своем временном масштабе. Чем выше слой, тем бóльшим временны́м масштабом он оперирует. Каждый слой находит на своем масштабе решение локальной задачи, поставленной для него более высоким слоем, разбивая ее на подзадачи для более низкого слоя.

показаны два первых слоя Deep Control, на примере которых мы поясним, как именно происходит управление поведением в этой архитектуре.

Входная информация поступает в управляющую систему (искусственный мозг робота) из внешнего мира в виде единого сенсомоторного вектора $\boldsymbol{x}_t = (r, \boldsymbol{o}, \boldsymbol{a})_t$, объединяющего показания всех сенсоров $\boldsymbol{o}$ и актуаторов $\boldsymbol{a}$ управляемой системы (тела робота) в данный момент дискретного времени. Показания одного из сенсоров выделены в отдельный *подкрепляющий сигнал* r, который служит для обучения системы и обрабатывается особым

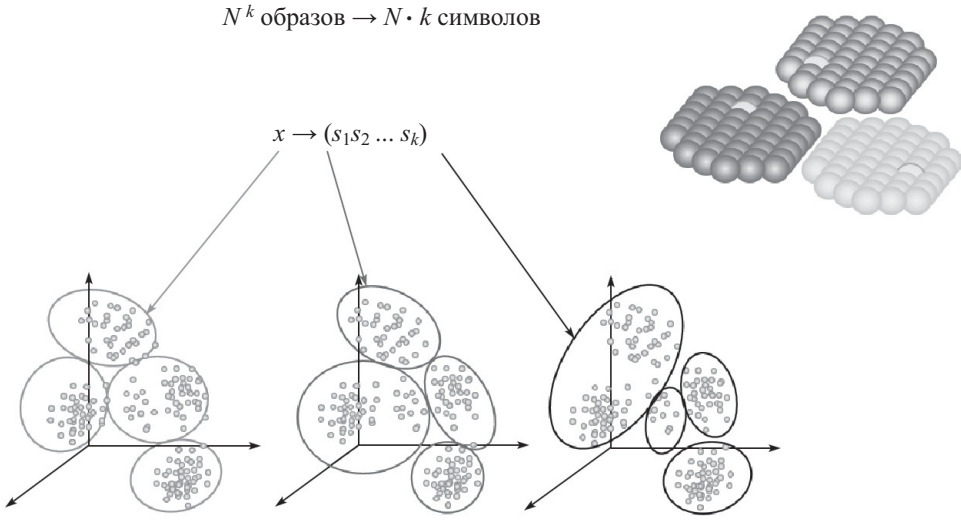


Рис. 6. Дискретное кодирование в Кодере.

образом. В ответ управляющая система выдает прогноз следующего сенсомоторного вектора в очередной момент времени $\hat{\boldsymbol{x}}_{t+1} = (\hat{r}, \hat{\boldsymbol{o}}, \boldsymbol{a})_{t+1}$, а именно: прогноз показаний всех сенсоров, которые она не контролирует, и *реальные* управляющие сигналы для всех актуаторов управляемой системы, которые она контролирует.

Входным элементом каждого слоя является *Кодер*, который кодирует непрерывный векторный сигнал набором дискретных символов $\boldsymbol{x}_t \rightarrow \boldsymbol{s}_t$, т.е. осуществляет дискретное кодирование. В простейшем варианте Кодер состоит из нескольких модулей, каждый из которых производит свой вариант кластеризации входных векторов, сохраненных в *эпизодической памяти*. Как показано на рис. 6, разнообразие дискретных кодов возрастает экспоненциально с числом модулей, так что требуемого для управления разнообразия всегда можно добиться даже с небольшим числом модулей. Так, например, 7 модулей по 30 кластеров в каждом достаточно для кодирования более чем 10^{10} образов (верхняя оценка количества когнитивных актов за всю человеческую жизнь).

В мозге дискретное кодирование производится гиперколонками неокортекса, так как из-за взаимной конкуренции колонок активной в каждой гиперколонке может быть лишь одна колонка. Такие гиперколонки размером порядка 1 мм^2 были экспериментально исследованы Маунткаслом [13], а их теоретическая модель предложена Кохоненом [14] (см. рис. 7).

Дискретный сигнал из Кодера передается в *Парсер*, аналог рабочей памяти в стандартной модели. Парсером обычно называют синтаксический анализатор, структурирующий входные данные. В нашем случае Парсер производит анализ временно́го ряда из поступающих к нему символов, выделяя в нем характерные цепочки символов, *морфемы* $\mathbf{w}_t$. Таким образом, Парсер фор-

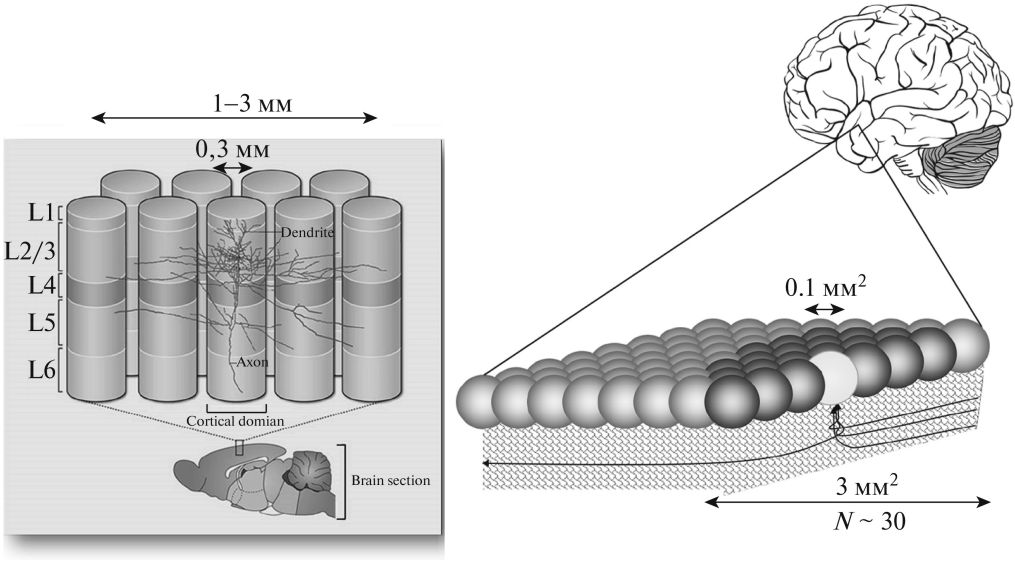


Рис. 7. Дискретное кодирование в неокортексе.

мирует укрупненное описание текущего контекста уже в виде цепочек морфем, а не символов. По мере накопления опыта Парсер обучается выявлять все более крупные морфемы, рекурсивно объединяя между собой наиболее часто встречающиеся пары более коротких морфем, начиная с единичных символов. Выявление и запоминание морфем в мозге может осуществляться гипотетическими *рекурсивными модулями* коры, отличающимися тем, что они "смотрят" сами на себя и поэтому способны кодировать временны́е последовательности (рис. 8).

Силы ассоциативных связей между кодами морфем, отражающие то, как часто морфема $\mathbf{w}'$ следует за морфемой $\mathbf{w}$, образуют *Семантическую память* $R_{\mathbf{ww}'}$, названную так потому, что она отражает характер употребления морфем, от которого только и зависят значения действий. Действия, осуществляемые в сходных ситуациях, т.е. перед и после определенных действий, имеют, очевидно, сходные назначения аналогично тому, как значения слов в языке определяются контекстами их употребления. Семантическая память $R_{\mathbf{ww}'}$ помнит, какие следующие морфемы (т.е. цепочки сенсомоторных состояний) и насколько часто встречались в данном контексте. Как и Кодер, Семантическая память разбита на независимые модули — *головы*, работающие каждая со своим входным алфавитом символов, поступающих от соответствующих модулей Кодера. Модульный дизайн матрицы $R_{\mathbf{ww}'}$ существенно снижает сложность и время вычислений. Это аналог нашей модели мира, хранящейся в неокортексе, предположительно — в рекурсивных модулях.

К этой модели мира надо добавить еще и модель своих собственных предпочтений — насколько желанны для нас различные состояния мира в контексте наших действий. Эти предпочтения, выявляемые в процессе обучения

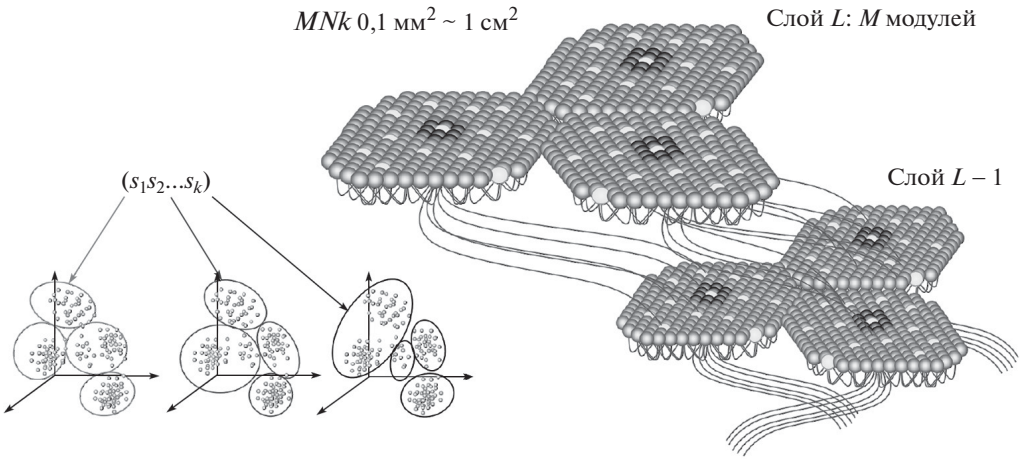


Рис. 8. Гипотетические рекурсивные модули неокортекса, соответствующие архитектуре Deep Control. Центральная гиперколонка каждого модуля с глобальными связями кодирует свою входную информацию активностью одной из своих кортикальных колонок. Окружающие ее гиперколонки с локальными связями кодируют последовательности таких символов в центральной гиперколонке, т.е. морфемы.

с подкреплением, задаются отдельной *функцией ценности* $V_{\mathbf{w}}$, за которую в мозге отвечают базальные ганглии. Семантическая память и функция ценности обучаются Парсером, который распознает и порядок следования морфем, и соответствующие им подкрепления. В простейшем случае используется алгоритм обучения SARSA [15]:

$$V(\mathbf{w}) \leftarrow V(\mathbf{w}) + \alpha(r + \gamma V(\mathbf{w}') - V(\mathbf{w})).$$

Модель мира и модель наших предпочтений используются *Планировщиком* для планирования поведения. Если вычислительный слой является самым верхним, то информация о текущем контексте, распознанным Парсером, передается непосредственно Планировщику. Зная последнюю распознанную морфему, какие морфемы могут следовать за ней и ценность каждой такой морфемы-кандидата, Планировщик выбирает оптимальную в данном контексте следующую морфему, которая и становится его текущим планом.

Этот план затем пошагово транслируется *Декодеру*. В первом слое Декодер переводит его из символьной формы в векторную — предсказывает следующий сенсомоторный вектор, т.е. непосредственно взаимодействует с внешним миром. В остальных слоях Декодер формирует с помощью своей эпизодической памяти планы для Планировщика нижележащего слоя — ранжированный список возможных морфем-кандидатов, из которого последний выбирает оптимальную для текущего момента морфему. Тем самым каждый слой выбирает наилучший вариант исполнения планов вышележащего слоя с учетом поступающей от нижележащего слоя входной информации.

Информация от нижележащего слоя передается в вышележащий слой Парсером в момент распознания им очередной морфемы. Эта морфема

передается Кодеру вышележащего слоя в форме семантического вектора для его последующего дискретного кодирования. Семантический вектор $\mathbf{X_w} = R_{...\mathbf{w}} R_{\mathbf{w}...}$ каждой морфемы определяется частотами морфем, предшествующих и следующих за данной морфемой. Таким образом, морфемы, употребляемые сходным образом, будут иметь близкие семантические векторы и соответственно будут закодированы Кодером следующего слоя близкими дискретными кодами с большим числом одинаковых компонент.

Итак, мы описали в общих чертах, как в архитектуре Deep Control происходит формирование иерархии планов поведения, вложенных друг в друга и постоянно адаптирующихся к изменяющимся внешним обстоятельствам. Заметим, что от нас не требовалось задавать никаких правил поведения. Все паттерны поведения, кодируемые морфемами на каждом временном масштабе, появляются автоматически в процессе активного взаимодействия системы с внешним миром на основе полученных при этом данных. Искусственный мозг с такой архитектурой способен автоматически формировать свою картину мира и постоянно совершенствовать свое поведение, нацеленное на максимизацию ожидаемого потока подкреплений. Он не ограничен решением какой-то одной определенной задачи и может накапливать опыт решения разных задач в разных контекстах, постоянно накапливая знания о мире и своем опыте взаимодействия с ним. Можно сказать, что он обладает свободой воли, так как он исполняет лишь свои собственные планы, вырабатываемые на верхнем (мотивационном) слое иерархии. Управление его поведением, например нацеливание на решение определенной задачи, происходит не директивно, а через управление подкреплениями, например увеличением соответствующей награды, чтобы повлиять на его текущую мотивацию.

4. ADAM: прототип искусственной психики роботов

ADAM (Adaptive Deep Autonomous Machine) представляет собой действующий прототип искусственной психики с архитектурой Deep Control, разрабатываемый в лаборатории Когнитивных архитектур МФТИ с целью проверить работоспособность предложенного подхода и апробировать различные алгоритмы обучения для новой архитектуры [16]. Разработанная и отлаженная в рамках проекта ADAM архитектура сильного ИИ может использоваться в самых разных сервисах и продуктах, требующих креативного машинного мышления. Мы надеемся, что эта архитектура будет положена в основу будущих операционных систем роботов и специализированных чипов их искусственного мозга.

ADAM представляет собой программу на языке Julia, управляющую поведением робота или программного агента в симуляторе реальности:

$$\textbf{Environment}\,(\mathbf{a}, \mathbf{o})_t \rightarrow (r, \mathbf{o})_{t+1}.$$

Псевдокод ADAM может быть представлен в следующем виде:

ADAM(parameters,$(r, \mathbf{o}, \mathbf{a})_{1:t}$) $\rightarrow (r, \mathbf{o}, \mathbf{a})_{t+1}$
track_record[1] $\leftarrow$ **explore_environment**() # gather initial learning data
adam $\leftarrow$ **new_adam**(parameters) # create new adam
adam.layer[L=1] $\leftarrow$ **new_layer**(adam, track_record[1]) # adam's first layer
while(stop_criteria)
$\hat{r}, \hat{\mathbf{o}}, \mathbf{a} \leftarrow$ **predict!**(adam, $r, \mathbf{o}, \mathbf{a}$) # generate new action/prediction
create next layer if needed:
if(expand_criteria) adam.layer[L+1] $\leftarrow$
$\leftarrow$ **new_layer**(adam, track_record[L+1])

Код ADAM тестируется на задачах из коллекции OpenAI Gym, а также используется в прикладных задачах, в частности — в автоматической биржевой торговой системе, разрабатываемой в МФТИ. Предварительные результаты показывают, что ADAM действительно обучается достигать решения задач за все более короткое время, как показано на рис. 9. В частности, в задаче CartPole требуется научиться балансировать обратный маятник в те-

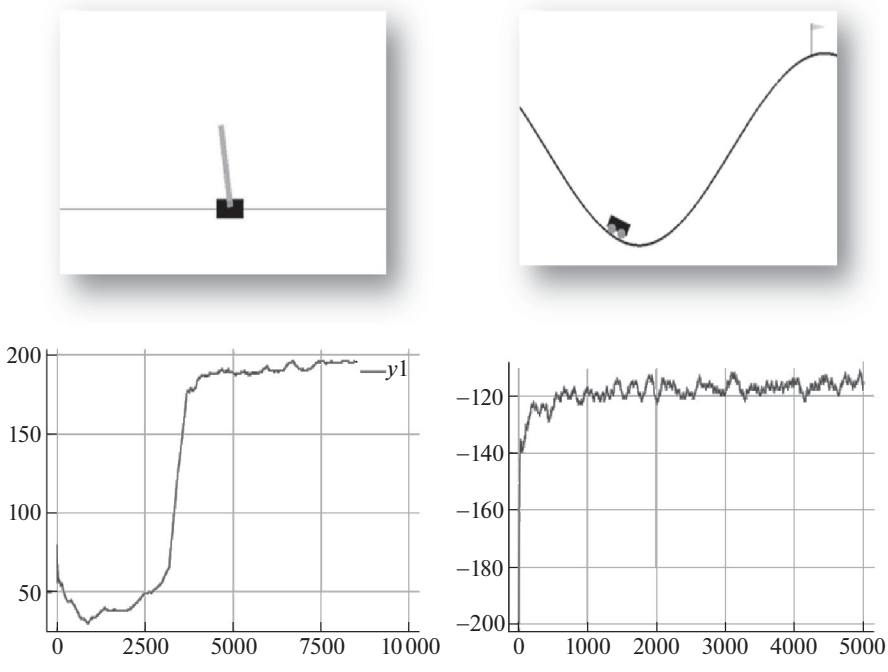


Рис. 9. Пример обучения кода ADAM с одним вычислительным слоем задачам CartPole и MountainCar из библиотеки OpenAI Gym. Ось абсцисс — количество эпизодов обучения, ось ординат — количество подкреплений, полученных в каждом эпизоде.

чение как минимум 200 тактов, а в задаче MountainCar — достигнуть флага, забравшись на склон быстрее, чем за 200 тактов. Причем в первом случае дается единичная награда за каждый такт удачного балансирования, а во втором — отрицательная единичная награда за каждый такт до достижения флага. Как видно из рис. 9, награды, полученные в каждом эпизоде, растут по мере обучения решения обеих задач.

5. Заключение

В данной статье описана модель искусственной психики ADAM с иерархической архитектурой глубокого обучения с подкреплением Deep Control. ADAM способен планировать свое поведение на многих масштабах времени, вписывая планы более низких уровней в планы более высоких и согласовывая поток планов, спускаемых сверху-вниз, с потоком сенсорной информации, поступающей снизу-вверх. По мере накопления опыта взаимодействия с внешней средой и роста числа слоев ADAM обучается целенаправленному поведению на все более долгих временных масштабах. Этот подход может быть использован при создании операционных систем автономных роботов, способных накапливать опыт обучения решению самых разных задач.

СПИСОК ЛИТЕРАТУРЫ

1. *Russell S., Norvig P.* Artificial Intelligence: A Modern Approach. (3rd Edition). Pearson, 2009.
2. *Николенко С., Кадурин А., Архангельская Е.* Глубокое обучение. Погружение в мир нейронных сетей. Питер, 2018.
3. *Bengio Y.* From System-1 Deep Dearning to System-2 Deep Learning // Thirty-third Conf. on Neural Information Processing Syst. 2019.
4. *Kotseruba I., Tsotsos J.K.* 40 Years of Cognitive Architectures: Core Cognitive Abilities and Practical Applications // Artificial Intelligence Review. 2020. V. 53. No. 1. P. 17–94.
5. *Silver D., Singh S., Precup D., Sutton R.S.* Reward is Enough // Artificial Intelligence. 2021. 103535.
6. *Laird J.E., Lebiere C., Rosenbloom P.S.* A Standard Model of the Mind: Toward a Common Computational Framework Across Artificial Intelligence, Cognitive Science, Neuroscience, and Robotics // AI Magazine. 2017. V. 38. No. 4. P. 13–26.
7. *Russell S.* Human Compatible: Artificial Intelligence and the Problem of Control. Viking, 2019.
8. *Silver D., et al.* Mastering chess and shogi by self-play with a general reinforcement learning algorithm // arXiv preprint arXiv:1712.01815. 2017.
9. *Шумский С.А.* Глубокое структурное обучение: новый взгляд на обучение с подкреплением // Сб. науч. тр. XX Всеросс. науч. конф. Нейроинформатика-2018. Лекции по нейроинформатике. С. 11–43. М.: 2018.
10. *Шумский С.А.* Реинжиниринг архитектуры мозга: роль и взаимодействие основных подсистем // Сб. науч. тр. XVII Всеросс. науч. конф. Нейроинформатика-2015. Лекции по нейроинформатике. С. 13–45. М.: 2015.

11. *Clark A.* Surfing Uncertainty: Prediction, Action, and the Embodied Mind. Oxford University Press, 2015.
12. *Friston K.J.* Waves of Prediction // PLoS Biology. 2019. V. 17. No. 10. e3000426.
13. *Mountcastle V.B.* The Columnar Organization of the Neocortex // Brain: a journal of neurology. 1997. V. 120. No. 4. P. 701–722.
14. *Kohonen T.* Self-organized Formation of Topologically Correct Feature Maps // Biological cybernetics. 1982. V. 43. No. 1. P. 59–69.
15. *Sutton R.S., Barto A.G.* Reinforcement learning: An introduction. MIT press, 2018.
16. *Шумский С.А., Басков О.В.* Программный Агент глубокого иерархического обучения с подкреплением ADAM Deep Control // Государственная регистрация программ для ЭВМ. RU 2021660307.

Статья представлена к публикации членом редколлегии О.П. Кузнецовым.

Поступила в редакцию 31.10.2021

После доработки 21.01.2022

Принята к публикации 26.01.2022

© 2022 г. А.О. ИСХАКОВА, канд. техн. наук (shumskaya.ao@gmail.com),
Д.А. ВОЛЬФ, канд. техн. наук (runsolar@mail.ru),
Р.В. МЕЩЕРЯКОВ, д-р техн. наук (mrv@ieee.org)
(Институт проблем управления им. В.А. Трапезникова РАН, Москва)

СПОСОБ СНИЖЕНИЯ РАЗМЕРНОСТИ ПРОСТРАНСТВА ПРИЗНАКОВ ПРИ РАСПОЗНАВАНИИ РЕЧЕВЫХ ЭМОЦИЙ С ИСПОЛЬЗОВАНИЕМ СВЕРТОЧНЫХ НЕЙРОННЫХ СЕТЕЙ[1]

Рассматриваются архитектуры сверточных нейронных сетей, используемые для оценки эмоционального состояния человека по его речи. Решается задача повышения эффективности распознавания эмоций за счет снижения вычислительной сложности данного процесса. Для этого предлагается способ преобразования входных данных в форму, подходящую для алгоритмов машинного обучения.

Ключевые слова: распознавание речевых эмоций, речевой сигнал, звук, идентификация эмоционального состояния, выявление агрессии, классификация речевых сигналов, социо-киберфизическая система, сверточная нейронная сеть.

DOI: 10.31857/S0005231022060046, **EDN:** ACJOSQ

1. Введение

Основным средством человеческого общения является речь, которая содержит характеристические параметры, отражающие в том числе психоэмоциональное состояние говорящего. Распознавание эмоций человека играет важную роль во взаимодействии человека с компьютером, так как оно является дополнительным каналом информации. У людей распознавание эмоций является естественной частью речевого общения, в то время как способность распознавать автоматически с помощью программируемых устройств все еще остается предметом исследований. Возможность автоматического определения эмоций по голосу и речи человека необходима для развития успешных диалоговых систем [1], например, в процессах обучения, мониторинга пожилых людей, людей с ограниченными возможностями, в системах интерактивного развлечения и т.д. Задача идентификации эмоционального состояния человека востребована в различных сферах: телекоммуникации, индустрии развлечений, обучении, медицине и др.

[1] Исследование выполнено при финансовой поддержке Российского фонда фундаментальных исследований в рамках проекта № 18-29-22104.

Решение задачи автоматического распознавания речевых эмоций (РРЭ, англ. SER) с помощью вычислительных систем является предметом исследований ученых на сегодняшний день [2]. Это позволит более эффективно решать задачи определения эмоциональной составляющей мультимедиа материалов, распространяющихся в виртуальной среде. Автоматический анализ содержимого, выявление гнева, злости, агрессии в видео- и аудиоматериалах позволит решить задачу классификации разнородного Интернет-контента по степени его деструктивного воздействия на пользователя [3, 4]. Разработка методов РРЭ в данном случае должна отвечать требованиям, продиктованным соответствующей платформой применения — исследовать материалы быстро и не затрачивая значимых вычислительных ресурсов. При таких условиях становится реальным создание социо-киберфизической системы управления и мониторинга информации в целях противодействия проявлению деструктивного воздействия на пользователей. Повышение скорости вычислений и снижение их сложности возможны за счет уменьшения размерности обучающих данных.

Цель данной статьи — представление нового способа снижения размерности пространства входных признаков, использующего современные сверточные нейронные сети, в задачах распознавания речевых эмоций.

2. Создание обучающих выборок и предобработки при разработке систем РРЭ

В качестве обучающих выборок (данных) в задачах РРЭ обычно используются признаки в виде коэффициентов LPC (Linear Predictive Coding — кодирование с линейным прогнозированием), LPCC (линейные прогнозирующие кепстральные коэффициенты) и MFCC (Mel Frequency Cepstral Coefficients — частотные кепстральные коэффициенты мел). Создается вектор признаков для каждого высказывания путем анализа глобальной статистики (среднее, медиана и т.д.) по всем кадрам [5]. Качество извлечения признаков напрямую влияет на точность распознавания речевых эмоций. Наиболее распространенный метод извлечения признаков включает MFCC [6]. Признаки MFCC получаются в результате применения кратковременного преобразования Фурье (STFT) к исходному сигналу с использованием типа постобработки, которая включает кепстральный анализ. Подробное описание процесса извлечения MFCC-признаков рассматривается в [7, 8]. MFCC были доминирующими функциями, используемыми для распознавания речи сверточными нейронными сетями (СНС). Успех использования СНС был обусловлен их способностью представлять спектр амплитуд речи в компактной форме в качестве информации для обучения и распознавания. Между тем MFCC содержат не только информацию об эмоциональных характеристиках, но и важную информацию о говорящем. Исследования, направленные на то, какие характерные признаки эмоций извлекать из речевого сигнала, имеют большое значение [9].

Недостатком является сложность качественной оценки признаков, что может влиять на снижение точности распознавания. Трудно гарантировать, что хорошие результаты могут быть достигнуты за счет использования различных баз данных, так как люди выражают эмоции по-разному, а признаки однозначного определения эмоций отсутствуют. Успех и производительность методов машинного обучения во многом зависят от выбора представленных данных [10, 11].

На основе выделяемого набора информативных признаков строится классификатор, который обучается на предварительно подготовленном наборе звуковых фрагментов. Наиболее популярными техниками классификации являются следующие: поиск ближайших соседей, метод опорных векторов, скрытые марковские модели, модель смеси нормальных распределений, модели на основе нечеткой логики, байесовские классификаторы максимума вероятности [12].

Классификация эмоциональных состояний производится в соответствии либо с задачами построения анализатора (оценки удовлетворенности, уровня стресса, усталости и т.п.), либо с выбранной моделью описания (набор базовых эмоций, непрерывная классификация и т.п.). Как правило, с ростом числа возможных вариантов классификации точность распознавания эмоциональных состояний снижается. Поэтому количество классов, используемых для обучения, выбирается небольшим.

3. Двумерные СНС для решения задач РРЭ

Основные виды СНС основаны на двух общих архитектурах: AlexNet и GoogleNet [13]. Ключевая идея СНС состоит в локальной связности и распределении весов нейронов, которые объединяются в слои. Каждый нейрон в слое получает входные данные от набора нейронов, расположенных в предыдущем слое. Активации, вычисленные каждым ядром, собираются в матрицы, которые называются картами признаков и представляют собой фактические выходные данные сверточных слоев. Последний слой СНС — это слой вывода фактического предсказания сети, он состоит из полностью связанных нейронов так, что каждый из них принимает в качестве входных данных все выходные данные предыдущих слоев. С учетом успеха проектирования архитектур СНС для классификации двумерных массивов классификация речевых эмоций следовала тенденции использования массивов спектральных величин, известных как речевые спектрограммы. Для решения проблемы РРЭ типичная СНС также предназначена для анализа речевых характеристик, которые представлены в виде многомерного массива [11].

В меньшей степени, чем AlexNet или GoogleNet, для приложений распознавания обычно используют более простые виды СНС, основанные на архитектурах типа LeNet-5 [14]. Выбор размера как традиционной (полносвязной) нейронной сети, так и СНС является сложной задачей. Например, для достижения приемлемой эффективности должны подстраиваться размеры весовых

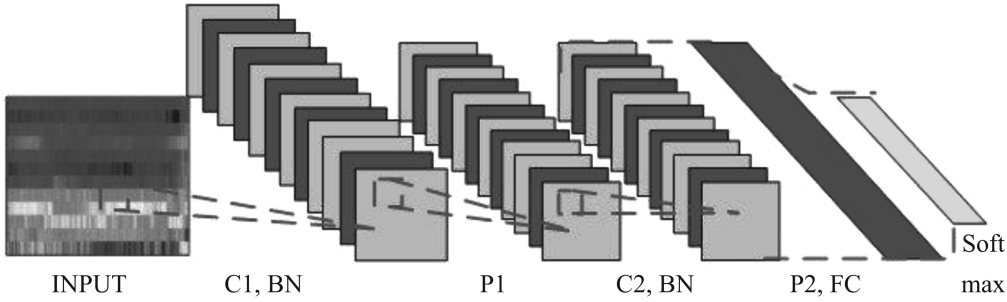


Рис. 1. Блок-диаграмма сверточной нейронной сети, предложенной Мишелем Валенти (Valenti-CNN).

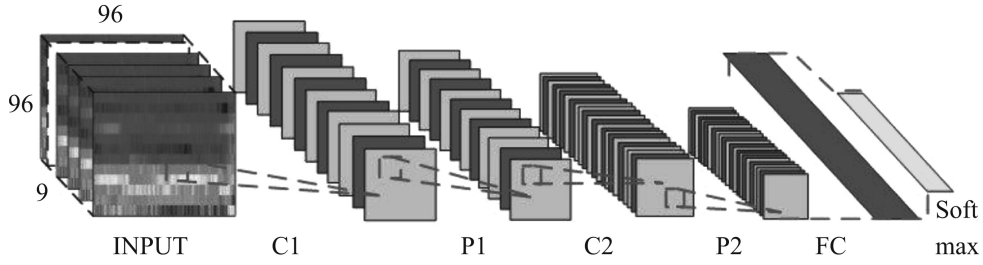


Рис. 2. Блок-диаграмма сверточной нейронной сети, предложенной Н. Хаджароласвади и Х. Демирелем (3D-CNN).

матриц в слоях. В общих случаях СНС выбираются эмпирически, в зависимости от характеристик обучающих данных, прошедших предварительную обработку.

В работе Мишеля Валенти [15] была предложена сеть CNN–Valenti-CNN (рис. 1). На вход сети подаются аудиопоследовательности в виде специально подготовленных логарифмических спектрограмм. Для этого применено кратковременное преобразование Фурье (STFT) с перекрытием окнами Хэмминга, далее абсолютные значения каждого полученного бина возведены в квадрат и применен мел-фильтр.

Еще одно интересное решение было предложено в работе Ноушини Хаджароласвади и Хасана Демиреля — 3D-CNN [16] (рис. 2). На вход сети там подается 88-мерный вектор, содержащий различные аудиохарактеристики в виде MFCC, частоты основного тона, интенсивности сигнала и т.д. Параллельно на вход подается частотный спектр каждого кадра.

Яфенг Ню и др. в [17] предложили оригинальную двумерную сверточную нейронную сеть (рис. 3), основанную на принципе визуализации сетчатки глаза и выпуклой линзы. На вход сети подаются спектрограммы разных размеров с эффектом, полученным при изменении фокусного расстояния. Таким образом достигалось увеличение числа тренировочных данных (аугментация)

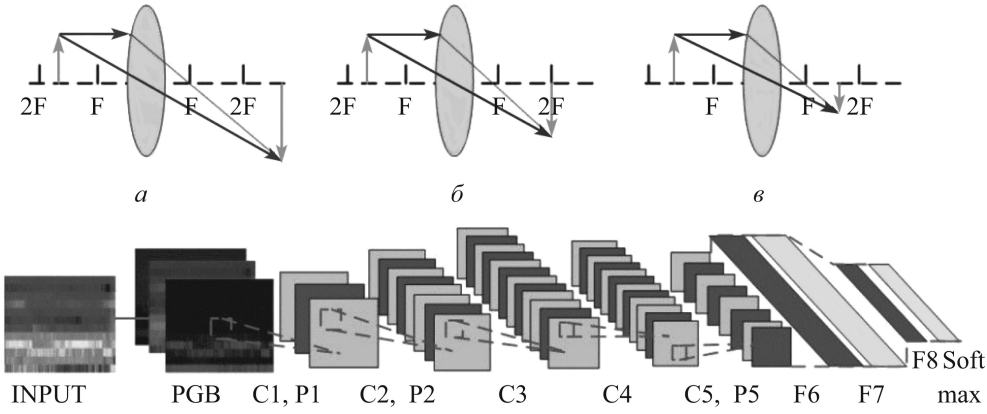


Рис. 3. Блок-диаграмма сверточной нейронной сети, основанной на принципе визуализации сетчатки глаза и выпуклой линзы.

путем изменения расстояния между спектрограммой и выпуклой линзой. Для этого были выбраны изображения в различных точках фокусирования, принадлежащие интервалам L1(F<L1<2F), L2(L2=2F) и L3(L3>2F).

В настоящее время СНС применяется к РРЭ многими исследователями, и в этом направлении уже достигнуты значительные результаты. Например:

1. Для сети, предложенной Яфенгом Ню и др., эксперименты проводились для речевых баз EmoDB [18] и SAVEE [19, 20]. Достигнута точность около 99% из семи видов эмоций.
2. Ч. Хуан и др. [21] обучили модель СНС, которая является стабильной и надежной в сложных сценах и превосходит некоторые хорошо зарекомендовавшие себя способы для решения задач РРЭ. Достигнуты результаты: точность 78% по базе SAVEE, 84% по базе Emo-DB.
3. С. Прасомфан [22] обнаружил эмоции, используя информацию внутри спектрограмм. Затем с помощью нейронной сети осуществил классификацию эмоции, используя базу EmoDB, и получил точность до 83,28% по пяти эмоциям.
4. Н. Семвал [23] предложила способ автоматического определения речевых эмоций с использованием многодоменных акустических моделей выбора и классификации. Этот подход был протестирован с базами EmoDB и BML (RED). Для мультиклассовой классификации достигается точность 80% для EmoDB и 73% для RED.

Однако для обучения глубокой нейронной сети требуется значительный объем данных, в то время как данные, предоставляемые существующими базами общих речевых эмоций, очень ограничены.

4. Одномерные СНС для решения задач РРЭ

Двумерные СНС были исследованы переходом к одномерным архитектурам, которые позволяют существенно снизить размерность обучающих при-

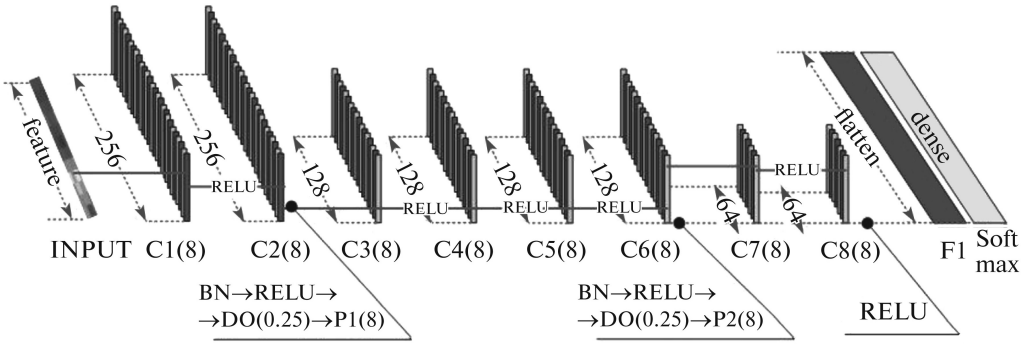


Рис. 4. Нейронная сеть (Reza 1-D CNN) для распознавания эмоций в речи, предложенная Реза Чу.

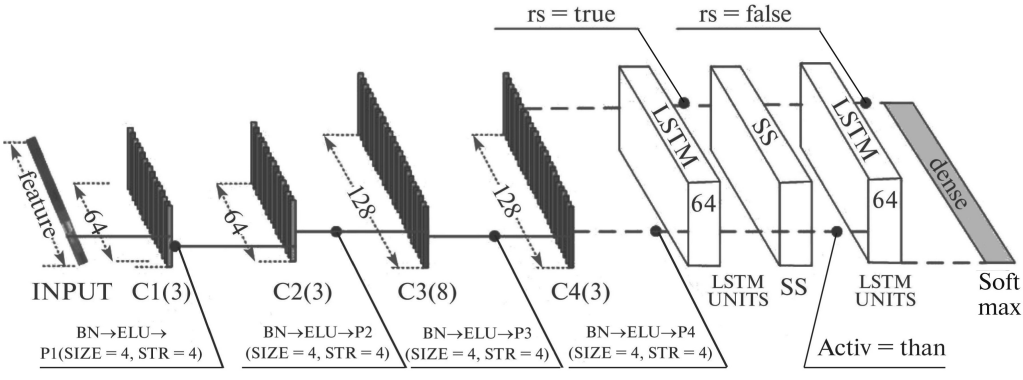


Рис. 5. Нейронная сеть (Vandana-Raian 1-D CNN-RNN) для распознавания эмоций в речи, предложенная В. Раджан.

знаков. Широкая популярность применения одномерных сверточных нейронных сетей для решения задач РРЭ возникла относительно недавно — начиная с 2019 г.

Так, Реза Чу в [24] предложил одномерную нейронную сеть — Reza 1-D CNN. Указанная нейронная сеть — это наиболее подходящая модель СНС для представления аудио-кортексиального органа слуховой системы человека в формальном описании (рис. 4, табл. 2 в справочной информации).

В это же время Ц. Чжао [25] предлагает СНС (реализация Vandana-Raian 1-D CNN-RNN [26]) с дополнительными рекуррентными слоями LSTM (рис. 5, табл. 3 в справочной информации). В отличие от модели Reza 1-D CNN в сети отсутствует регуляризация (dropout). Число сверточных ядер увеличивается в направлении выходного слоя с целью моделирования последовательностей. Полносвязный слой (Dense) получает выход из ячейки LSTM и рассчитывает логиты для каждого элемента выходной последовательности. Указанная нейронная сеть представляет собой гибридную архитектуру.

Можно заметить, что структура СНС для решения задач РРЭ имеет типичную архитектуру. Основное отличие заключается либо в расширении числа сверхточных ядер к полносвязному слою, либо к их уменьшению, а также отсутствием или наличием LSTM каскадов. В данной статье не рассматриваются параллельные архитектуры, так как такие сети нацелены на повышение точности классификации и используют иные акустические признаки дополнительно к MFFC. В настоящем исследовании допускается, что особенностей MFFC достаточно для того, чтобы решать задачу РРЭ.

5. Результаты экспериментальных расчетов

Для достижения поставленной цели была проведена собственная реализация рассмотренных выше архитектур нейронных сетей и проведено их обучение с наиболее популярными базами данных.

В первом эксперименте были обучены двумерные СНС для того, чтобы получить собственные оценки классификации. Во втором эксперименте был осуществлен переход к одномерным архитектурам. В третьем эксперименте проведены снижение размерности пространства обучающих признаков и сопоставление полученных результатов с предыдущим экспериментом.

Для тестирования двумерных СНС были выбраны следующие базы данных эмоций: Surrey Audio-Visual Expressed Emotion (SAVEE), Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) [27], Toronto emotional speech set (TESS) [28], Crowd-sourced Emotional Multimodal Actors Dataset (CREMA-D) [29] и Emo-DB.

Для каждого акустического образца из базы были извлечены мелкепстральные коэффициенты со следующими параметрами: длительность аудио 1–4 с, частота дискретизации 44 100 Гц, 64 MFCC коэффициента.

Архитектура нейронной сети, предлагаемая Яфенг Ню и др., была заменена на архитектуру сети LeNet-5 [30]. В эксперимент была добавлена одно-

Таблица 1. Результаты тестирования сверточных нейронных сетей

Accuracy (оценки абсолютной точности для MFCC)	Сверточная нейронная сеть			
	LeNet-5	Valenti-cnn	3D-CNN	1D-cochlea-cnn
Входной слой	64 x 774	64 x 774	64 x 774	64
Обуч. параметров	5,942,666	60,614,922	1,642,954	160,202
CREMAD	0,39	0,44	0,43	0,41
SAVEE	0,48	0,5	0,5	0,6
RAVDESS	0,43	0,39	0,54	0,42
TESS	0,99	0,99	0,99	0,99
EMO-DB	0,34	0,13	0,31	0,64
UNITED	0,67	0,71	0,74	0,68

мерная СНС (1D-cochlea-cnn), рассматриваемая в [31]. После обучения нейронных сетей были получены результаты, которые представлены в табл. 1. Числовые значения в таблице показывают абсолютную точность классификации каждой из СНС для соответствующей базы. Решения для баз CREMAD, SAVEE, RAVDESS, TESS и Emo-DB являются частными случаями, а мультилингвальное решение United — общим (объединенная база).

Результаты экспериментов с применением одномерных сверточных нейронных сетей показывают, что одномерные СНС для задач РРЭ не уступают двумерным аналогам. В [31] представлены эксперименты с одномерной сверточной сетью для задачи распознавания эмоционального состояния агрессии, где достигается точность в 75%.

В эксперименте каждый признак – это массив, состоящий из 49 536–131 072 элементов. В общем случае на вход двумерных СНС подаются матрицы размерностями 32, ..., 64 на 774, ..., 2048, ..., N. Для снижения размерности пространства признаков была принята гипотеза о том, что признак, задающий эмоцию в речи, сохраняется в случае усреднения мел-кепстральных коэффициентов по частотной шкале [31].

Для следующего эксперимента были выбраны две базы CREAMD и IEMOCAP, которые были объединены в единую базу. Из нее были отобраны восемь эмоций в следующих пропорциях по гендерному типу: male_happy (радость) — 671, male_angry (злость) — 671, male_sad (печаль) — 671, female_angry — 600, female_happy — 600, female_sad — 600, male_neutral — 575, female_neutral — 512.

После приведения двумерных признаков MFCC (2D-MFCC) к среднему вектору получены одномерные MFCC признаки (1D-MFCC). Длина каждого обучающего признака представляла собой массив размерностью 2048 элементов.

Данные для обучения выбранных одномерных сетей получились следующими:

— размер тренировочных признаков для обучения — 7042;

— набор тестовых признаков — 2347 (кросс-валидация);

— объем тренировочных признаков для каждой эпохи — 50.

На рис. 6 показаны одномерные MFCC-признаки для последующего машинного обучения. Полученные признаки не масштабированы по временной шкале.

После трехсот эпох обучения точность данных проверки для сети Reza-1-D-CNN варьируется в пределах 26%, а для сети Vandana-Raian-CNN-RNN — в пределах 24%. На графиках ошибки (рис. 7) заметно, что модель не способна хорошо сходиться даже с восемью целевыми классами. Однако для речевой базы RAVDESS Р. Чу декларирует, что для сети Reza-1-D-CNN достигает более 70% точности. Осуществляется это за счет упрощения модели в виде разбиения MFCC-признаков только на мужские или женские эмоции. Для сети Vandana-Raian-CNN-RNN и базы Emo-DB достигается результат

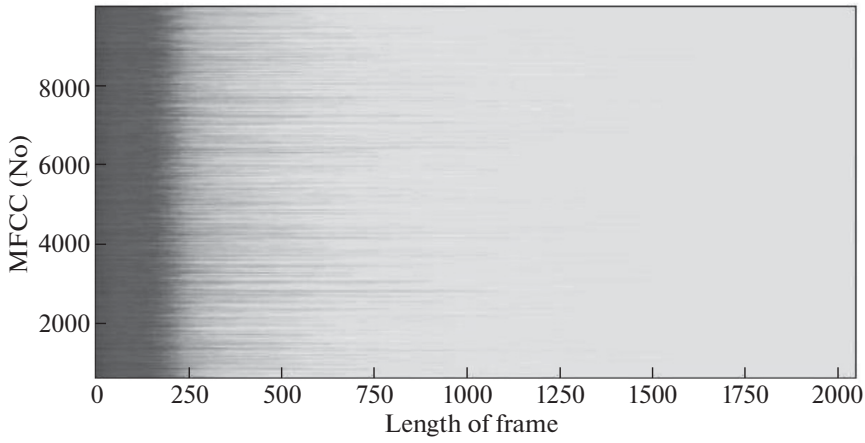


Рис. 6. Одномерные MFCC признаки на основе баз данных эмоций CREMAD и IMPOCAP.

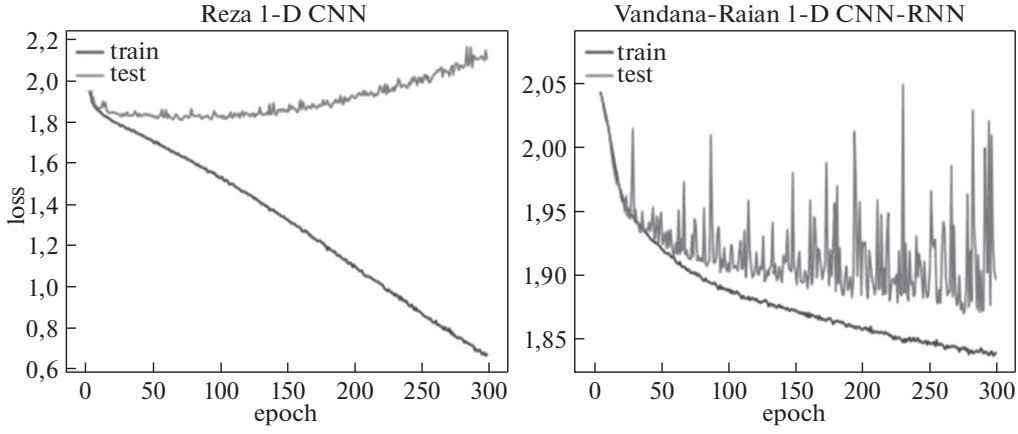


Рис. 7. Графики ошибки в процессе обучения моделей Reza-1-D-CNN и Vandana-Rajan-1-D-CNN-RNN с 1-D MFCC на основе баз данных эмоций CREMAD и IMPOCAP.

в 61%. Тем не менее для объединенных баз оценки классификации оставляют желать лучшего. Полученные результаты демонстрируют низкую эффективность классификации из-за усложнения структуры данных. Следует отметить, что в табл. 1 оценки для базы Emo-DB также невысоки.

В следующем эксперименте одномерные признаки MFCC были рассмотрены как временной ряд. Далее было применено преобразование Фурье к каждому из признаков. После преобразования были получены масштабированные признаки, представляющие собой массив из 64 элементов (рис. 8).

После повторного обучения точность данных проверки для сети Reza-1-D-CNN достигла 28%, а для сети Vandana-Raian-CNN-RNN — 27%. Графики

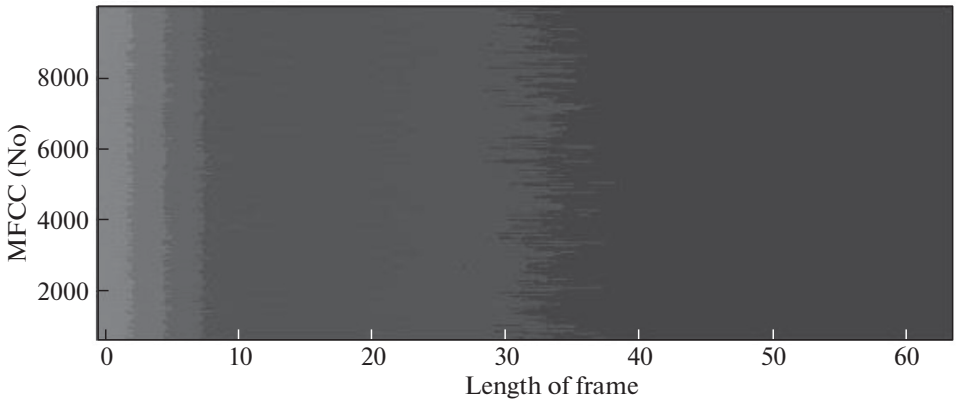


Рис. 8. 1-D-MFCC-FT признаки на основе баз данных эмоций CREMAD и IMPOCAP.

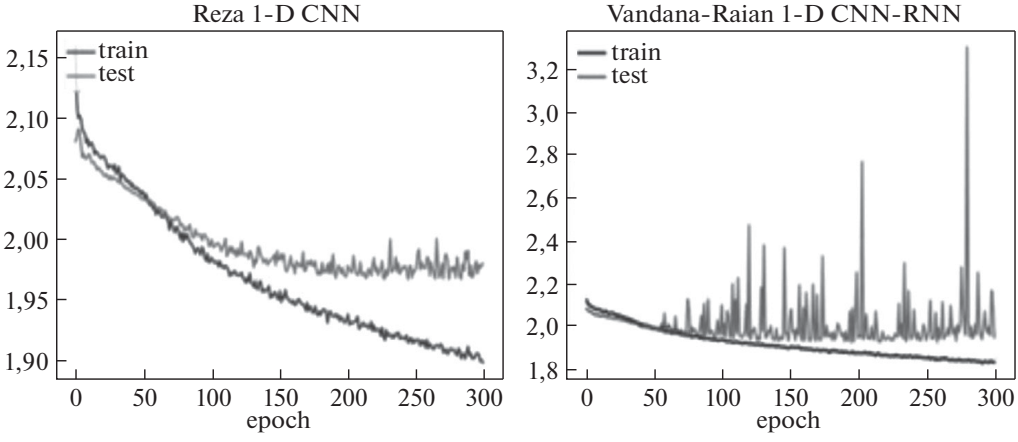


Рис. 9. Графики ошибки в процессе обучения моделей Reza 1-D CNN и Vandana-Rajan-1-D-CNN-RNN с 1-D MFCC-FT признаками, на основе баз данных эмоций CREMAD и IMPOCAP.

ошибок для сетей Reza-1-D-CNN и Vandana-Rajan-1-D-CNN-RNN с новыми признаками (1-D-MFCC-FT) показаны на рис. 9.

Из графиков видно, что оценки классификации согласуются с оценками предыдущего эксперимента. По сравнению со вторым экспериментом полученный способ позволяет снизить размерность обучающего признака в 32 раза.

Несмотря на то что расчет спектрограмм не полностью соответствует концепции сквозной сети, поскольку он допускает дополнительный этап предварительной обработки (преобразование 1D-MFCC в спектрограмму) перед моделью CHC, обработка минимальна, и наиболее важно, что сохраняется целостность сигнала. Предлагаемый подход к выделению признаков позволяет

значительно сократить длину обучающих признаков, обеспечивая простую трансформацию данных в новое пространство признаков. С практической точки зрения данный подход можно использовать для улучшения характеристик пространственного хранения или для вычислительной продуктивности алгоритмов обучения. Данный способ снижения размерности предлагается использовать в задачах РРЭ.

6. Заключение

Предложен подход приведения речевых данных, содержащих эмоциональную составляющую в речи, в форму, подходящую для алгоритмов машинного обучения. Очевидно, что качество и объем акустических признаков определяют, насколько хорошо алгоритмы машинного обучения способны обучаться. Следовательно, критически важно провести исследование и предварительную обработку признаков, прежде чем передавать их значения алгоритму обучения. Результаты эксперимента показывают, что небольшие сети, или сети, имеющие относительно малое число параметров, обладают недостаточной емкостью, а потому присутствует эффект недообученности, демонстрируется низкая эффективность, поскольку они не могут выявлять внутреннюю структуру сложных данных.

Предложенный авторами статьи подход для предобработки данных и выделения признаков способствует улучшению характеристик пространственного хранения и вычислительной продуктивности алгоритмов обучения. Полученные результаты важны для исследований, связанных с обработкой и анализом речевых сигналов, выделением определенных эмоциональных свойств говорящих [32]. Применение предложенного в статье метода в задачах анализа электронной информации позволит повысить эффективность работы за счет снижения вычислительной нагрузки, уменьшения пространства признаков и, соответственно, повышения скорости расчетов.

Справочная информация

1) Сокращения для конфигураций нейронных сетей:
Layer — слой;
LT — layer type (тип слоя);
SF — same filters (фильтры одного рода);
KS — kernel size (размер ядра свертки);
Strides — шаг свертки;
Activation — функция активации;
BN — batch Normalization (нормализация);
Dropout — регуляризация;
MP (P) — Max pooling (слой понижения размерности);
LSTM — Long short-term memory (слой с рекуррентной нейронной сетью);
AA — attention activation (слой активации рекуррентного слоя);

Flatten — полносвязный слой;

Dense — выходной полносвязный слой.

2) Конфигурации нейронных сетей (табл. 2, 3)

Таблица 2. Конфигурация одномерной нейронной сети — Reza-1-D-CNN

Layer	LT	SF	KS	Strides	Padding	BN	Activation	Dropout
1	CNN (SF)	256	8	1	same		ReLu	
2	CNN (SF)	256	8	1	same	+	ReLu	0,25
3	MP (P)		8	1				
4	CNN (SF)	128	8	1	same		ReLu	
5	CNN (SF)	128	8	1	same		ReLu	
6	CNN (SF)	128	8	1	same		ReLu	
7	CNN (SF)	128	8	1	same	+	ReLu	0,25
8	MP (P)		8	1				
9	CNN (SF)	64	8	1	same		ReLu	
10	CNN (SF)	64	8	1	same		ReLu	
11	flatten							
12	Dense						Softmax	

Таблица 3. Конфигурация одномерной нейронной сети — Vandana-Raian 1-D CNN-RNN

Layer	LT	SF	KS	Strides	Padding	BN	Activation	Dropout
1	CNN (SF)	64	3	1	same	+	elu	
2	MP (P)		4	4				
3	CNN (SF)	64	3	1	same	+	elu	
4	MP (P)		4	4				
5	CNN (SF)	128	3	1	same	+	elu	
6	MP (P)		4	4				
7	CNN (SF)	128	3	1	same	+	elu	
8	MP (P)		4	4				
9	LSTM	64						
10	AA						tanh	
11	LSTM	64						
12	Dense						Softmax	

СПИСОК ЛИТЕРАТУРЫ

1. *Мещеряков Р.В., Бондаренко В.П.* Диалог как основа построения речевых систем // Кибернетика и системный анализ. 2008. № 2. С. 30–41.

2. *Papakotas M., Siantikos G., Giannakopoulos T. et al.* IoT Applications with 5G Connectivity in Medical Tourism Sector Management: Third-Party Service Scenarios // GeNeDis 2016. Advances in Experimental Medicine and Biology. 2016. V. 989. P. 155–164. 2016. https://doi.org/10.1007/978-3-319-57348-9_12

3. *Okhapkin V., Okhapkina E., Iskhakova A. et al.* Application of neural network modeling in the task of destructive content detecting // CEUR workshop proceedings. Proceedings of the 3rd International Conference on R. Piotrowski's Readings in Language Engineering and Applied Linguistics, PRLEAL 2019. St. Petersburg, Russia, 2020. P. 85–94.

4. *Iskhakova A., Iskhakov A., Meshcheryakov R.* Research of the estimated emotional components for the content analysis // Journal of Physics: Conference Series. 2019. V. 1203. P. 1–10. https://doi.org/10.1088/1742-6596/1203/1/012065

5. *Scheirer E., Slaney M.* Construction and evaluation of a robust multifeature speech/music discriminator // IEEE International Conference on Acoustics, Speech, and Signal Processing. Munich, Germany, 2002. P. 1331–1334. https://doi.org/10.1109/ICASSP.1997.596192

6. *Hossan M.A., Memon S., Gregory M.A.* A novel approach for MFCC feature extraction // 2010 4th International Conference on Signal Processing and Communication Systems. Gold Coast, QLD, Australia, 2010. P. 1–5. https://doi.org/10.1109/ICSPCS.2010.5709752

7. *Logan B.* Mel Frequency Cepstral Coefficients for Music Modeling. https: //ismir2000.ismir.net/papers/logan_abs.pdf

8. *Rabiner L.R., Juang B.H.* Fundamental of Speech Recognition. USA: Prentice Hall, 1993.

9. *Nwe T.L., Foo S.W., Silva L.C.* Speech emotion recognition using hidden Markov models // Speech Communication. 2003. V. 41. No. 4. P. 603–623. https://doi.org/10.1016/S0167-6393(03)00099-2

10. *Zou D., Niu Y., He Z., Tan H.* A breakthrough in speech emotion recognition using deep retinal convolution neural networks. https: //arxiv.org/abs/1707.09917

11. *Lim W., Jang D., Lee T.* Speech Emotion Recognition using Convolutional and Recurrent Neural Networks // 2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA). Jeju, Korea (South), 2016. P. 1–4. https://doi.org/10.1109/APSIPA.2016.7820699

12. *Prasomphan S.* Improvement of speech emotion recognition with neural network classifier by using speech spectrogram // 2015 International Conference on Systems, Signals and Image Processing (IWSSIP). London, UK, 2015. P. 73–76. https://doi.org/10.1109/IWSSIP.2015.7314180

13. *Pakoci E., Popovic B., Pekar D.* Improvements in Serbian Speech Recognition using Sequence-Trained Deep Neural Networks // SPIIRAS Proceedings. 2018. Vol. 3(58). P. 53-76. https://doi.org/10.15622/sp.58.3

14. *Bengio Y., Hinton G.* Deep learning // Nature. 2015. V. 521. P. 436–444. https://doi.org/10.1038/nature14539.

15. *Valenti M., Squartini S., Diment A. et al.* A convolutional neural network approach for acoustic scene classification // 2017 International Joint Conference on Neural Networks (IJCNN). Anchorage, AK, 2017. P. 1547–1554. https://doi.org/10.1109/IJCNN.2017.7966035

16. *Hajarolasvadi N., Demirel H.* 3D CNN-Based Speech Emotion Recognition Using K-Means Clustering and Spectrograms // Entropy. 2019. V. 21(5) 479. P. 1–17. https://doi.org/10.3390/e21050479

17. *Niu Y., Zou D., Niu Y., He Z., Tan H.* A breakthrough in speech emotion recognition using deep retinal convolution neural networks. Preprint. https://arxiv.org/abs/1707.09917

18. *Burkhardt F., Paeschke A., Rolfes M., Sendlmeier W.F., Weiss B.* A Database of German Emotional Speech // INTERSPEECH 2005 — Eurospeech, 9th European Conference on Speech Communication and Technology. Lisabon, Portugal, 2005. P. 1–4. https://doi.org/10.21437/Interspeech.2005-446

19. *Haq S., Jackson P.J.B., Edge J.D.* Audio-Visual Feature Selection and Reduction for Emotion // Proceedings of the International Conference on Auditory-Visual Speech Processing 2008, Tangalooma Wild Dolphin Resort, Moreton Island, Queensland, Australia, 2008. P. 185–190.

20. *Haq S., Jackson P.J.B.* Speaker-Dependent Audio-Visual Emotion Recognition // Proceedings of the International Conference on Auditory-Visual Speech Processing, Norwich, UK, 2009. P. 53–58.

21. *Huang Z., Dong M., Mao Q., Zhan Y.* Speech Emotion Recognition Using CNN // MM '14: Proceedings of the 22nd ACM international conference on Multimedia. Orlando, Florida, USA, 2014. P. 801–804. https://doi.org/10.1145/2647868.2654984

22. *Prasomphan S.* Improvement of speech emotion recognition with neural network classifier by using speech spectrogram // 2015 IEEE International Conference on Systems, Signals and Image Processing. London, UK, 2015. P. 73–76. https://doi.org/10.1109/IWSSIP.2015.7314180

23. *Semwal N., Kumar A., Narayanan S.* Automatic speech emotion detection system using multi-domain acoustic feature selection and classification models // 2017 IEEE International Conference on Identity, Security and Behavior Analysis (ISBA). New Delhi, India, 2017. P. 1–6.

24. *Chu R.* Speech Emotion Recognition with Convolutional Neural Network. 2019. https://towardsdatascience.com/speech-emotion-recognition-with-convolution-neural-network-1e6bb7130ce3

25. *Jianfeng Z., Mao X., Chen L.* Speech emotion recognition using deep 1D & 2D CNN LSTM networks // Biomedical Signal Processing and Control. 2019. V. 47. P. 312–323. https://doi.org/10.1016/j.bspc.2018.08.035

26. *Rajan V.* 1D Speech Emotion Recognition. 2021. https: //github.com/vandana-rajan/1D-Speech-Emotion-Recognition

27. *Livingstone S.R., Russo F.A.* The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English // PLoS ONE. 2018. V. 13(5). P. 1–35. https://doi.org/10.1371/journal.pone.0196391

28. *Dupuis K., Pichora-Fuller M.K.* Toronto emotional speech set (TESS). https://doi.org/10.5683/SP2/E8H2MF https://dataverse.scholarsportal.info/dataset.xhtml?persistentId= doi:10.5683/SP2/E8H2MF

29. *Cao H., Cooper D.G., Keutmann M.K. et al.* CREMA-D: Crowd-sourced emotional multimodal actors dataset // IEEE transactions on affective computing. 2014. V. 5(4). P. 377–390. https://doi.org/10.1109/TAFFC.2014.2336244

30. *Franti E., Ispas I., Dragomir V. et al.* Voice Based Emotion Recognition with Convolutional Neural Networks for Companion Robots // Romanian Journal of Information Science and Technology. 2018. V. 20(3). P. 222–240.

31. *Iskhakova A., Wolf D., Meshcheryakov R.* Automated Destructive Behavior State Detection on the 1D CNN-Based Voice Analysis // Speech and Computer. SPECOM 2020. Lecture Notes in Computer Science. 2020. V. 12335. P. 184–193. https://doi.org/10.1007/978-3-030-60276-5_19

32. *Исхакова А.О., Вольф Д.А., Исхаков А.Ю.* Неинвазивный нейрокомпьютерный интерфейс для управления роботом // Высокопроизводительные вычислительные системы и технологии. 2021. Том 5. № 1. С. 166–171.

Статья представлена к публикации членом редколлегии О.П. Кузнецовым.

Поступила в редакцию 17.11.2021

После доработки 19.01.2022

Принята к публикации 26.01.2022

© 2022 г. А.И. ПАНОВ, канд. физ.-мат. наук (panov.ai@mipt.ru)
(Федеральный исследовательский центр
“Информатика и управление” РАН, Москва;
Московский физико-технический институт
(национальный исследовательский университет))

ОДНОВРЕМЕННОЕ ПЛАНИРОВАНИЕ И ОБУЧЕНИЕ В ИЕРАРХИЧЕСКОЙ СИСТЕМЕ УПРАВЛЕНИЯ КОГНИТИВНЫМ АГЕНТОМ[1]

Задачи планирования поведения и обучения принятию решений в динамической среде в системах управления интеллектуальными агентами обычно разделяют и рассматривают отдельно. Предложена новая объединенная иерархическая постановка задачи одновременно планирования и обучения (SLAP) в контексте предметного обучения с подкреплением и описана архитектура когнитивного агента, решающего данную задачу. Предложен новый алгоритм обучения действиям в частично наблюдаемой внешней среде с использованием подкрепляющего сигнала, предметного описания состояний внешней среды и динамически обновляемых планов действий. Рассмотрены основные свойства и преимущества предложенного алгоритма, среди которых — отсутствие фиксированного когнитивного цикла, вследствие которого ранее приходилось использовать разделение подсистем планирования и обучения, возможность строить и обновлять модель взаимодействия со средой, что повышает эффективность обучения. Предложено теоретическое обоснование некоторых положений данного подхода, предложен модельный пример и продемонстрирован принцип работы SLAP агента при управлении беспилотным автомобилем.

Ключевые слова: обучение с подкреплением, планирование поведения, когнитивный агент, иерархическое планирование, системы управления, беспилотный транспорт, мобильные роботы.

DOI: 10.31857/S0005231022060058, **EDN:** ACLEUU

1. Введение

Современные системы управления беспилотным транспортом и мобильными робототехническими платформами реализуют модульный подход к генерации автономного поведения [1]. Различные подсистемы отвечают за выполнение определенного рода подзадач: генерация траектории движения, реализация предложенной траектории с учетом динамики объекта управления, детекция и сегментирование объектов во внешней среде, планирование действий по манипуляции объектами, обучение модели взаимодействия со средой

[1] Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проект № 18-29-22027).

и т.д. При усложнении задач, которые ставятся перед объектом управления, увеличивается количество необходимых подсистем и усложняется их внутренняя организация, усложняется межмодульное взаимодействие.

Однако в последнее время в области разработки общих систем искусственного интеллекта наметилась обратная тенденция по объединению функциональности различных модулей в связи с тем, что для повышения эффективности и адаптивности решения перечисленных выше подзадач требуется комплексирование результатов или во многих случаях одновременная взаимосвязанная работа разных подсистем [2]. Примерами подобных ситуаций могут служить варианты интеграции подсистем компьютерного зрения в задаче управления беспилотным автомобилем, двигающимся в среде с большим количеством других автомобилей и пешеходов, когда для повышения эффективности предсказания траекторий других участников движения необходимо интегрировать в этот модуль работу подсистем сегментации и трекинга объектов [3].

Большое внимание в настоящее время уделяется применению методов машинного обучения в подсистемах, отвечающих как за непосредственное управление движением робототехнической платформы [4], так и за высокоуровневое планирование перемещения и поведения [5]. В данном случае основной задачей является уменьшение роли заранее заданных эвристик и вручную сформированных правил поведения на основе априорных знаний о задаче с целью повышения адаптивности методов и робастности получаемых решений при изменении условий внешней среды.

В настоящей статье предлагается новый подход по интеграции подсистем планирования поведения (т.е. действий как по перемещению, так и, например, манипуляции предметами внешней среды) и обучения поведению, в котором формируется адаптивная стратегия по достижению поставленной перед агентом цели [6]. Такая интеграция является естественной, так как обе подсистемы представляют собой различную реализацию модуля последовательного принятия решений [7]. Однако при планировании необходима модель функционирования внешней среды, а модуль обучения может автоматически формировать такую модель в явном или неявном виде. Для эффективного учета возможностей обеих подсистем предлагается использовать иерархическую организацию как для всей системы управления, так и для разделения высокоуровневого планировщика, для которого уже не требуется полной и точной модели, и низкоуровневой стратегии, которая обучается на основе оригинального метода обучения с подкреплением.

В данном исследовании предложена новая версия иерархической гибридной архитектуры STRL управления сложными техническими объектами [8] с целью выделения подсистем, обучающихся в процессе взаимодействия со средой. На стратегическом уровне управления впервые выделены три подсистемы — предметного представления модели среды, планирования поведения и обучения достижению подцелей. Основным вкладом данной статьи является новый подход к задаче интеграции подсистем планирования и обучения ко-

гнитивного агента, под которым подразумевается мобильная робототехническая платформа или беспилотное транспортное средство. Предлагается оригинальная иерархическая постановка задачи одновременного планирования и обучения (simultaneous planning and learning, SLAP). Во второй части статьи данный подход представлен в виде архитектуры SLAP агента, решающего поставленную задачу на основе иерархического подхода, в котором предлагается использовать планирование поведения по дереву Монте-Карло на верхнем уровне и предметное обучение с подкреплением для частично наблюдаемой среды на нижнем уровне иерархии действий. Представлены теоретическое обоснование для механизма обучения актора и критика в данной постановке. Кроме иллюстративного примера на клеточной среде, предложена схема реализации SLAP агента для задачи управления маневрами беспилотного автомобиля.

2. Архитектура управления поведением STRL2

В условиях динамической среды, свойства и поведение которой заранее не известны когнитивному агенту, в качестве которого будет пониматься мобильная робототехническая платформа, предлагается использовать обновленную версию архитектуры STRL, в которой сделан акцент на возможность обучения как в процессе выполнения действий в среде, так и на предобучение на заранее собранных наборах данных. На рис. 1 представлены схема основных подсистем архитектуры и базовые процедуры управления и передачи информации между модулями. Кратко рассмотрим основные особенности этой архитектуры.

Архитектура STRL2 является иерархической и состоит из трех базовых уровней. На среднем тактическом уровне, также как и в оригинальной версии, решаются задачи классического управления с использованием конкретной модели динамики объекта управления: задачи стабилизации, следования по траектории и т.п. Реактивный уровень осуществляет интегрирование уравнений модели динамики с учетом геометрических ограничений, которые накладываются при планировании траектории на верхнем уровне [9]. Результатом является выработка управляющего сигнала на органы управления.

На тактическом уровне в STRL2 предлагается уделить больше внимания задачам компьютерного зрения, а не только построению и прогнозированию траектории движения во внешней среде. На рис. 1 заполненными блоками отмечены подсистемы, требующие либо интерактивного обучения или предварительного обучения на заранее подготовленных наборах данных или в симуляторах. На тактическом уровне такими обучающимися подсистемами являются подсистемы нейросетевого картирования и локализации (SLAM), сегментации и трекинга объектов во внешней среде по RGB-D изображению или по данным с лазерных дальномеров (лидаров). Данные подсистемы формируют так называемую сенсорную ситуацию, по которой уже возможно построение специфических представлений (графов регулярной структу-

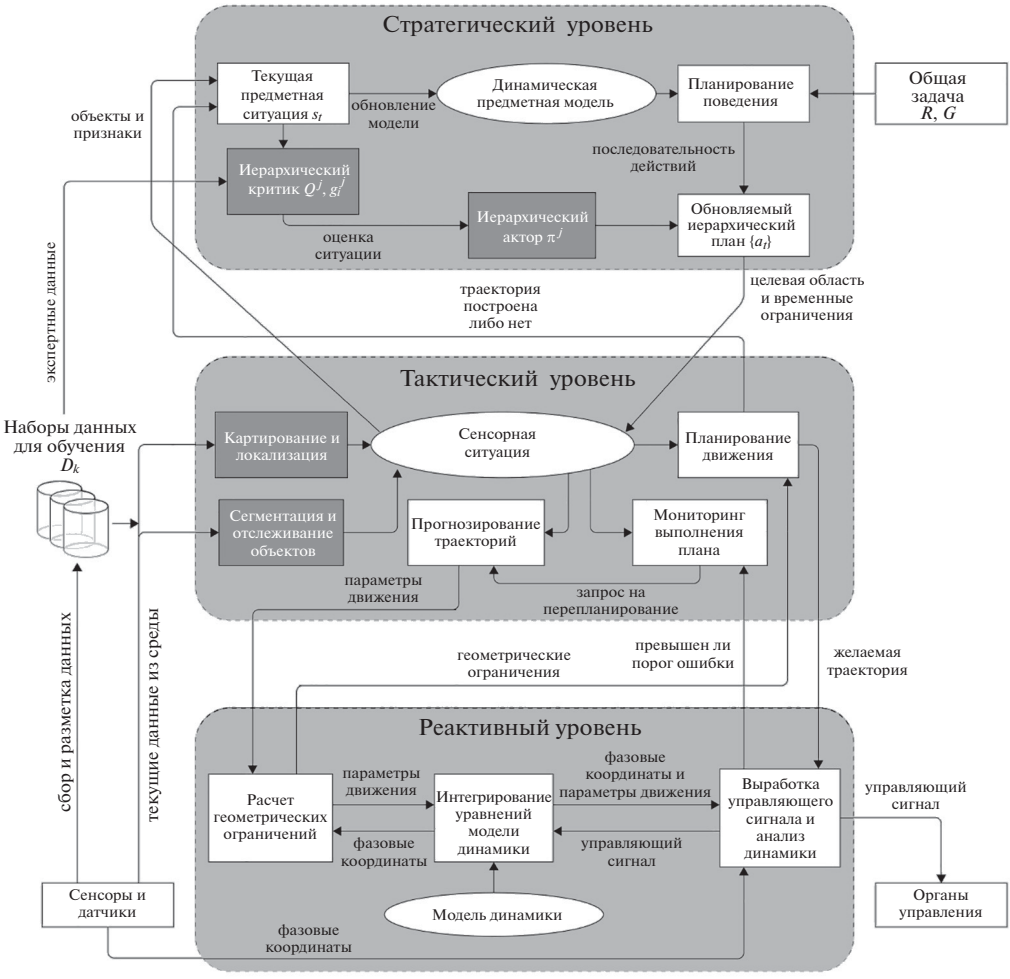


Рис. 1. Основные компоненты и межмодульное взаимодействие в архитектуре STRL2.

ры), необходимых для построения пути и мониторинга движения по траектории [10].

На стратегическом уровне STRL2 информация о текущей сенсорной ситуации используется для обновления долговременной предметной модели, по которой агент строит высокоуровневый план поведения. На этом уровне обучающимися подсистемами являются подсистема формирования оценки текущей ситуации относительно итоговой цели, поставленной перед агентом (модуль критика), и подсистемами актора, который автоматически формирует стратегию по достижению подцелей, выставленных планировщиком по модели. Зачастую критик и актор требуют предобучения в симуляционных средах с использованием заранее заданного сигнала вознаграждения, прежде чем они могут быть использованы для генерации поведения в реальной среде.

В предложенной архитектуре STRL2 делается акцент на формирование адаптивного поведения когнитивного агента в заранее неизвестной динамической среде без необходимости координировать свои действия с другими участниками общей деятельности, в то время как в первой версии архитектуры большее внимание уделялось именно многоагентной составляющей и распределению ролей в коалиции [11].

В подразделе 2.1 будет дана полная постановка задачи работы агента на стратегическом уровне, на котором предлагается объединить в единый цикл работу подсистем планирования и обучения.

Опишем формальную постановку задачи одновременного планирования и обучения с использованием предметно-ориентированной концепции иерархического обучения с подкреплением. Вначале напомним понятие частично наблюдаемого марковского процесса [12], формально представляющего процесс взаимодействия агента и среды, затем расширим его на иерархический случай, введем понятие предметной ситуации и, наконец, объединим это с формальным определением плана действий агента.

2.1. Частично наблюдаемый марковский процесс принятия решений

Итак, пусть $< S, O, A, T, R, G, \Omega >$ — частично наблюдаемый марковский процесс принятия решений (POMDP), где:

- $S = \{s_1, \ldots, s_n\}$ — конечное множество состояний внешней среды, в которой действует агент,
- $O = \{o_1, \ldots, o_k\}$ — конечное множество наблюдений агента, включающих в себя описание объектов (предметов), выделяемых из состояний среды (предполагается, что наблюдение содержит лишь некоторую часть информации о состоянии, т.е. $k < n$),
- $A = \{a_1, \ldots, a_m\}$ — конечное множество действий, в том числе и составных,
- $T : S \times A \to \Pi(S)$ — функция переходов, определяющая по текущему состоянию и действию распределение вероятностей на состояниях среды в следующий момент времени (здесь и далее будем обозначать через $\Pi(X)$ множество вероятностных распределений на конечном множестве X),
- $R : S \times A \to \mathbb{R}$ — функция вознаграждений,
- $G : S \to \{0, 1\}$ — целевая функция, определяющая момент остановки эпизода взаимодействия,
- $\Omega : S \times A \to \Pi(O)$ — функция наблюдений, определяющая распределение вероятностей для наблюдений в текущем состоянии.

В общей постановке задачи предполагается, что все функции в определении частично наблюдаемого марковского процесса принятия решений (POMDP) T, R, G, Ω агенту неизвестны и он может лишь оценивать их в результате взаимодействия со средой, выполняя некоторое действие $a_t \in A$ и получая из среды некоторое наблюдение $o_{t+1} \in O$ и вознаграждение $r_{t+1} = R(s_t, a_t)$. Таким образом, агенту заранее известно только множество A, множество O он может восстановить в явном виде в процессе взаимодействия со средой, а множество S он может оценить только по наблюдениям.

Целью агента является построение такой функции $\pi : O \to \Pi(A)$, задающей вероятностное распределение на множестве действий A при условии текущего наблюдения $o \in O$, при котором максимизируется ожидаемое суммарное вознаграждение (отдача):

$$\mathbb{E}_\pi \left[\sum_{t:G(s_t)\neq 1} \gamma^t R(s_t, a_t) \right] \to \max_\pi, \tag{1}$$

где γ – дисконтирующий множитель. Здесь предполагается, что суммирование идет до тех пор, пока целевая функция $G(s_t)$ не примет значение единица. Подсчет математического ожидания по стратегии подразумевает усреднение по траекториям в пространстве состояний, по которым считается отдача. Ожидаемую отдачу можно подсчитать и для каждого состояния (функция полезности $V(s)$), и для пары состояние-действие (функция $Q(s, a)$).

Функция $\pi(o|s)$ называется стратегией агента и служит для определения последовательности действий агента по заданной последовательности состояний среды. В постановке задачи безмодельного обучения с подкреплением в частично наблюдаемой среде предполагается реактивное поведение агента, при котором агент не прогнозирует реакцию среды в каждый момент времени t и генерирует новое действие $\mathbb{E}_\pi \sum_{t=0}^{\tau} \gamma^t R(s_t, a_t)$ в предположении, что вся существенная информация для принятия решения содержится в состоянии $\mathbb{E}_\pi \sum_{t=0}^{\tau} \gamma^t R(s_t, a_t)$, которое он определяет на основе текущего наблюдения o_t. Вероятность пребывания в следующем состоянии среды $b_{t+1}(s_{t+1})$ (предполагаемое состояние) определяется агентом по текущему наблюдению o_t в соответствии с выражением:

$$b_{t+1}(s_{t+1}|o_{t+1}) = \frac{\Omega(o_{t+1}|s_{t+1}, a_t) \sum_{s_t} T(s_{t+1}|s_t, a_t) b_t(s_t)}{\sum_{s_{t+1}} \left(\Omega(o_{t+1}|s_{t+1}, a_t) \sum_{s_t} T(s_{t+1}|s_t, a_t) b_t(s_t) \right)}. \tag{2}$$

Используя понятие предполагаемого состояния b_t, можно ввести обычный марковский процесс принятия решений на непрерывном множестве таких состояний. В этом случае функции полезности будут определяться для предполагаемых состояний b_t:

$$V^\pi(b_t) = \mathbb{E}_\pi \sum_t \gamma^t \sum_s b_t(s) R(s, a_t). \tag{3}$$

2.2. Иерархическая постановка и параметризация

Введем иерархию на множестве действий, следуя концепции полумарковского процесса принятия решений и подхода умений [13]. Введем так называемое длящееся во времени действие, или умение, $\kappa =< I_\kappa, \pi_\kappa, \beta_\kappa >$, где $I_\kappa \subseteq O$ – инициирующее множество наблюдений, π_κ – стратегия, реализующая данное

умение, $\beta_\kappa : O \to \{0,1\}$ – терминальная функция, останавливающая реализацию умения. Множество умений будем обозначать через κ. Расширение множества действий за счет множества умений приводит к определению полумарковского процесса принятия решений и введению функций полезности состояния $V_\kappa(b)$ и умения $Q_\kappa(b,\kappa)$.

В иерархической постановке агент должен сформировать как стратегию π_κ на множестве умений (высокоуровневая стратегия) внутренние стратегии для каждого умения π_κ (низкоуровневые стратегии) и функции остановки для каждого умения β_κ. Не снижая общности постановки задачи, можно считать, что инициирующие множества для всех умений включают все возможные наблюдения $I_\kappa = O$. Будем параметризовать стратегию π_κ и функцию остановки β_κ с помощью наборов параметров θ и ϑ соответственно. В иерархической постановке цель агента – максимизировать отдачу, начиная с предполагаемого состояния b_0 и умения κ_0, — запишется в виде

$$\mathbb{E}_{\kappa,\theta,\kappa}\left[\sum_{t:G(s_t)\neq 1}\gamma^t\sum_s b_t(s)R(s,a_t)\,\middle|\,b_0,\kappa_0\right]\to\max_{\theta,\vartheta}. \tag{4}$$

Определим полезность выполнения конкретного действия a в рамках умения κ в предполагаемом состоянии $b(s)$ (Q_a-функция) и полезность самого умения κ в $b(s)$ (Q_κ-функция). Полезность умения определяется его внутренней стратегией и полезностью каждого действия: $Q_\kappa(b,\kappa) = \sum_a \pi_{\kappa,\theta}(a|o)Q_a(b,\kappa,a)$, где полезность действия в свою очередь записывается через функцию переходов среды (здесь и далее через штрих будем обозначать следующий момент времени):

$$\begin{aligned} Q_a(b,\kappa,a) &= \sum_s b(s|o)R(s,a) + \\ &+\gamma\sum_{o'}\left(\sum_{s'}\Omega(o'|s',a)\sum_s T(s'|s,a)b(s|o)\right)\tilde{Q}_\kappa(b'(s'|o),\kappa). \end{aligned} \tag{5}$$

В определении полезности действия Q_a используется поправка к полезности умения $\tilde{Q}_\kappa$, которая учитывает возможность остановки умения при следующем наблюдении o':

$$\tilde{Q}_\kappa(b',\kappa) = \big(1-\beta_{\kappa,\vartheta}(o')\big)\,Q_\kappa(b',\kappa) + \beta_{\kappa,\vartheta}(o')V_\kappa(b'). \tag{6}$$

2.3. Предметная ситуация

Наблюдение, получаемое агентом, практически во всех значимых окружениях представляет собой некоторую сцену, состоящую из объектов или предметов. Декомпозиция предметной сцены на отдельные взаимосвязанные составляющие может оказаться полезной в том случае, когда такие взаимосвязи отделимы от самих предметов, а действия агента могут быть отнесены

не ко всей сцене, а к концертному целевому предмету. Формально такая декомпозиция для марковского процесса принятия решений описывается в так называемой объектно-ориентированной поставке [14, 15]. В постановке задачи одновременного обучения и планирования будет рассматриваться случай, когда взаимосвязи объектов несущественны для принятия решений агентом и принципиально только наличие тех или иных предметов в сцене.

Итак, пусть и состояние среды s, и наблюдение агента o представляют собой некоторое множество независимых объектов, которые относятся к конечному числу классов $C = \{c_1, c_2, \ldots, c_k\}$. Каждый класс характеризуется своим набором атрибутов или признаков $\{f_1^c, f_2^c, \ldots, f_{n_c}^c\}$, а объект $e \in E$ класса $c(e) \in C$ описывается конкретными значениями данных признаков $e = \{d_1^c, d_1^c, \ldots, d_{n_c}^c\}$. Как состояние s, так и наблюдение агента o, таким образом, представляет собой объединение состояний объектов $\bigcup_{i=1}^{n} e_i$. Будем считать, что частичная наблюдаемость выражается в том, что состояние s и наблюдение o отличаются друг от друга набором объектов и (или) значениями характеризующих их признаков. Выделение объектов по наблюдению происходит с помощью некоторой функции $\Phi^p : O \to 2^E$, которая будет считаться заранее заданной.

В предположении независимого присутствия предметов в среде в текущем наблюдении возможна декомпозиция функции переходов и функции вознаграждений по отдельным классам объектов: $T = \{T_{c_i} | c_i \in C\}$, $R = \{R_{c_i} | c_i \in \in C\}$. Низкоуровневая стратегия агента будет состоять из действий, условиями выполнения для которых будет служить наличие объекта определенного класса, т.е. множество действий также разбивается на подмножества в соответствии с количеством классов $A = \{A_{c_i} | c_i \in C\}$. Проведя такую декомпозицию задачи, возможно определить и выписать соотношение на полезность не всего состояния или наблюдения, а на полезность конкретного объекта, имеющегося в текущем наблюдении. Все соотношения на функции полезности, определенные в подразделе 2.4, остаются в силе с той лишь поправкой, что в текущем наблюдении агент выбирает действие жадно, в соответствии с наибольшей полезностью конкретного объекта $Q_a(b, \kappa, a) = = \arg\max_{e \in o} Q_a(e, \kappa, a_{c(e)})$. Будем считать, что умения агента не поддаются аналогичной декомпозиции и зависят от всего наблюдения целиком.

2.4. Обновление плана поведения

В постановке задачи одновременного обучения и планирования агент автоматически строит обновляемую модель среды $M =< \hat{T}, \hat{R}, \hat{\Omega} >$, которая позволяет находить план B достижения цели $G(s_l) = 1$ за счет моделирования переходов при некоторой модельной стратегии π:

$$B^\pi =< o_0, r_0, a_0, o_1, r_1, a_1, \ldots, a_{l-1}, o_l >,$$

где $s_i = \hat{T}(s_{i-1}, a_{i-1})$, $r_i = \hat{R}(s_i, a_i)$, $a_i \sim \pi(a|o_i)$, а заключительное наблюдение o_l соответствует целевому состоянию s_l согласно функции $\hat{\Omega}$. Здесь

под $\hat{T}, \hat{\Omega}$ и $M =< \hat{T}, \hat{R} >$ будем понимать приближенные (аппроксимируемые) значения функций переходов, наблюдений и вознаграждений соответственно. План поведения агента, таким образом, составляется жадным образом в точности до небольшой поправки, отвечающей за исследование среды, в предположении корректности текущего приближения модели $M =< \hat{T}, \hat{R} >$.

Под процессом обучения агента будет пониматься итерационное обновление модели $M =< \hat{T}, \hat{R} >$, функций полезности Q, стратегии π и соответственно плана поведения B^{π}. Возможны четыре основных варианта составления общей схемы обучения агента с использованием фазы планирования по модели. Приведем краткие алгоритмические схемы для этих вариантов. Будем обозначать собранный агентом опыт в виде множества прецедентов $D = \{(o_t, r_t, a_t)_{t=1}^{T}\}$. Траекторией будем называть некоторую последовательность таких прецедентов в порядке их формирования при взаимодействии со средой. Первый вариант интеграции представляет собой обучение с использованием планируемых ("воображаемых") траекторий [16]:

1. Агент предсказывает ("воображает") траектории с некоторого состояния s_t, используя модель $M = < \hat{T}, \hat{R} >$.
2. Агент обновляет свою стратегию, используя прецеденты из предсказываемых траекторий.
3. Агент набирает новый опыт взаимодействия со средой по обновленной стратегии.
4. По собранному опыту D агент обновляет модель среды $M = < \hat{T}, \hat{R} >$.
5. Шаги 1–4 повторяются до сходимости.

Второй вариант — обучение с использованием разделения стратегий — подразумевает использование модели только на начальной стадии взаимодействия со средой, постепенно расширяя горизонт ее применения в процессе уточнения:

1. Агент планирует и выполняет первые шаги в траектории, используя модель $M = < \hat{T}, \hat{R} >$.
2. Агент использует текущую стратегию для продолжения траекторий в процессе взаимодействия со средой.
3. По собранному опыту агент обновляет модель и стратегию, одновременно выбирая критерий остановки планирования и запуска интерактивной стратегии.

Обучение с имитацией эпизодов возможно только в случае наличия копий среды, в которых агент имитирует свое поведение:

1. Используем возможность запускать симуляции действий агента в среде, чтобы с текущего шага проиграть некоторое количество эпизодов для обновления этой модели.
2. На основе обновленной модели определяем полезность состояний и выбираем действие, выполняемое в среде.
3. Обновляем стратегию, используя выбранное действие в качестве эталонного.

В предлагаемой в данной статье постановке задачи одновременного обучения и планирования используется иерархия для разделения применения модели и интерактивной стратегии:

1. Агент на верхнем уровне иерархии действия использует модель $M = = < \hat{T}, \hat{R} >$ для получения плана на множестве умений.
2. Стратегия каждого умения формируется в интерактивном режиме в процессе взаимодействия со средой.
3. Набранный опыт используется агентом для одновременного уточнения модели на умениях и стратегий каждого умения.

Принимая во внимание все приведенные уточнения постановки задачи одновременного обучения и планирования, получается иерархическая постановка, в которой агенту необходимо максимизировать получаемую в рамках эпизода отдачу с возможностью декомпозиции наблюдения по предметному принципу, автоматическому построению стратегий умений и при одновременном автоматическом формировании модели среды, используемой для планирования на множестве умений.

3. Архитектура SLAP агента

Для решения поставленной в разделе 2 общей задачи одновременного обучения и планирования в данной статье предлагается использовать следующую архитектуру интеллектуального SLAP агента (см. рис. 2). Подсистему обучения агента разделим на две составляющие, как это принято в теории обучения с подкреплением. Критик обновляет функцию полезности действия Q_a, а актор формирует стратегию π_κ в рамках текущего умения. Цикл взаимодействия SLAP агента со средой будет выглядеть следующем образом:

1. Агент использует текущую модель для того, чтобы сформировать план $B^\pi = < o_0, r_0, a_0, o_1, r_1, a_1, \ldots, a_{l-1}, o_l >$ на множестве умений с помощью процедуры SLAPplan. В общем случае предполагается, что уровней иерархии действий может быть несколько (вложенные умения) и может быть сформировано несколько вложенных планов: от высокоуровневого до низкоуровневого. Далее считаем, что план B^π является низкоуровневым.
2. В соответствии с планом агент B^π выбирает текущее умение κ_t.
3. Агент получает из среды текущее наблюдение, которое с помощью функции Φ^p переводится в набор предметов $o_t \to \{e_1, \ldots, e_n\}$.
4. В соответствии со стратегией π_κ для умения κ_t агент выбирает текущее действие a_t.
5. Агент выполняет действие a_t в среде и получает новые наблюдения и вознаграждение.
6. С помощью процедуры SLAPlearn агент проводит оценку выполненного действия и самого умения с помощью критика, а затем обновляет функции аппроксимации критика и актора.
7. Если текущее умение завершилось, выполняется перепланирование, и затем происходит переход к шагу 3.

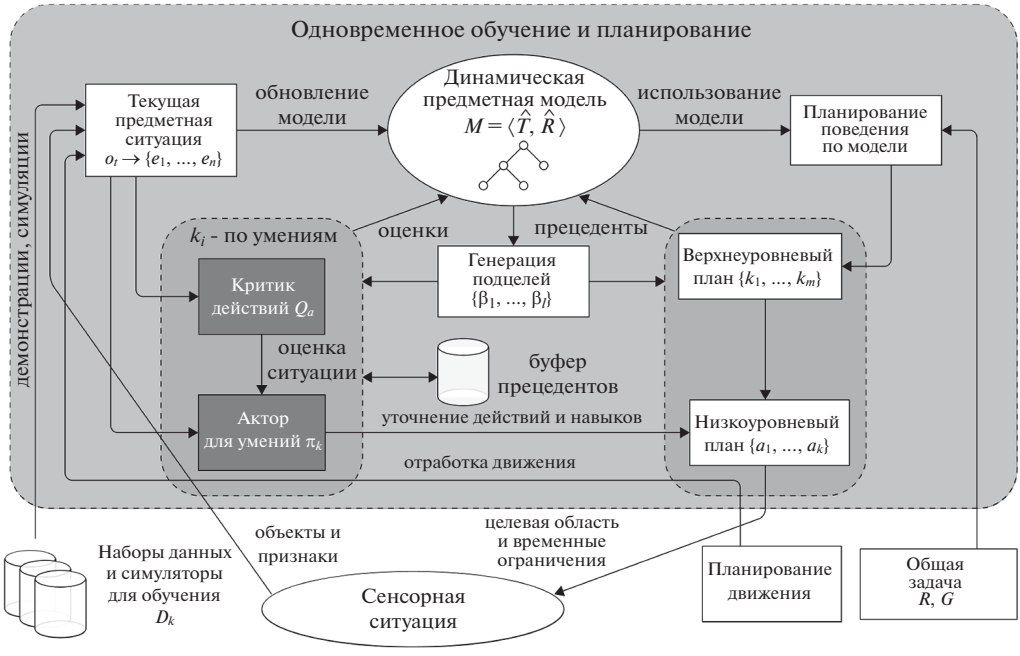


Рис. 2. Архитектура SLAP агента с подсистемами обучения (выделены темным цветом) и планирования поведения. Данная схема представляет собой верхний уровень архитектуры STRL, которая изображена на рис. 1, с детализацией взаимодействия методов обучения и планирования.

Кратко изложим принципы работы процедур SLAPplan и SLAPlearn. При планировании агент использует модель $M = < \hat{T}, \hat{R}, \hat{\Omega} >$ для построения плана своего поведения. В настоящей статье предлагается использовать реализацию модели в виде расширенного дерева поиска Монте-Карло, где каждый узел дерева отвечает за конкретный объект, выделяемый из наблюдения. Ребро дерева соответствует выбору некоторого объекта из наблюдения, выполнению некоторого действия (умения) и переходу к наблюдению, где выделяется следующий объект. В случае, когда для выполнения выбирается действие (умение), для которого в модели неизвестно следующее наблюдение, образуются новые узлы с соответствующими объектами, выделенными из наблюдения, полученного из среды.

Планирование $SLAPplan(M, e)$ в данном дереве $M = < \hat{T}, \hat{R}, \hat{\Omega} >$ происходит за счет поиска кратчайшего пути с учетом дополнительного веса для действий, направленных на исследование среды:

1. Выбираем планируемое умение $\kappa \leftarrow \arg\max_{\kappa} Q_\kappa(e) + \eta\sqrt{\frac{\log N(e)}{N(e,\kappa)}}$. Здесь второе слагаемое отвечает за верхнюю доверительную границу (UTC) эффективной стратегии исследования среды, η – константа, N – счетчики.
2. Производим переход по дереву $e' \leftarrow \hat{T}_c(\kappa)$, $r \leftarrow \hat{R}_c(\kappa)$, где c – класс объекта e.

3. Производим планирование для следующего объекта с подсчетом получаемого вознаграждения $\tilde{R} \leftarrow r + \gamma SLAPplan(M, e')$.
4. Обновляются счетчики $N(e, \kappa) \leftarrow N(e, \kappa) + 1$, $N(e) \leftarrow N(e) + 1$.
5. Настраивается модель — обновляем полезность умения $Q_\kappa(b, \kappa) \leftarrow$ $\leftarrow Q_\kappa(b, \kappa) + \frac{\tilde{R} - Q_\kappa(b,\kappa)}{N(e,\kappa)}$, где b – предполагаемое состояние, для которого выделяется объект e.

В процедуре обучения критика и актора SLAPlearn производится обновление как критерия достижения подцели β_κ, так и стратегии π_κ конкретного, выбранного на верхнем уровне иерархии умения. Напомним, что здесь используется параметризация с помощью набора параметров ϑ для условия завершения и набора θ — для стратегии:

1. Агент выбирает действие в соответствии с текущей стратегией умения $a \sim \pi_{\kappa,\theta}(a|o)$.
2. Агент выполняет действие a, наблюдает o' и r.
3. Критик обновляет оценку полезности действия в рамках текущего умения:

$$Q_a(b, \kappa, a) \leftarrow Q_a(b, \kappa, a) +$$
$$+ \alpha \left(r + \gamma \left((1 - \beta_{\kappa,\vartheta}(o'))Q_\kappa(b', \kappa) + \beta_{\kappa,\vartheta}(o') \max_{\kappa'} Q_\kappa(b', \kappa') \right) - Q_a(b, \kappa, a) \right),$$

   где α – шаг обучения критика, g – обновляемое целевое значение для критика.
4. Актор обновляет параметры для стратегии и для подцели

$$\theta \leftarrow \theta + \alpha_\theta \nabla_\theta \log \pi_{\kappa,\theta}(a|o) Q_a(b, \kappa, a),$$
$$\vartheta \leftarrow \vartheta + \alpha_\vartheta \nabla_\vartheta \tilde{Q}_\kappa(b', \kappa),$$

   где α_θ и α_ϑ – шаги обучения.
5. Обновляем значение градиента полезности умения $\nabla_\theta Q_\kappa$, который может быть использован на верхнем уровне иерархии.
6. Если в соответствии с $\beta_{\kappa,\vartheta}$ подцель достигнута, то в соответствии с высокоуровневым планом выбирается новое умение κ.

Здесь предполагается, что полезность текущего умения передается из подсистемы планирования, где обновление полезности происходит во время обновления самой модели. В разделе 5 дан вывод выражений для градиента $\nabla_\theta Q_\kappa$ (теорема о градиенте критика) и для градиента $\nabla_\vartheta \tilde{Q}_\kappa$ генератора подцелей (теорема о градиенте актора).

4. Модельный пример

В качестве модельного примера рассмотрим задачу обучения навигации до некоторой заданной целевой точки в клеточной среде с препятствиями и управляемыми объектами (дверьми) (см. рис. 3,*a*). Среда организована в виде комнат с узкими проходами между ними таким образом, чтобы поддерживать формирование двухуровневой иерархии действий. Верхний уровень:

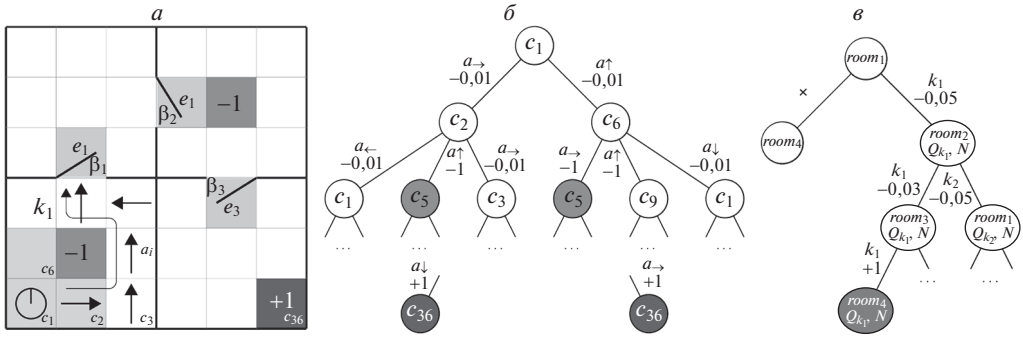


Рис. 3. *а* – Модельный пример перемещения агента по клеточной среде с комнатами, открывающимися проходами между ними и опасными состояниями, в которых завершается эпизод взаимодействия; *б* – соответствующая марковскому процессу принятия решения модель, которая строится агентом без иерархической декомпозиции; *в* – модель клеточной среды, которую строит агент с умениями.

умения по переходу от одной комнаты к другой, нижний уровень: действия по перемещению в четыре стороны для достижения выхода из комнаты или достижения целевой клетки в комнате. Опишем среду подробнее.

В данном примере наблюдение агента o_t — это область видимости вокруг агента радиусом в две клетки (на рис. 3 обозначены серым цветом). Множество действий агента A состоит из четырех действий: поворот на 90° по часовой или против часовой стрелки, проход прямо, открытие двери. На верхнем уровне иерархии агенту доступны умения κ_i, состоящие из действий a_i. Подцелями β_i, в которых завершаются умения, являются клетки, обозначенные светло-серым цветом. Выделяемыми с помощью функции Φ^p объектами e_i в среде являются сами комнаты, двери каждой комнаты, принадлежащие классу объектов c_0, и отдельные клетки, в которых в данный момент находится агент. При этом каждой комнате и клетке соответствует свой класс c_i. Функция вознаграждения R реализована следующим образом: каждый момент времени агенту дает небольшое отрицательное вознаграждение $(-0,01)$, за попадание в темные клетки — (-1) с завершением эпизода, за достижение цели (темная клетка справа внизу) дается $+1$. В качестве выбираемой параметризации используется линейная модель, взвешивающая признаки, соответствующие предметам, выделяемым по наблюдению.

Предметная модель, которую строит и обновляет агент, представлена на рис. 3,*в*. Для сравнения на рис. 3,*б* показана модель, которую агент аппроксимировал бы без иерархического представления действий. Процедуры планирования $SLAPplan(M, e)$ и обучения SLAPlearn для агента для представленного пример будут выглядеть следующим образом.

SLAPplan:

1. Выбираем планируемое умение κ по переходу из текущей комнаты e в соседнюю e' в соответствии с предсказываемой по дереву полезностью данного умения с учетом параметра исследования среды.

2. По текущему виду дерева (так как модель обновляется в процессе обучения, она может не соответствовать итоговой модели на рис. 3,*в*) производится переход в соседнюю комнату e' по стратегии умения π_κ с подсчетом вознаграждения в виде суммы вознаграждений после каждого действия стратегии π_κ.
3. Рекурсивно запускается аналогичная процедура для новой комнаты e'.
4. Обновляются счетчики $N(e,\kappa) \leftarrow N(e,\kappa)+1$, $N(e) \leftarrow N(e)+1$.
5. Настраивается модель — обновляется полезность умения $Q_\kappa(b,\kappa)$.

SLAPLearn:

1. Агент выбирает действие по переходу в соседнюю клетку или открытию двери в соответствии с текущей стратегией умения $a \sim \pi_{\kappa,\theta}(a|o)$, учитывая, что сейчас агент находится в определенной комнате и выполняется текущее умение κ.
2. Агент выполняет действие a, обновляется наблюдение — в область видимости агента попадают новые предметы (клетки, дверь).
3. Критик обновляет оценку полезности действия в рамках текущего умения, вычисляя TD ошибку.
4. Актор обновляет параметры для стратегии и для подцели, вычисляя градиент.
5. Если в соответствии с $\beta_{\kappa,\vartheta}$ подцель достигнута, то в соответствии с высокоуровневым планом выбирается новое умение κ.

5. Теоремы о градиенте

В разделе 2 была представлена общая схема взаимодействия SLAP агента со средой, где для обучения агента необходимо знание целевого значения критика g и для градиента функции полезности стратегии ∇J. Проведем краткие выкладки для вычисления их значений.

Выражение $\sum_{s'} \Omega(o|s',a) \sum_s T(s'|s,a) b(s|o)$ представляет собой вероятность получения агентом следующего наблюдения o', что будем обозначать как $p(o|a,b)$. В начале найдем выражение для градиента полезности умения $Q_\kappa(b,\kappa)$ относительно параметров реализующей его стратегии θ:

$$
\begin{aligned}
(7)\quad \nabla_\theta Q_\kappa(b,\kappa) &= \nabla_\theta \sum_a \pi_{\kappa,\theta}(a|o) Q_a(b,\kappa,a) = \\
&= \sum_a \left(\nabla_\theta \pi_{\kappa,\theta}(a|o)\right) Q_a(b,\kappa,a) + \\
&+ \sum_a \pi_{\kappa,\theta}(a|o) \nabla_\theta \left(\sum_s b(s|o) R(s,a) + \gamma \sum_o p(o|a,b) \tilde{Q}_\kappa(b'(s'|o),\kappa) \right) = \\
&= \sum_a \left(\nabla_\theta \pi_{\kappa,\theta}(a|o)\right) Q_a(b,\kappa,a) + \sum_a \pi_{\kappa,\theta}(a|o) \sum_{o'} \gamma p(o|a,b) \nabla_\theta \tilde{Q}_\kappa(b'(s'|o),\kappa).
\end{aligned}
$$

Используя выражение в определении поправленной полезности $\tilde{Q}_\kappa$, найдем ее градиент:

$$\nabla_\theta \tilde{Q}_\kappa(b',\kappa) = (1-\beta_{\kappa,\vartheta}(o')) + \sum_{\kappa'} \left(\beta_{\kappa,\vartheta}(o')\pi(\kappa'|b')\right) \nabla_\theta Q_\kappa(b',\kappa'). \tag{8}$$

Подставляя это в выражение для градиента полезности умения, получим:

$$\begin{aligned}\nabla_\theta Q_\kappa(b,\kappa) &= \sum_a \left(\nabla_\theta \pi_{\kappa,\theta}(a|o)\right) Q_a(b,\kappa,a) + \\ &+ \sum_a \pi_{\kappa,\theta}(a|o) \sum_{o'} \gamma p(o|a,b) \nabla_\theta \tilde{Q}_\kappa(b'(s'|o),\kappa) = \\ &= \sum_a \left(\nabla_\theta \pi_{\kappa,\theta}(a|o)\right) Q_a(b,\kappa,a) + \sum_{o'}\sum_{\kappa'} p(o',\kappa'|b,\kappa) \nabla_\theta Q_\kappa(b',\kappa'),\end{aligned} \tag{9}$$

где $p(o',\kappa'|b,\kappa)$ задает расширенный марковский процесс принятия решений, в котором состояниям соответствует пара наблюдение-умение. Учитывая марковское свойство при раскрытии рекурсии в определении $\nabla_\theta Q_\kappa$, получается доказательство теоремы 1.

Теорема 1 (о градиенте критика умений). *Для фиксированного множества марковских умений со стохастической реализующей стратегией, дифференцируемой по параметрам θ, градиент ожидаемой дисконтированной отдачи по параметрам θ с начальными условиями (b_0,κ_0) равен*

$$\nabla_\theta Q_\kappa(b,\kappa) = \sum_{b,\kappa} \mu_\kappa(b,\kappa|b_0,\kappa_0) \sum_a \left(\nabla_\theta \pi_{\kappa,\theta}(a|o)\right) Q_a(b,\kappa,a), \tag{10}$$

где $\mu_\kappa(b,\kappa|b_0,\kappa_0)$ – дисконтированные частоты появления предполагаемого состояния и умения по траекториям, начинающимся с начальных условий (b_0,κ_0).

В разделе 4 для обновления параметров генератора подцелей было указано на необходимость вычисления градиента $\nabla_\vartheta \tilde{Q}_\kappa$. Используем определение для этой функции полезности:

$$\begin{aligned}\nabla_\vartheta \tilde{Q}_\kappa &= \nabla_\vartheta \beta_{\kappa,\vartheta}(o')(V_\kappa(b') - Q_\kappa(b',\kappa)) + \\ &+ (1-\beta_{\kappa,\vartheta}(o')) \sum_a \pi_{\kappa,\theta}(a|o') \sum_{o''} \gamma p(o''|a,b') \nabla_\vartheta \tilde{Q}_\kappa(b'',\kappa).\end{aligned}$$

Здесь также замечаем наличие рекурсии, и использование структуры расширенного марковского процесса принятия решений приводит к теореме 2.

Теорема 2 (о градиенте генератора подцелей). *Для фиксированного множества марковских умений со стохастической реализующей стратегией, дифференцируемой по параметрам ϑ, градиент ожидаемой дисконтированной отдачи по параметрам ϑ с начальными условиями (b_1,κ_0) равен*

$$\nabla_\vartheta \tilde{Q}_\kappa(b,\kappa) = \sum_{b',\kappa} \mu_\kappa(b',\kappa|b_1,\kappa_0) \nabla_\vartheta \beta_{\kappa,\vartheta}(o')(V_\kappa(b') - Q_\kappa(b',\kappa)), \tag{11}$$

где $\mu_\kappa(b', \kappa | b_1, \kappa_0)$ *– дисконтированные частоты появления предполагаемого состояния и умения по траекториям, начинающимся с начальных условий* (b_1, κ_0).

Таким образом, сформулированы две теоремы, которые позволяют получить выражения для обновления набора параметров в процедуре SLAPLearn.

6. Возможности применения подхода SLAP в управлении беспилотным транспортным средством

Предложенная архитектура SLAP агента может служить теоретическим обоснованием реализации адаптивной системы управления беспилотным автомобилем. На рис. 4 представлена схема интеграции SLAP агента в широко распространенную в индустрии систему управления беспилотными автомобилями Apollo. В данном случае предполагается, что стандартный модуль планирования будет заменен модулем, который помимо планирования позволяет сохранять опыт, дообучаться и использовать настроенные стратегии для улучшения и ускорения этапа планирования. Таким образом, наряду с рядом подсистем, которые используются в Apollo и в STRL (локализации, построение карты и т.д.), за адаптивное планирование поведения здесь отвечает SLAP агент. В системе Apollo интеграция всех этих модулей осуществляется на основе модифицированной робототехнической операционной системы (Apollo Cyber RT и системы реального времени RTOS), которая позволяет обмениваться сообщениями в асинхронном режиме различным подсистемам.

Рассмотрим задачу обгона автомобилем динамических препятствий (других автомобилей) на многополосном шоссе (рис. 5). Модуль построения карты выдает информацию о границах дороги, полосах и разрешенных скоростях. Модуль локализации позволяет агенту определить свое положение и скорости на шоссе. Модуль построения сенсорной модели передает информацию о движущихся объекта и их скоростях. Модуль предсказания траектории выдает предполагаемые траектории всех объектов на сцене с некоторым горизонтом планирования. В модуле одновременного планирования и обучения реализуется схема по иерархическому адаптивному планированию. На верхнем уровне составляется абстрактный план по действиям, которые должен совершить агент (перестроение направо, перестроение налево). Данный план

Рис. 4. Схема использования SLAP агента в качестве модуля в системе управления Apollo.

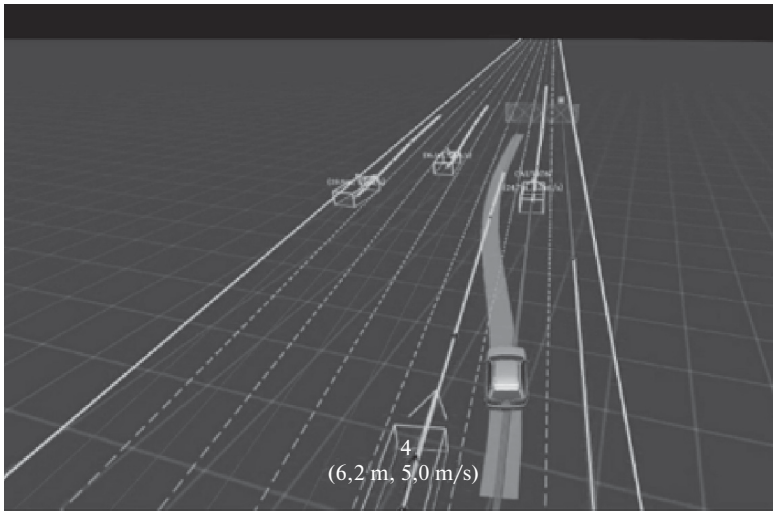


Рис. 5. Пример реализации сценария обгона динамических препятствий в системе управления Apollo с использованием предложенной концепции адаптивного планирования. Светлые тонкие линии — предсказываемые траектории объектов, светлая полоса — планируемая траектория агента.

строится путем поиска кратчайшего на графе поведенческого дерева (вычислительный аналог дерева Монте-Карло в Apollo), в котором реализованы основные правила и условиях обгона (см. подробности реализации в [17]).

Каждый шаг высокоуровневого плана уточняется в процессе обучения (реализации построенных планов) в симуляторе, и каждая часть общего маневра обгона настраивается для субоптимальной реализации соответствующей части общей траектории с использованием обучения с подкреплением или с его "мягкой версией" — эволюционным программированием. Наконец, сглаживание построенных траекторий с учетом динамики объекта управления происходит в модуле управления движением.

Интеграция в общую схему управления автомобилем, реализуемую в Apollo, подсистемы SLAP позволяет добиться большей адаптивности получаемых решений и позволяет автоматизировать процесс задания условий для совершения безопасного маневра.

7. Заключение

В статье представлен новый подход к задаче интеграции подсистем планирования и обучения в иерархических системах управления мобильными робототехническими платформами и беспилотными транспортными средствами. Была предложена оригинальная иерархическая постановка задачи одновременного планирования и обучения, в которой предлагается провести двойную декомпозицию задачи. Первая декомпозиция реализуется за счет выделения абстрактных действий — умений и обучаемых стратегий, которые реализу-

ют их на операционном уровне. Вторая декомпозиция касается выделения предметной среды в ситуации и упрощении модели, используемой для обучения актора, формирующего стратегию, и критика, оценивающего полезность ситуаций. В статье предложены два примера, в которых продемонстрированы особенности работы архитектуры SLAP агента, решающего поставленную задачу: модельная задача с комнатами и важная индустриальная задача планирования маневров беспилотного автомобиля. Предложенные принципы обучения SLAP агента теоретически обоснованы.

В дальнейшем предполагается развитие предложенного подхода в двух направлениях. Первое направление предполагает теоретическое исследований сложностных свойств предложенных алгоритмов, критериев сходимости процесса обучения и нижних гарантируемых оценок качества получаемых стратегий агента. Второе направление предполагает продолжение проработки принципов обновления и использования объектной модели сенсорной ситуации для более эффективной работы процедур планирования.

СПИСОК ЛИТЕРАТУРЫ

1. *Trafton G.J., et al.* ACT-R/E: An Embodied Cognitive Architecture for Human-Robot Interaction // J. Human-Robot Interaction. 2013. V. 2. No. 1. P. 30–54.
2. *Goertzel B.* From Abstract Agents Models to Real-World AGI Architectures: Bridging the Gap // Lecture Notes in Computer Science / ed. Everitt T., Goertzel B., Potapov A. Cham: Springer International Publishing, 2017. V. 10414. P. 3–12.
3. *Wu J., et al.* Track to Detect and Segment: An Online Multi-Object Tracker // 2021 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR). IEEE, 2021. P. 12347–12356.
4. *Likhachev M., Ferguson D.* Planning long dynamically feasible maneuvers for autonomous vehicles // Int. J. Robotics Research. 2009. V. 28. No. 8. P. 933–945.
5. *Aitygulov E., Kiselev G., Panov A.I.* Task and Spatial Planning by the Cognitive Agent with Human-like Knowledge Representation // Interactive Collaborative Robotics. ICR 2018. Lecture Notes in Computer Science / ed. Ronzhin A., Rigoll G., Meshcheryakov R. Springer, 2018. V. 11097. P. 1–12.
6. *Саттон Р.С., Барто Э.Г.* Обучение с подкреплением. М.: БИНОМ. Лаборатория знаний, 2011. Изд. 2-е.
7. *Moerland T.M., Broekens J., Jonker C.M.* Model-based Reinforcement Learning: A Survey. 2020. P. 421–429.
8. *Макаров Д.А., Панов А.И., Яковлев К.С.* Архитектура многоуровневой интеллектуальной системы управления беспилотными летательными аппаратами // Искусственный интеллект и принятие решений. 2015. № 3. С. 18–33.
9. *Yakovlev K., et al.* Combining Safe Interval Path Planning and Constrained Path Following Control: Preliminary Results // Interactive Collaborative Robotics. ICR 2019. Lecture Notes in Computer Science. 2019. V. 11659. P. 310–319.
10. *Staroverov A., et al.* Real-Time Object Navigation with Deep Neural Networks and Hierarchical Reinforcement Learning // IEEE Access. 2020. V. 8. P. 195608–195621.
11. *Киселев Г.А.* Интеллектуальная система планирования поведения коалиции робототехнических агентов с STRL архитектурой // Информационные технологии и вычислительные системы. 2020. № 2. С. 21–37.

12. *Pack L., Littman M.L., Cassandra A.R.* Planning and acting in partially observable stochastic domains // Artificial Intelligence. 1998. V. 101. P. 99–134.
13. *Bacon P.-L., Harb J., Precup D.* The Option-Critic Architecture // Proc. of the AAAI Conf. on Artificial Intelligence. 2017. V. 31.
14. *Keramati R., et al.* Strategic Object Oriented Reinforcement Learning. 2018.
15. *Watters N., et al.* COBRA: Data-Efficient Model-Based RL through Unsupervised Object Discovery and Curiosity-Driven Exploration. 2019.
16. *Hafner D., et al.* Dream to Control: Learning Behaviors by Latent Imagination // Int. Conf. on Learning Representations. 2020.
17. *Jamal M., Panov A.* Adaptive Maneuver Planning for Autonomous Vehicles Using Behavior Tree on Apollo Platform // Artificial Intelligence XXXVIII. SGAI 2021. Lecture Notes in Computer Science / ed. Bramer M., Ellis R. 2021. V. 13101. P. 327–340.

Статья представлена к публикации членом редколлегии О.П. Кузнецовым.

Поступила в редакцию 31.10.2021

После доработки 09.01.2022

Принята к публикации 26.01.2022

© 2022 г. Т.В. АФАНАСЬЕВА, д-р техн. наук (afanaseva.tv@rea.ru)
(Российский экономический университет им. Г.В. Плеханова, Москва)

ГРАНУЛЯЦИЯ МНОГОМЕРНЫХ ВРЕМЕННЫХ РЯДОВ В ЗАДАЧЕ ДЕСКРИПТИВНОГО АНАЛИЗА СОСТОЯНИЯ И ПОВЕДЕНИЯ СЛОЖНЫХ ОБЪЕКТОВ

Многомерные временные ряды, являясь источником скрытых знаний, могут выступать моделями восприятия объектов во многих прикладных областях. Статья посвящена разработке концептуальных положений гранулярных вычислений многомерных временных рядов, на основе которых предложена методика дескриптивного анализа, позволяющая получать информационные гранулы о состоянии и поведении объекта наблюдения, выраженные в текстовой форме с использованием протоформ. Рассмотрено применение грануляции многомерного временного ряда в дескриптивном анализе развития экономики РФ.

Ключевые слова: многомерный временной ряд, грануляция, протоформа, дескриптивный анализ.

DOI: 10.31857/S000523102206006X, **EDN:** ACSAXP

1. Введение

Одним из направлений в развитии прикладных интеллектуальных систем являются моделирование и реализация когнитивного процесса представления информации об объектах окружающего мира, согласованной с экспертными представлениями. Понятие грануляции возникло как естественная потребность в обобщенном представлении информации, ориентированном на человека, для поддержки процессов понимания данных и преобразования информации в семантически значимые сущности. Информационная грануляция и связанный с этим понятием термин “информационная гранула” были введены Лофти Заде [1] как некоторая общая концепция представления и структурирования информации, характеризующая сложные объекты. Под гранулой в смысле Л. Заде [2] понимается группа концептуально значимых сущностей, объединяемых отношениями неразличимости, эквивалентности, сходства, близости, для определения которой требуется задать обобщенные ограничения. Исходная посылка теории грануляции Л. Заде заключалась в описании связи восприятия человеком объектов окружающего мира и их представлениями в терминах естественного языка в виде информационных гранул (ИГ). Принимая во внимание, что для представления терминов характерна неточность, обусловленная ограниченностью языка, восприятием пространства, времени, движения и когнитивными возможностями человека,

Л. Заде предложил математическую конструкцию для описания лингвистически значимых ИГ в виде пропозиций (коротких предложений), формальную основу которых образуют протоформы [2–4]. Как абстракции данных ИГ не только отражают природу данных, но и могут эффективно фиксировать дополнительные знания предметной области, сообщаемые пользователем [5]. Так, понятие гранулярной мета-онтологии введено в публикации [6], и предложен новый подход к представлению знаний о жизненном цикле сложной технической системы, опирающийся на онтологическое моделирование и теорию грануляции информации. Идея грануляции в представлении знаний о свойствах объектов является ключевой во многих прикладных областях, в частности, в таких как медицина, электронная коммерция, транспорт, управление, в исследованиях по кибербезопасности, в обработке и анализе больших данных, при решении задач сентимент-анализа [7–13]. Развитие направления представления знаний о свойствах объектов в виде ИГ привело к грануляции темпоральной информации, представленной в виде одномерных временных рядов [14–21]. Учитывая сложную структуру временного ряда, исследователи предложили использовать грануляцию для представления знаний о его поведении в лингвистической форме [22–25]. Опираясь на концепцию нечеткой грануляции и операцию обобщения на основе лингвистического резюмирования, был предложен подход к грануляции одномерного временного ряда на основе протоформ с нечеткими квантификаторами [26–30]. Авторы в публикации [31] определяют иерархический язык темпоральных правил для выражения сложных шаблонов, представленных в многомерных временных рядах. Семиотическая иерархия временных паттернов, которые не заданы априори, строится из семиотических троек: уникальный символ, грамматическое правило и определяемая пользователем метка, при этом необходим эксперт, чтобы интерпретировать правило.

Отмечая актуальность и достижения в области гранулярных вычислений, отметим, что решение задачи грануляции объектов, информация о которых представлена в виде многомерных временных рядов, не нашло достаточного отражения в научных публикациях. В то же время многомерные временные ряды (МВР) являются объектом анализа в прикладных задачах, решаемых в рамках различных классов систем: телекоммуникационных, финансовых, образовательных, транспортных, производственных, медицинских, коммерческих, социальных, экологических. Приведем основные положения теории грануляции информации, значимые для дескриптивной аналитики сложных объектов.

1. Информационные гранулы (ИГ) обладают свойствами компактности, структурности, иерархичности, ограниченности, лингвистической интерпретируемости [2]. Выделяют четкие интервальные (c-гранулы), нечеткие гранулы (f-гранулы) и гранулы в виде протоформ (p-гранулы), последние служат основой создания пропозиций, выражающих знания о свойствах объектов в форме предложений естественного языка [3, 24, 29].

2. В основе гранулярных вычислений (Granular Computing) лежит процесс автоматического создания ИГ, последующая обработка которых требует меньше времени, что важно в контексте больших данных [3–5, 18, 27].
3. Процесс грануляции информации основан на моделировании когнитивных операций абстрагирования и обобщения в процессах восприятия и представления свойств объектов [2, 5, 31], что является фундаментальным для интеллектуальных систем.

Поэтому цель данной статьи — разработка концептуальных положений грануляции МВР для извлечения информационных гранул, характеризующих состояние и поведение объекта исследования в лингвистической форме.

2. Задача грануляции МВР в контексте дескриптивного анализа объектов

Определим объект дескриптивного анализа O в виде совокупности элементов $G = \{g_i, i = 1, 2, \ldots, gk\}$, описываемых множеством показателей $M = \{m_j,\ j = 1, 2, \ldots, mk\}$, числовые значения которых изменяются на временном интервале $T = [1, tk]$. Тогда модель восприятия объекта O определим в виде многомерного временного ряда:

$$X = \{x_{ijt}, x_{ijt} \subseteq \mathbb{R},\ i = 1, 2, \ldots, gk;\ j = 1, 2, \ldots, mk;\ t = 1, 2, \ldots, tk\}\,. \tag{1}$$

В настоящей статье задача дескриптивного анализа объекта заключается в том, чтобы получить знания о состоянии и поведении объекта исследования в виде ИГ. Учитывая, что назначение ИГ — представлять знания, извлеченные из данных, согласованные с экспертными знаниями, на различных уровнях абстракции с использованием лингвистических терминов [3, 5], введем два класса ИГ: expert-defined гранулы и data-extracted гранулы.

Expert-defined гранулы (e-гранулы) рассматриваются в виде ИГ, основанных на экспертных знаниях о специфике моделей восприятия данных, и определяются типом решаемой задачи. Формальную основу expert-defined гранул могут составлять множества, интервалы, нечеткие множества, грубые множества, лингвистические переменные, правила и функции агрегирования [7, 8]. При этом в качестве e-гранул могут применяться c-гранулы и f-гранулы, которые определяют метод грануляции МВР и точку зрения лица, принимающего решения. Результат грануляции МВР будем рассматривать в виде data-extracted гранул (d-гранул), которые сжато представляют свойства объекта, распределенные по его показателям и элементам. Чтобы представить эти свойства в контексте состояния и поведения объекта в лингвистической форме, целесообразно использовать пропозиции, формально задаваемые в виде протоформ вида [4, 28]:

$$m \text{ is } Z, \tag{2}$$

$$Qx's \text{ are } Z, \tag{3}$$

где m представляет показатель объекта (например, энергопотребление), x обозначает некоторую сущность объекта (например, подсистема), Z определяет data-extracted гранулу, описывающую состояние или поведение объекта (например, эффективное или стабильное), Q обозначает data-extracted гранулу, в виде квантификатора (например, большинство), обобщающего сущности с одинаковыми Z. В формулах (2) и (3) глаголы “is” (является) и “are” (являются) определяют отношение принадлежности data-extracted гранулы Z к объектам левой части этих формул. Текстовое выражение этих глаголов зависит от контекста описания свойств объектов, а их примеры приведены в табл. 3.

Тогда постановку задачи грануляции МВР рамках дескриптивного анализа объекта O сформулируем в следующим виде: имея многомерный временной ряд X и набор expert-defined гранул E, заданных на $W \supseteq X$, требуется определить множество data-extracted гранул D, т.е. построить отображение

$$F : X \times E \to D. \tag{4}$$

3. Концептуальная модель expert-defined гранул

В рамках поставленной задачи анализируемые свойства рассматриваются как некоторые качественные характеристики, которые резюмируют состояние объекта O и его поведение по МВР в лингвистических терминах. Исходя из этого для описания состояния объекта могут использоваться лингвистические термины $Y \in Sy$, описывающие качественные уровни значений X, например, из множества $Sy = \{$“низкий”, “средний”, “высокий”$\}$, а для описания поведения — лингвистические термины $B \in Sb$, характеризующие тенденции изменения качественных уровней на временном интервале $T = [1, tk]$, значения которых содержатся в множестве $Sb = \{$“рост”, “стабильность”, “падение”, “колебание”$\}$. Заметим, что каждый термин из множества Sy обобщает некоторый интервал значений на X, ассоциированный с показателем $m \in M$ объекта в заданный момент времени $t \in T$, в то время как термин из множества Sb обобщает последовательность значений показателя $m \in M$ на заданном временном интервале T.

С каждым лингвистическим термином из множеств Sy и Sb согласно теории ИГ [2] необходимо сопоставить обобщенное ограничение, заданное на X, в виде математического описания, например, с использованием понятия интервала, множества, последовательности, функции, группы, нечеткого или грубого множества [3]. Тогда термины $Y \in Sy$ и $B \in Sb$ вместе с их обобщенными ограничениями rY и rB могут быть использованы для гранулярного представления состояния и поведения объекта O. Например, для термина B могут быть использованы ограничения:

$$B = \begin{cases} \text{рост}, & \text{если} \quad a > c_{\max}, \\ \text{стабильность}, & \text{если} \quad a \in [c_{\min}, c_{\max}], \\ \text{падение}, & \text{если} \quad a < c_{\min}, \end{cases}$$

где a определяет оценку коэффициента в уравнении регрессии $x_t = at + b$; интервал $[c_{\min}, c_{\max}]$ включает значения a, имеющие малую вариабельность относительно некоторой константы.

Чтобы агрегировать множество похожих ИГ, целесообразно использовать протоформы с квантификаторами частотности [26, 30], в которых квантификатор $Q \in Sq$ будет резюмировать множество лингвистически эквивалентных ИГ с использованием лингвистических терминов, например, из множества $Sq = \{$“все”, “большинство”, “половина”, “меньшинство”, “ни одного”$\}$. Учитывая вышеприведенное, определим набор е-гранул E в виде протоформ [4] для представления экспертных знаний о состоянии и поведении объекта:

$$\begin{aligned} E1:&\ Y \text{ is } rY, \quad Y \in Sy, \quad rY \in R,\\ E2:&\ B \text{ is } rB, \quad B \in Sb, \quad rB \in R,\\ E3:&\ Q \text{ is } rQ, \quad Q \in Sq, \quad rQ \in R, \end{aligned}$$

где выражение “A is rA” обозначает, что лингвистический термин A ограничен математической конструкцией r, глагол “is” (является) используется для текстового выражения этого ограничения. Так, например, нечеткий терм A “большинство” может быть ограничен функцией принадлежности $r_A(z)$, определенной на множестве частот [0,1] в виде правила:

$$r_A(z) = \begin{cases} 1 & \text{при} \quad z \geq 0{,}85, \\ 2z - 0{,}6 & \text{при} \quad 0{,}4 < z < 0{,}85, \\ 0 & \text{при} \quad z \leq 0{,}4. \end{cases}$$

Примеры использования лингвистической переменной как обобщенного ограничения для временного ряда приведены в публикациях [18, 32]. Введенные выше термины образуют терминологический словарь $S = \{Sy, Sb, Sq\}$, используемый для представления свойств в рамках дескриптивного анализа объекта. Заметим, что количество и состав е-гранул могут быть изменены в зависимости от контекста решаемой задачи. Множество математически заданных обобщенных ограничений $R = \{rY, rB, rQ\}$ является ключевым компонентом е-гранул, так как определяет способ грануляции МВР. На основе введенных обозначений определим модель expert-defined гранулы в виде

$$E =< Z, R, W,\ Ig >,$$

где Z обозначает лингвистический термин для анализируемого свойства, а его семантика определяется построенными на некотором множестве значений $W \supseteq X$ обобщенными ограничениями R в рамках выбранной теории грануляции Ig [2, 3, 8].

4. Грануляция МВР для дескриптивного анализа свойств объекта

Учитывая свойство иерархичности ИГ, грануляцию МВР будем рассматривать на нескольких уровнях, соответствующих значению показателя $m \in M$

и множеству показателей M, моменту времени $t \in T$ и временному интервалу наблюдения T, элементу $g \in G$ и множеству элементов G. В этом случае с каждым уровнем декомпозиции МВР сопоставим способ грануляции $f \in F$, генерирующий на основе e-гранул $E = \{E1, E2, E3\}$ data-extracted гранулы, соответствующие терминам терминологического словаря S, которые затем объединяются формальной конструкцией протоформы p. В табл. 1 приведены состав и параметры гранулярных вычислений, распределенных по уровням грануляции МВР "снизу-вверх". В первом столбце табл. 1 приведены уровни грануляции, для которых указаны представления модели МВР согласно выражению (1). При этом представление уровня 0 соответствует элементу модели МВР, определенному для конкретного показателя в заданный момент времени, в то время как уровень 3 определяет представление МВР (1) в виде множеств всех элементов, показателей и временных интервалов. Уровни грануляции 2 могут ассоциироваться с проекциями МВР по элементу или по атрибуту, или по моменту времени. Как видно из столбца 2 табл. 1, на каждом уровне грануляции МВР происходит обобщение гранул предыдущего уровня, которое выражает, с одной стороны, свойство иерархичности ИГ, а с другой — соответствует иерархической природе МВР, что важно в дескриптивном анализе. Результат информационной грануляции определяется обобщенными ограничениями из множества R и представлен в унифицированной форме в виде d-гранул. Отметим, что свойство иерархичности ИГ проявляется также в наборе генерируемых d-гранул, так как на основе ИГ состояния Y формируются ИГ тенденции B, а ИГ-квантификаторы Q образуются как для гранул-состояния, так и для гранул-тенденций. В последнем столбце табл. 1 приведены виды формируемых протоформ (2) и (3), связывающих d-гранулы и элементы различных уровней грануляции МВР. С использованием вышеприведенной грануляции МВР разработана методика дескриптивного анализа объекта O, позволяющая получать d-гранулы о его состоянии и поведении согласно выражению (4) и представлять эти знания в текстовой форме, она включает следующие этапы:

1. Представление экспертных знаний о состоянии и поведении объекта O в виде e-гранул и разработка способов их применения в гранулярных вычислениях:

   а) создание терминологического словаря используемых лингвистических терминов $S = \{Sy, Sb, Sq\}$ для дескриптивного анализа в зависимости от контекста задачи;

   б) определение множества обобщенных ограничений $R = \{rY, rB, rQ\}$ в виде математических выражений для каждого лингвистического термина, входящего в словарь S в рамках используемой теории грануляции [2, 3, 8];

   в) разработка алгоритмов гранулярных вычислений F для выбранных уровней грануляции МВР согласно табл. 1.
2. Реализация гранулярных вычислений F на МВР, формирование d-гранул и вывод множества пропозиций о состоянии и поведении объекта O:

а) применение алгоритмов грануляции F для получения d-гранул для МВР, используя состав и структуру параметров вычислений, приведенных в столбце 2 табл. 1;

б) формирование текста пропозиций с использованием протоформ P, приведенных в столбце 3 табл. 1.

Предложенная методика обеспечивает описание свойств, характеризующих состояние и поведение объекта исследования на разных уровнях грануляции по МВР в виде предложений на естественном языке. В данной методике ИГ поведения B описывают изменение значений показателя МВР в

Таблица 1. Многоуровневая грануляция МВР X на d-гранулы

Уровень грануляции X	Гранулярные вычисления, формирующие data-extracted гранулы D	Вид формируемых протоформ P
0: $m \in M$, $g \in G$, $t \in T$	$z0 = f0\,(x_{i,j,t}, E1, rY(X))$, $i, j, t = \text{const}$, $Y = z0$	$p1$: m is Y for g for t
1: $M, g \in G$, $t \in T$	$z11 = f11\,(x_j, z0, E3, rQ(m))$, $x_j = x_{i,j,t}$, $i, t = \text{const}$, $j = 1, 2, \ldots mk$, $Qm's = z11$, $Y = z0$	$p2$: $Qm's$ are Y for g for t
1: $G, m \in M$, $t \in T$	$z12 = f12\,(x_i, z0, E3, rQ(g))$, $x_i = x_{i,j,t}$, $g \in G$, $Y \in Sy$, $x_i = x_{i,j,t}$, $j, t = \text{const}$, $i = 1, 2, \ldots gk$, $Qg's = z12$, $Y = z0$	$p3$: $Qg's$ are Y for m for t
1: $T, g \in G$, $m \in M$	$z13 = f13\,(x_t, E2, rB(T))$, $x_t = x_{i,j,t}$, $i, j = \text{const}$, $t = 1, 2, \ldots, tk$, $B = z13$	$p4$: m is B for g for T
2: G, M, $t \in T$	$z21 = f21\,(x_{ij}, z11, E3, rQ(m))$, $x_{ij} = x_{i,j,t}$, $t = \text{const}$, $i = 1, 2, \ldots gk$, $j = 1, 2, \ldots mk$, $Qm's = z21$, $Y = z0$	$p5$: $Qm's$ are Y for t for G
2: $G, M, t \in T$	$z22 = f22\,(x_{ij}, z12, E3, rQ(g))$, $x_{ij} = x_{i,j,t}$, $t = \text{const}$, $i = 1, 2, \ldots gk$, $j = 1, 2, \ldots mk$, $Qg's = z22$, $Y = z0$	$p6$: $Qg's$ are Y for t for M
2: $G, T, m \in M$	$z23 = f23\,(x_{it}, z13, E3, rQ\,(g))$, $x_{it} = x_{i,j,t}$, $j = \text{const}$, $i = 1, 2, \ldots ik$, $t = 1, 2, \ldots, tk$, $Qg's = z23$, $B = z13$	$p7$: $Qg's$ are B for m for T
2: $G, T, m \in M$	$z24 = f24\,(x_{it}, z0, E3, rQ\,(g))$, $x_{it} = x_{i,j,t}$, $j = \text{const}$, $i = 1, 2, \ldots ik$, $t = 1, 2, \ldots, tk$, $Qg's = z24$, $Y = z0$	$p8$: $Qg's$ are Y for m for T
2: $M, T, g \in G$	$z25 = f25\,(x_{jt}, z13, E3, rQ\,(m))$, $x_{jt} = x_{i,j,t}$, $i = \text{const}$, $j = 1, 2, \ldots jk$, $t = 1, 2, \ldots, tk$, $Qm's = z25$, $B = z13$	$p9$: $Qm's$ are B for g for T
3: G, M, T	$z31 = f31\,(x_{ijt}, z23, E3, rQ(g))$, $i = 1, 2, \ldots, gk$, $j = 1, 2, \ldots mk$; $t = 1, 2, \ldots, tk$, $Qg's = z31$, $B = z13$	$p10$: $Qg's$ are B for M for T
3: G, M, T	$z32 = f32\,(x_{ijt}, z24, E3, rQ(m))$, $i = 1, 2, \ldots, gk$, $j = 1, 2, \ldots mk$; $t = 1, 2, \ldots, tk$, $Qm's = z32$, $B = z13$	$p11$: $Qm's$ are B for G for T

виде глобальной тенденции отдельного временного ряда и не учитывают его локальные тенденции. В то же время локальные тенденции могут служить эффективным средством для извлечения зависимостей, описывающих их пересечение, совпадение, опережение или отставание в МВР. В рамках предложенной методики эту задачу можно решить в три этапа. На первом этапе определить e-гранулы для локальных тенденций и соответствующие протоформы их описания. На втором этапе при грануляции МВР 1-го уровня во временном ряду показателя получить d-гранулы, выполнив темпоральную декомпозицию на временные интервалы с использованием e-гранул локальных тенденций. На третьем этапе извлечь d-гранулы, характеризующие темпоральные зависимости между локальными тенденциями.

5. Применение грануляции МВР для дескриптивного анализа региональных социально-экономических показателей

Изложенная выше методика была апробирована в задаче анализа состояния и тенденций развития экономики РФ по множеству социально-экономических показателей. Цель дескриптивного анализа в контексте решаемой задачи – получить оценку и выявить проблемы в состоянии и динамике развития экономики по субъектам РФ и показателям, объединенных в группы "информационное общество", "наука и инновации", "предпринимательство", "рынок труда" и "эффективность экономики". Для исследования были выбраны 15 показателей развития экономики за девять лет с 2010 по 2018 г. по 83 субъектам РФ[1], которые образовали МВР $X(83,\ 15,\ 9)$. Следуя методике, представленной в разделе 4, на первом этапе были разработаны нечеткие e-гранулы. В словарь терминов состояния были включены оценки уровня развития экономики субъектов РФ $Sy = \{$"низкий", "средний", "высокий"$\}$, а словарь терминов развития (поведения) содержал значения $Sb = \{$"рост", "стабильность", "падение"$\}$. Лингвистические термины Y моделировались равнобедренными треугольными функциями принадлежности нечетких множеств, носителями которых являлись следующие интервалы: "низкий": $0 \le x < 0{,}5$; "средний": $0{,}3 \le x < 0{,}9$; "высокий": $0{,}8 \le x \le 1{,}2$ (x – нормированное значение показателя). Чтобы получить нечеткое значение термина $B \in Sb$, был использован алгоритм, приведенный в [18], идея которого заключается в преобразовании временного ряда отдельного показателя в нечеткий временной ряд [32] и агрегировании интенсивностей изменений нечетких значений показателей. Словарь квантификаторов частотности Q содержал лингвистические термины $Sq = \{$"все", "большинство", "половина", "меньшинство", "ни одного"$\}$, в качестве обобщенных ограничений для которых использовались нечеткие множества, построенные на универсальном множестве частотности обнаружения гранул Y или B. Нечеткие термы Q моделировались треугольными функциями принадлежности, параметры которых приведены в табл. 2.

[1] Федеральная служба государственной статистики [Электронный ресурс]. URL: http://old.gks.ru/wps/wcm/connect/rosstat_main/rosstat/ru/ (дата обращения: 25.03.2021).

Таблица 2. Параметры функций принадлежности квантификаторов

Лингвистические термины квантификатора	Параметры носителя функции принадлежности нечеткого квантификатора Q
все	80, 100, 100
более половины	50, 70, 90
половина	40, 50, 60
менее половины	10, 30, 50
ни одного	0, 0, 20

Таблица 3. Протоформы и пропозиции, полученные в результате грануляции МВР

Протоформа	Пропозиция
$p1$: m is Y for g for t m = уровень ЗП, Y = Ниже Нормы, g = Астраханская область, t = 2018	В 2018 г. в Астраханской области показатель уровень ЗП был Ниже Нормы
$p2$: $Qm's$ are Y for g for t $Qm's$ = более половины, Y = Ниже Нормы, g = Республике Адыгея, t = 2017	В 2017 г. в Республике Адыгея более половины показателей были Ниже Нормы
$p3$: $Qg's$ are Y for m for t $Qg's$ = более половины, Y = Ниже Нормы, m = Внутр. затраты на научные исследования и разработки, t = 2018	В 2018 г. более половины субъектов имели показатель Внутр. затраты на научные исследования и разработки Ниже Нормы
$p5$: $Qm's$ are Y for t for G $Qm's$ = меньше половины, Y = Ниже Нормы, t = 2018	В 2018 г. меньше половины показателей экономики были Ниже Нормы
$p7$: $Qg's$ are B for m for T $Qg's$ = более половины, B = стабильность, m = Эффективность экономики, $T = [2010, 2018]$	С 2010 по 2018 г. более половины субъектов имели тенденцию стабильность в показателе Эффективность экономики
$p10$: $Qg's$ are B for M for T $Qg's$ = менее половины, B = негативная, $T = [2010, 2018]$	С 2010 по 2018 г. менее половины субъектов имели негативную тенденцию развития

Для всех протоформ вычислялась степень истинности, причем для протоформ с нечетким квантификатором Q использовалась формула лингвистического резюмирования, приведенная в [29]. С помощью порогового значения истинности ($\varepsilon \geq 0{,}7$) были выбраны протоформы для получения пропозиций, характеризующих состояние и тенденции развития экономики РФ с 2010 по 2018 г., некоторые из которых приведены в табл. 3. Отметим, что при переходе от протоформ к пропозициям в контексте задачи выявления проблем

были введены лингвистические оценки “ниже нормы” для состояний “низкий” и “средний”, и “негативная” тенденция для показателей, имеющих тенденцию “падение”. В результате дескриптивного анализа получены набор d-гранул, характеризующих регионы РФ и социально-экономические показатели с точки зрения наличия или отсутствия проблем в состоянии и динамике развития. Иерархия d-гранул позволяет анализировать объект исследования на разных уровнях абстракции, что является востребованным в системах поддержки принятия решений.

6. Заключение

В статье разработаны концептуальные основы грануляции МВР, расширяющие возможности представления свойств МВР в виде информационных гранул состояния и поведения сложных объектов. Предложена новая методика дескриптивного анализа объектов, основанная на многоуровневой грануляции МВР с использованием введенных expert-defined и data-extracted гранул.

Отличиями предложенной методики являются человекоцентричность, ориентация на поддержку принятия решений, формирование текстовых описаний о состоянии и поведении объектов в виде информационных гранул, которые в дальнейшем могут быть использованы для исследования зависимостей в свойствах объектов. Также отметим возможность сегментирования элементов, входящих в состав объекта, по лингвистически значимым для анализа оценкам из терминологического словаря. Результативность методики дескриптивного анализа на основе многоуровневой грануляции показана при анализе развития экономики в контексте субъектов РФ. Будущие исследования будут направлены на разработку подходов к решению задачи сходства и выявления зависимостей data-extracted гранул для последующего применения в диагностическом предиктивном анализе сложных объектов.

СПИСОК ЛИТЕРАТУРЫ

1. *Zadeh L.* Fuzzy Sets and Information Granularity // Advances in Fuzzy Set Theory and Appl., World Science Publishing, Amsterdam. 1979. P. 3–18.
2. *Zadeh L.* Toward a Theory of Fuzzy Information Granulation and Its Centrality in Human Reasoning and Fuzzy Logic // Fuzzy Sets and Syst. 1997. V. 90. P. 111–127.
3. *Zadeh L.* Generalized Theory of Uncertainty (GTU) – Principal Concepts and Ideas // Computational statistic & Data analysis. 2006. V. 51. P. 15–46.
4. *Zadeh L.* A Prototype-Centered Approach to Adding Deduction Capabilities to Search Engines – the Concept of a Protoform // Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS 2002). 2002. P. 523–525.
5. *Pedrycz W.* Granular Computing for Data Analytics: A Manifesto of Human-centric Computing // IEEE/CAA J. Autom. Sinica. 2018. V. 5. No. 6. P. 1025–1034.
6. *Федотова А.В., Ветров А.Н., Тарасов В.Б.* Грануляция информации при моделировании жизненного цикла сложных технических систем // Науковедение. 2013. № 5 (18).

7. *Pedrycz W., Skowron A., Kreinovich V.* Handbook of Granular Computing. Willey, 2008.
8. *Dubois D., Prade H.* Bridging Gaps Between Several Forms of Granular Computing // Granul. Comput. 2016. V. 1. P. 115–126. https://doi.org/10.1007/s41066-015-0008-8
9. *Yen G., Beliakov G., Triguero I., Pratama M., Zhang X., Li H.* Data Mining and Granular Computing in Big Data and Knowledge Processing. 2019. https://doi.org/10.1109/ACCESS.2019.2908776
10. *Pedrycz W.* Information Granules and Their Use in Schemes of Knowledge Management // Scientia Iranica. 2011. V. 18. No. 3. P. 602–610.
11. *Han Liu, Mihaela Cocea.* Fuzzy Information Granulation Towards Interpretable Sentiment Analysis // Granul. Comput. 2017. V. 2. No. 4. P. 289–302. https://doi.org/10.1007/s41066-017-0043-8
12. *Бутенков С.А.* Структурная организация гранулированных вычислений при обработке данных на реконфигурируемых вычислительных системах // Изв. ЮФУ. Технич. науки. 2018. № 8. С. 250–262.
13. *Бутакова М.А., Климанская Е.В., Чернов А.В.* Формальные структуры и представления для гранулярных вычислений // Современные наукоемкие технологии. 2018. № 5. С. 36–40.
14. *Bargiela A., Pedrycz W.* Granulation of Temporal Data: a Global View on Time Series // 22nd Int. Conf. of the North American Fuzzy Information Processing Society. 2003. P. 191–196. https://doi.org/10.1109/NAFIPS.2003.1226780
15. *Donga R., Pedrycz W.* A Granular Time Series Approach to Long-term Forecasting and Trend Forecasting // Physica A. 2008. V. 387. P. 3253–3270.
16. *Al-hmouz R., Pedrycz W.* Models of Time Series with Time Granulation // Knowledge and Inform. Syst. 2016. V. 48. No. 3. P. 561–580. https://doi.org/10.1007/s10115-015-0868-x
17. *Ярушкина Н.Г. и др.* Интеграция нечетко-гранулярных и онтологических методов в задаче анализа временных рядов // Автоматизация процессов управления. 2015. № 2 (40). С. 72–79.
18. *Afanasieva T., Moshkina I.* Descriptive Model of Temporal Features of Multivariate Time Series Based on Granulation // CEUR Workshop Proc. 2020. V. 2667. P. 287–292.
19. *Ярушкина Н.Г., Афанасьева Т.В., Тимина И.А.* Нечеткая грануляция в моделировании и прогнозировании объема телекоммуникационного трафика // Наукоемкие технологии. 2013. Т. 14. № 5. С. 67–72.
20. *Ярушкина Н.Г., Афанасьева Т.В.* Гранулярное моделирование временных рядов // Тринадцатая национальная конф. по искусственному интеллекту КИИ-2012. 2012. С. 143–148.
21. Онтологический и нечеткий анализ слабоструктурированных информационных ресурсов / под науч. ред. Н.Г. Ярушкиной. Ульяновск: УлГТУ, 2016.
22. *Jun M., LiXia W., XiuKun W., TsauYoung L.* Granulation-based Symbolic Representation of Time Series and Semi-supervised Classification // Computers & Math. with Appl. 2011. V. 62. No. 9. P. 3581–3590. https://doi.org/10.1016/j.camwa.2011.09.006
23. *Pedrycz W., Homenda W., Jastrzebska A., Yu F.* Information Granules and Granular Models: Selected Design Investigations // 2020 IEEE Int. Conf. on Fuzzy Systems (FUZZ-IEEE). 2020. P. 1–8. https://doi.org/ 10.1109/FUZZ48607.2020.9177696

24. *Novak V.* Linguistic Characterization of Time Series // Fuzzy Sets and Syst. 2016. V. 285. P. 52–72.

25. *Glockner I., Knoll A.* Fuzzy Quantifiers for Data Summarization and Their Role in Granular Computing // Proc. Joint 9th IFSA World Congr. and 20th NAFIPS Int. Conf. 2001. V. 4. P. 2029–2034. https://doi.org/ 10.1109/NAFIPS.2001.944380

26. *Kacprzyk J., Wilbik A., Zadroïny S.* Linguistic Summarization of Time Series Under Different Granulation of Describing Features // RSEISP 2007. 2007. V. 4585. P. 230–240. https://doi.org/10.1007/978-3-540-73451-2_25

27. *Kacprzyk J., Zadrozny S.* Linguistic Summaries of Time Series: A Powerful Tool for Discovering Knowledge on Time Varying Processes and Systems // Informatyka Stosowana. 2014. V. 1. P. 149–160.

28. *Kacprzyk J., Wilbik A., Zadroïny S.* Linguistic Summarization of Time Series Using a Fuzzy Quantifier Driven Aggregation // Fuzzy Sets and Syst. V. 159. No. 12. P. 1485–1499.

29. *Afanasieva T.V., Rodionova T.E.* Methodology of Patient-oriented Assessment of Cardiovascular Health of Men Using Fuzzy Sets and Formal Conceptual Analysis // World Scientific Proc. Series on Computer Engineering and Information Science Developments of Artificial Intelligence Technologies in Computation and Robotics. 2020. P. 857–865.

30. *Zadeh L.* A Computational Approach to Fuzzy Quantifiers in Natural Languages // Computers and Math. with Appl. 1983. V. 9. P. 149–184.

31. *Mörchen F., Ultsch A.* Mining Hierarchical Temporal Patterns in Multivariate Time Series. 2004. V. 3238. P. 127–140.

32. *Song Q., Chissom B.* Fuzzy Time Series and Its Models // Fuzzy Sets and Syst. 1993. V. 54. P. 269–277.

Статья представлена к публикации членом редколлегии О.П. Кузнецовым.

Поступила в редакцию 26.11.2021

После доработки 11.01.2022

Принята к публикации 26.01.2022

© 2022 г. Д.А. ЕГУРНОВ (egurnovd@yandex.ru),
Д.И. ИГНАТОВ, канд. техн. наук (dignatov@hse.ru)
(Национальный исследовательский университет
"Высшая школа экономики", Москва)

ТРИКЛАСТЕРЫ БЛИЗКИХ ЗНАЧЕНИЙ ДЛЯ АНАЛИЗА ТРЕХМЕРНЫХ ДАННЫХ[1]

Работа посвящена проблеме трикластеризации в многозначных триадических контекстах в терминах одного из многомерных расширений анализа формальных понятий, которая может быть рассмотрена как поиск плотных подтензоров в трехмерных тензорах над полем действительных чисел. Предлагаются два метода решения этой задачи: NOAC — вариант метода OAC-трикластеризации для числовых данных на основе дельта-операторов и триадическая версия метода k-средних с уточненной метрикой на основе манхэттенского расстояния и предикатов близости по каждому из трех измерений. Проведены численные эксперименты как на реальных, так и на синтетических данных, подтверждающие превосходство метода NOAC в смысле критериев качества найденных трикластеров.

Ключевые слова: трикластеризация, анализ формальных понятий, трехмерные тензоры, многозначные контексты.

DOI: 10.31857/S0005231022060071, **EDN:** ACVQYG

1. Введение

В современном мире характерно наличие насыщенной информационной среды, в которой скорость появления ресурсов и генерации новых данных растет с каждым годом. Часто главной проблемой, препятствующей успешному сбору, систематизации и извлечению знаний из этих данных, является отсутствие в них четко определенной структуры. Проблема анализа данных из неструктурированных источников занимает значительное место в современной прикладной математике и информатике [1]. Кластерный анализ предоставляет широкий спектр подходов и методов для определения внутренней структуры данных и классификации объектов. Основная идея заключается в объединении сходных по некоторым критериям объектов в группы, называемые кластерами.

Существуют методы, решающие задачи кластеризации объектов или признаков (одномерная кластеризация), и альтернативные методы, сохраняющие

[1] Статья подготовлена в результате проведения исследования в рамках Программы фундаментальных исследований Национального исследовательского университета "Высшая школа экономики" (НИУ ВШЭ).

объектно-признаковое описание сходства (бикластеризация) [2]. Тем не менее трикластеризация и n-мерная кластеризация как дальнейшее ее расширение не были столь же тщательно изучены (см., например, обзор [3]). В данной работе авторы ставят целью разработку и исследование алгоритмов для трикластеризации вещественных данных при наличии пропущенных значений, основываясь на методах OAC-трикластеризации (OAC от Object, Attribute, Condition) [4], сравнение эффективности работы этих алгоритмов, а также обработку ими реальных данных и интерпретацию результатов.

В ходе разработки новых алгоритмов были рассмотрены существующие методы, решающие близкие задачи, а именно: OAC-трикластеризация, основанная на штрих-операторах [4], метод шкалирования формальных понятий (Conceptual Scaling) [5] и его реализация TriMax [5], а также метод межпорядкового шкалирования (Interordinal Scaling) [5]. По различным причинам они не могли быть напрямую использованы для решения поставленной в данной работе задачи, поэтому авторы предлагают два метода для поиска трикластеров близких значений в многозначных триадических контекстах: метод NOAC (Numerical OAC), являющийся расширением на многозначный случай OAC-трикластеризации, основанной на штрих-операторах, и классический алгоритм кластеризации k-средних (k-means), использующий авторскую метрику для вычисления расстояний (Tri-k-means).

Данная работа состоит из четырех частей. Первая часть содержит некоторые базовые теоретические определения, формирующие основу предметной области и закладывающие математический аппарат для последующих исследований. В ней также содержатся все общие формулировки, необходимые для понимания описанных в данной работе алгоритмов. Во второй части рассматриваются существующие методы поиска би- и трикластеров в полиадических данных, а также приводится анализ их возможностей. В третьей части предложены два алгоритма, решающие поставленную выше задачу. В четвертой части кратко описаны результаты экспериментов на синтетических и реальных данных. В заключении подводится итог проделанной работе.

2. Базовые определения

2.1. Диадический случай

Для полноценного погружения в предметную область следует начать с формулировки базовых определений анализа формальных понятий (Formal Concept Analysis — FCA) [6].

Пусть даны два множества: G и M. Элементы множества G называют объектами (от нем. Gegenständ – объект), а элементы множества M – признаками (от нем. Merkmal – признак). Пусть также дано бинарное отношение I, являющееся подмножеством декартового произведения этих множеств: $I \subseteq G \times M$. Формальным контекстом называется тройка $K = (G, M, I)$. Если пара (g, m), где $g \in G$ и $m \in M$, принадлежит отношению инцидентности I, говорят, что "объект g обладает признаком m". Это обозначается как $(g, m) \in I$ или gIm.

Обычно формальные контексты представляются в виде матриц или таблиц с булевыми значениями.

Теперь рассмотрим два отображения $\phi : 2^G \to 2^M$ и $\psi : 2^M \to 2^G$. Пусть $\phi(A) = \{m \mid \forall g \in A : gIm\}$, $\psi(B) = \{g \mid \forall m \in B : gIm\}$. Для них выполняются следующие условия: (для $A_1, A_2 \subseteq G; B_1, B_2 \subseteq M$)

$$A_1 \subseteq A_2 \Rightarrow \phi(A_2) \subseteq \phi(A_1),$$
$$B_1 \subseteq B_2 \Rightarrow \psi(B_2) \subseteq \psi(B_1).$$

Эти отображения задают операторы Галуа для множеств объектов и признаков. Так как по составу множества можно однозначно определить, какой из операторов применяется, обычно используется единое обозначение $(\cdot)'$. Верны следующие свойства (для $A, A_1, A_2 \subseteq G, B \subseteq M$):

1) $A_1 \subseteq A_2 \Rightarrow A_2' \subseteq A_1'$,
2) $A_1 \subseteq A_2 \Rightarrow A_1'' \subseteq A_2''$,
3) $A \subseteq A''$,
4) $A' = A'''$ (также $A'' = A''''$),
5) $(A_1 \cup A_2)' \Leftrightarrow A_1' \cap A_2'$,
6) $A \subseteq B' \Leftrightarrow B \subseteq A' \Leftrightarrow A \times B \in I$.

Повторное применение этого же оператора дает оператор $(\cdot)'' : 2^G \to 2^G$, удовлетворяющий следующим свойствам (для $X, Y \subseteq G$):

1) $X \subseteq Y \Rightarrow X'' \subseteq Y''$ (монотонность),
2) $X \subseteq X''$ (экстенсивность),
3) $(X'')'' = X''$ (идемпотентность).

Все перечисленные выше утверждения верны с точностью до замены множества объектов G на множество признаков M и наоборот, соответственно. Таким образом, оператор $(\cdot)''$ является оператором замыкания.

Пара (A, B) называется формальным понятием в формальном контексте (G, M, I), если $A \subseteq G$, $B \subseteq M$, $A' = B$, $B' = A$. Множество A называют объемом (extent) формального понятия, а множество B – содержанием (intent) формального понятия. Если представить формальный контекст в виде матрицы с булевыми значениями, то формальные понятия будут соответствовать максимальным подматрицам из единиц, которые можно получить из исходной с помощью перестановки строк и столбцов.

Множество всех формальных понятий формального контекста упорядочено отношением частичного порядка

$$(A_1, B_1) \geqslant (A_2, B_2) \Leftrightarrow A_2 \subseteq A_1 (\Leftrightarrow B_1 \subseteq B_2)$$

и образует полную решетку, называемую решеткой формальных понятий.

2.2. Многозначные диадические контексты

Многозначный диадический контекст принято описывать кортежем (G, M, W, I), где W является множеством значений контекста, т.е. множеством значений, которые признаки $m \in M$ могут принимать на объектах $g \in G$, а $I \subseteq G \times M \times W$. При этом если $(g, m, v) \in I$ и $(g, m, w) \in I$, то $v = w$. Такие контексты обычно представляются в виде таблиц, где в ячейке на пересечении строки, соответствующей объекту g, и столбца, соответствующего признаку m, записано значение $m(g) \in W$. В самом общем случае бикластером для таких данных является пара (A, B), где $A \subseteq G$, $B \subseteq M$ [2, 7].

2.3. Триадический случай и n-адический случай

Расширим приведенные выше определения на случай большей размерности. В этом разделе рассмотрим определения для общего случая большой размерности, а затем отдельно выведем представляющие интерес определения, необходимые для трикластеризации [8–11].

Во-первых, стоит обобщить понятие формального контекста. Пусть X_1, $X_2, \ldots, X_n$ — некоторые множества, а I — n-арное отношение, являющееся подмножеством их декартова произведения: $I \subseteq X_1 \times X_2 \times \ldots \times X_n$. В таком случае формальным n-адическим контекстом будет называться кортеж $K = (X_1, X_2, \ldots, X_n, I)$. В триадическом контексте $K = (G, M, B, I)$, по аналогии с диадическим случаем, множества G и M называются соответственно множествами объектов и признаков, а множество B называется множеством условий (от нем. Bedingungen — условия). Тогда тройка $(g, m, b) \in I$, где $g \in G$, $m \in M$, $b \in B$ может интерпретироваться как "объект g обладает признаком m при условии b".

Расширение оператора Галуа $(\cdot)'$ приобретает следующие разновидности: $2^{X_1} \to 2^{X_2} \times \ldots \times 2^{X_n}, \ldots, 2^{X_n} \to 2^{X_1} \times \ldots \times 2^{X_{n-1}}, \ldots, 2^{X_1} \times \ldots \times 2^{X_{n-1}} \to$ $\to 2^{X_n}, \ldots, 2^{X_2} \times \ldots \times 2^{X_n} \to 2^{X_1}$. Таким образом, в триадическом случае оператор производит переход из любого из трех множеств в декартово произведение оставшихся двух либо обратно.

Кортеж $(A_1, A_2, \ldots, A_n)$ называется n-адическим формальным понятием, если выполняется условие: $(A_1 \times \ldots \times A_{n-1})' = A_n, \ldots, (A_2 \times \ldots \times A_n)' = A_1$. В триадическом случае первые два элемента тройки множеств называются, как и в диадическом случае, объемом и содержанием соответственно. Третье множество называется модусом (от лат. Modus — мера, образ, способ). В n-адическом формальном контексте формальное понятие также является максимальным кубоидом.

В общем случае, в отличие от диадического, повторное применение оператора $(\cdot)'$ хотя и определено как $(\cdot)''$, не является оператором замыкания.

2.4. Многозначные триадические контексты

Дополним определение триадического контекста для многозначного случая. Пусть G, M, B — множества соответственно объектов, признаков и

условий. Пусть W — множество допустимых значений контекста. Тогда 4-арное отношение есть $I \subseteq G \times M \times B \times W$. Если кортеж $(g, m, b, w) \in I$, говорят, что "признак m принимает значение w на объекте g при условии b". $K = (G, M, B, W, I)$ называется многозначным формальным контекстом. Стоит заметить, что отношение I определено таким образом, что любому триплету $(g \in G,\ m \in M,\ b \in B)$ соответствует не более одного значения $w \in W$. Это можно представить в виде (частичной) функции означивания $V : Q \to W$, где $Q \subseteq G \times M \times B$. Тогда область определения Q этой функции будет называться областью определения многозначного формального контекста.

В общем случае трикластером называют тройку (X, Y, Z), где $X \subseteq G$, $Y \subseteq M$, $Z \subseteq B$ [3, 4] .

В методах, реализованных в рамках этой работы, рассматривается только случай, когда W является множеством вещественных чисел $\mathbb{R}$.

2.5. Критерии качества трикластеров

Для оценки качества обнаруживаемых трикластеров предлагается использовать показатели плотности и дисперсии.

Пусть $T = (X, Y, Z)$ — трикластер многозначного формального контекста $K = (G, M, B, W, I)$ с функцией означивания $V : Q \to W$. Определим плотность трикластера T как отношение количества входящих в него троек к его размеру:

$$\rho(T) = \frac{|Q \cap (X \times Y \times Z)|}{|X||Y||Z|}.$$

Можно заметить, что такая оценка не учитывает значения входящих в трикластер троек. Поэтому будем оценивать трикластеры еще и по дисперсии значений содержащихся в нем триплетов. Пусть $S = V(g, m, b) \mid (g, m, b) \in \in Q \cap (X \times Y \times Z)$ — выборка, состоящая из этих значений. Тогда несмещенная оценка выборочной дисперсии будет выглядеть так:

$$s^2 = \frac{\sum\limits_{i=1}^{|S|} S_i^2 - \left(\sum\limits_{i=1}^{|S|} S_i\right)^2 / |S|}{|S| - 1}.$$

Значения трикластера будем считать близкими при некотором параметре δ, если стандартное отклонение выборки будет меньше или равно параметру, т.е. $s \leqslant \delta$.

3. Обзор существующих методов многомерной кластеризации

Формальные понятия в бинарном контексте представляют собой максимальные кубоиды из единиц, что требует наличия достаточно жесткой струк-

туры. Это может быть полезно в контекстах без шума, пропущенных значений и ошибок, однако реальные данные редко соответствуют этим требованиям. Следовательно, при анализе в терминах формальных понятий с большой вероятностью будет происходить потеря некоторой части значимой информации, что приведет к получению нерелевантных результатов. Возможным решением этой проблемы является ослабление определения формального понятия, допускающее неполное заполнение структуры (теперь в нее также могут входить нули, "пропущенные тройки"). В общем случае такие сущности называются n-кластерами. В диадическом и триадическом случаях это соответственно би- и трикластеры.

Стоит заметить, что не существует единого определения трикластера, поэтому каждый раз он определяется исходя из потребностей конкретной задачи и порождающего метода [2, 3]. По этой причине в дальнейшем будем сравнивать эффективность различных порождающих методов с помощью таких общих параметров, как плотность, дисперсия и количество трикластеров [8].

3.1. OAC-трикластеризация

Задача OAC-трикластеризации довольно подробно описана в [4]: рассматриваются два метода, основанные на так называемых бокс- и штрих-операторах, и демонстрируется превосходство второго метода над первым по показателям плотности и разнообразия. По этой причине обратим внимание на последний, который, по сути, является расширением метода OA-бикластеризации, описанной в [7] на триадический случай. Для фиксированной тройки $(g, m, b) \in I$ выпишем для удобства штрих-операторы, используемые в методе, явно:

$$(g, m)' = \{b \mid (g, m, b) \in I\},$$
$$(g, b)' = \{m \mid (g, b, m) \in I\},$$
$$(m, b)' = \{g \mid (g, b, m) \in I\}.$$

Тогда OAC-трикластером, основанным на штрих-операторах, построенным для тройки $(g, m, b) \in I$, будет называться тройка множеств $T = (((m, b)', (g, b)', (g, m)')$.

Отличительной особенностью построенных таким образом трикластеров является наличие в них плотной подструктуры, $(m, b)' \times \{m\} \times \{b\}, \{g\} \times \times (g, b)' \times \{b\}, \{g\} \times \{m\} \times (g, m)'$, трехмерного креста из "единиц" [4].

Алгоритм построения списка трикластеров описывается следующим образом: для каждой тройки $(g, m, b) \in I$ пополняются двумерные массивы, содержащие результаты выполнения операций штрих, например, $Prime[g, m] = = Prime[g, m] \cup b$, сам трикластер содержит три указателя на соответствующие массивы $T = (^*Prime[m, b],^* Prime[g, b],^* Prime[g, m])$.

Сложность алгоритма по времени (в худшем случае) составляет $O(|I|)$. Однако самой ресурсоемкой процедурой становится проверка уникальности трикластера, требующая, в худшем случае, сравнения нового трикластера

со всеми уже найденными. Этот шаг можно ускорить за счет использования различных способов хеширования и эффективного хранения информации для сравнения.

3.2. Шкалирование формальных понятий и метод TriMax

Метод шкалирования формальных понятий (Conceptual Scaling), рассмотренный в [5], предполагает поиск бикластеров близких значений в многозначных контекстах с помощью средств анализа формальных понятий (АФП).

Пусть $K = (G, M, W, I)$ — многозначный диадический формальный контекст. Определим бикластер близких значений с параметром θ как бикластер, где $\forall g_i, g_l \in G$, $\forall m_i, m_k \in M \;\; |m_i(g_j) - m_k(g_l)| \leqslant \theta$, т.е. все значения принадлежат одному классу толерантности ($m_i(g_j) \simeq_\theta m_k(g_l)$).

Далее в [4] предлагается выделить из множества W классы толерантности, сопоставить каждому из них формальный контекст, содержащий только элементы, входящие в данный класс, и произвести в них поиск формальных понятий, также являющихся бикластерами близких значений. Эта идея обрела воплощение в виде алгоритма TriMax [5] .

В приложении к проблеме, решаемой в данной работе, главным недостатком этого метода является использование инструментария анализа формальных понятий, действующего только в диадическом случае.

3.3. Межпорядковое шкалирование

Метод, рассмотренный в подразделе 3.2, зависит от параметра θ и ищет только бикластеры, в которых значения отстоят друг от друга не более чем на величину параметра. В этом подразделе познакомимся с методом межпорядкового шкалирования (Interordinal Scaling), предложенным в [5] и позволяющим производить поиск бикластеров близких значений по всем доступным значениям параметра сразу с помощью инструментов триадического анализа формальных понятий (Triadic Concept Analysis — TCA). Этот подход заключается в дополнении многозначного диадического формального контекста до шкалированного триадического. Третьим измерением в таком случае становятся интервалы, в которые входит значение, принимаемое признаком на объекте.

Пусть (G, M, W, I) — многозначный диадический формальный контекст. Построим новое интервальное измерение (scale dimension) T из W с помощью процедуры межпорядкового шкалирования (Interordinal Scaling). Шкала представляет собой бинарное отношение $J \subseteq W \times T$, сопоставляющее каждому исходному значению из W соответствующие элементы из T. Пусть $T = \{[\min(W), w], \forall w \in W\} \cup \{[w, \max(W)], \forall w \in W\}$. Тогда $(w, t) \in J$ тогда и только тогда, когда $w \in t$, где $t \subseteq T$.

Пусть $Y \subseteq G \times M \times T$ — тернарное отношение. В таком случае $(g, m, t) \in$ $\in Y$ тогда и только тогда, когда $m(g) \in t$. Кортеж (G, M, T, Y) называется

шкалированным триадическим контекстом многозначного диадического контекста (G, M, W, I).

В [5] доказывается, что кортеж (A, B, U), где $A \subseteq G$, $B \subseteq M$, $U \subseteq T$, является триадическим формальным понятием, только если (A, B) является бикластером близких значений для некоторого $\theta \geqslant 0$.

Этот метод довольно сложно адаптировать для поиска трикластеров близких значений в многозначных контекстах, так как он требует поиска формальных понятий в контекстах размерности большей, чем исходная, а соответствующие инструменты для тернарного случая исследованы лишь в общем виде.

4. Предложенные методы

Как было продемонстрировано в предыдущей части, существующие методы многомерной кластеризации не приспособлены решать поставленную в данной работе задачу, но содержат идеи, опираясь на которые можно спроектировать требуемый метод. В этом разделе предлагаются два алгоритма для поиска трикластеров близких значений в многозначных триадических контекстах. Первый является расширением OAC-трикластеризации на многозначный случай, а второй представляет собой разновидность классического алгоритма одномерной кластеризации k-средних, в котором используется предложенная первым автором данной работы метрика.

4.1. Метод NOAC

Метод NOAC (Numerical OAC) получен модификацией OAC-трикластеризации, основанной на штрих-операторах. Он принимает параметр δ, который определяет, какие значения считаются близкими. Пусть даны многозначный триадический контекст $K = (G, M, B, W, I)$ и (частичная) функция означивания $V : G \times M \times B \to W$, переводящая триплет (g, m, b) в соответствующее значение w тогда и только тогда, когда $(g, m, b, w) \in I$. Область определения этой функции обозначим как $Q \subseteq G \times M \times B$. Как и в бинарном случае, будем строить трикластер от фиксированной тройки $(\tilde{g}, \tilde{m}, \tilde{b}) \in Q$. Переопределим операторы, используемые методом:

$$(\tilde{g}, \tilde{m})^\delta = \left\{ b \;\middle|\; (\tilde{g}, \tilde{m}, b) \in Q \wedge |V(\tilde{g}, \tilde{m}, b) - V(\tilde{g}, \tilde{m}, \tilde{b})| < \delta \right\},$$
$$(\tilde{g}, \tilde{b})^\delta = \left\{ m \;\middle|\; (\tilde{g}, m, \tilde{b}) \in Q \wedge |V(\tilde{g}, m, \tilde{b}) - V(\tilde{g}, \tilde{m}, \tilde{b})| < \delta \right\},$$
$$(\tilde{m}, \tilde{b})^\delta = \left\{ g \;\middle|\; (g, \tilde{m}, \tilde{b}) \in Q \wedge |V(g, \tilde{m}, \tilde{b}) - V(\tilde{g}, \tilde{m}, \tilde{b})| < \delta \right\}.$$

Назовем эти операторы δ-операторами. Тогда OAC-трикластером, основанным на δ-операторах, построенном на тройке $(\tilde{g}, \tilde{m}, \tilde{b}) \in Q$, будет называться тройка множеств $T = \left((\tilde{m}, \tilde{b})^\delta, (\tilde{g}, \tilde{b})^\delta, (\tilde{g}, \tilde{m})^\delta \right)$.

Можно заметить, что построенный таким образом трикластер содержит плотную трехмерную подструктуру из близких значений по аналогии с немногозначным случаем. В силу того что результат применения δ-операторов зависит от конкретной тройки, в алгоритме NOAC не используется предподсчет. Для каждой тройки $(g, m, b) \in Q$ совершим два шага:

1) Вычислим множества $(g, m)^\delta$, $(g, b)^\delta$ и $(m, b)^\delta$.
2) Если трикластер $T = ((m, b)^\delta, (g, b)^\delta, (g, m)^\delta)$ еще не содержится в множестве трикластеров, добавим его туда.

Перебор всех троек в худшем случае займет $O(|I|)$. Построение трикластера производится за $O(|G| + |M| + |B|)$. Следовательно, сложность алгоритма можно оценить как $O(|I| \cdot \max(|G|, |M|, |B|))$. Отсутствие предподсчета освобождает от необходимости хранить лишние данные, поэтому сложность по памяти составит $O(|I|)$, что необходимо для хранения всех троек. Проверка уникальности вычисленного трикластера существенно усложняется с ростом общего количества трикластеров (в худшем случае $O(|I| \log(|I|))$.

4.2. Метод Tri-k-means

Для сравнения предложенного выше метода с альтернативным вариантом был выбран классический алгоритм одномерной кластеризации k-средних с авторской метрикой, учитывающей близость по компонентам входного триконтекста. Пусть $K = (G, M, B, W, I)$ — многозначный триадический контекст. Пусть функция $V : G \times M \times B \to W$ переводит тройку (g, m, b) в соответствующее значение w тогда и только тогда, когда $(g, m, b, w) \in I$, а $Q \subseteq$ $\subseteq G \times M \times B$ – ее область определения.

Расстояние между тройками $t_1 = (g_1, m_1, b_1) \in Q$ и $t_2 = (g_2, m_2, b_2) \in Q$ вычисляется по формуле

$$\rho(t_1, t_2) = |V(t_1) - V(t_2)| + \gamma([g_1 \neq g_2] + [m_1 \neq m_2] + [b_1 \neq b_2]),$$

где γ является вещественным параметром, определяющим приоритетность близости значений внутри трикластера относительно расширения его по измерениям. Выражение $[a_1 \neq a_2]$ принимает значение 0, если координаты тройки в соответствующем измерении эквивалентны, и 1 иначе.

В результате работы алгоритма получаем k кластеров, состоящих из троек. Для сравнения с методом NOAC требуется перевести их в трикластеры. Пусть $H \subseteq Q$ – множество триплетов, входящих в кластер. Тогда трикластером близких значений будем называть тройку множеств

$$T = (\{g \mid \exists (g, m, b) \in H\}, \{m \mid \exists (g, m, b) \in H\}, \{b \mid \exists (g, m, b) \in H\}).$$

Таким образом, в соответствующие измерения получившегося трикластера войдут все значения, содержащиеся в тройках исходного кластера. Эту адаптацию классического алгоритма k-средних было решено назвать методом Tri-k-means. Вычислительная сложность одной итерации алгоритма k-средних линейно зависит от параметра k и общего количества трикластеров, по-

этому может быть оценена как $O(k \cdot |G||M||B|)$. Пусть i – количество итераций алгоритма, тогда время работы всего алгоритма в худшем случае можно оценить как $O(k \cdot i \cdot |G||M||B|)$.

5. Эксперименты

Исходный код всех инструментов, подробное описание данных и результаты экспериментов доступны по ссылке: `https://github.com/EgurnovD/TriclusteringToolbox`.

5.1. Описание данных

Утилитой ContextGenerator был создан эталонный контекст, состоящий из 1120 троек, сформированных в два кубоида со значениями 3 и 7. Для него построены наборы контекстов с пропущенными значениями (10–90%) и размытием значений с различной амплитудой (0,1–2).

Контекст с реальными данными был создан на основе набора данных 100k проекта GroupLens [12] (`https://grouplens.org/datasets/movielens/100k/`), содержащего информацию о 100 000 оценках по пятибалльной шкале, поставленных 1000 пользователями сайта MovieLens 1700 фильмам при наличии 19 жанров.

5.2. Результаты экспериментов

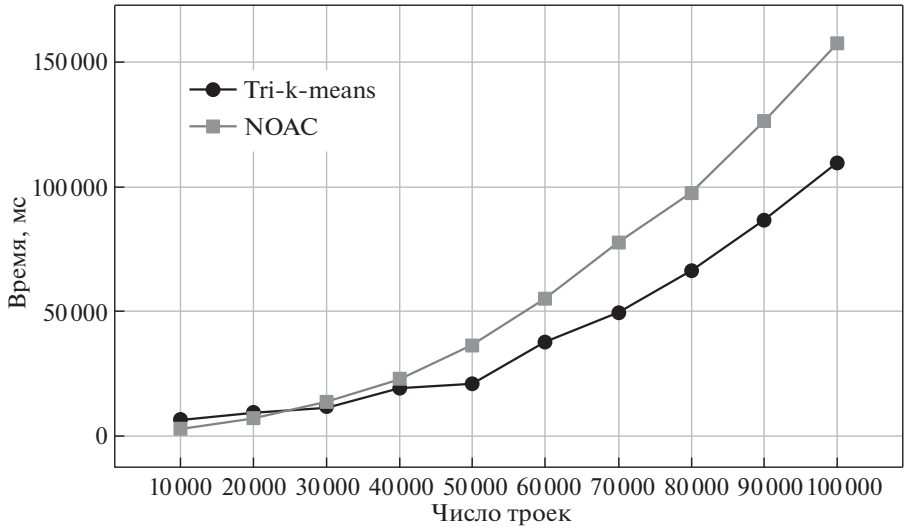
В первой серии экспериментов проверялась устойчивость алгоритмов к отсутствию данных. Метод NOAC смог найти эталонные трикластеры без дополнительной обработки при уровне потерь 10%. Метод Tri-k-means справился даже с 90% отсутствием данных за счет предварительного знания о количестве трикластеров (параметр k).

Вторая серия экспериментов была посвящена проверке на шумоустойчивость, в которой метод NOAC выдавал трикластеры с достаточно близкими значениями до тех пор, пока диапазоны размытия не пересеклись. Трикластеры, вычисленные методом Tri-k-means, в среднем имели большую дисперсию и перестали удовлетворять условию близости значений при меньшей амплитуде размытия.

На реальных данных и метод NOAC, и метод Tri-k-means генерируют трикластеры, удовлетворяющие условию близости значений, но метод NOAC порождает значительно более плотные трикластеры. Их можно интерпретировать как клубы по интересам, состоящие из пользователей, примерно одинаково оценивших схожие по жанрам фильмы. Отсутствующие значения в таком случае можно применить в рекомендательной системе, предполагая, что новые оценки будут дополнять существующие трикластеры, не сильно отклоняясь от среднего значения.

6. Заключение

В данной работе были рассмотрены современные методы би- и трикластеризации, а именно OAC-трикластеризация, шкалирование формальных



Зависимость времени работы от объема выборки.

понятий (Conceptual Scaling) и межпорядковое шкалирование (Interordinal Scaling), и предложено два алгоритма для поиска трикластеров близких значений в многозначных триадических контекстах. Один из них, являющийся расширением OAC-трикластеризации на многозначный случай, назван методом NOAC. Другой представляет собой классический алгоритм кластеризации k-средних и использует авторскую метрику для вычисления расстояний (метод Tri-k-means). Эксперименты показали, что на синтетических контекстах метод NOAC лучше справляется с размытием значений. Из реальных данных он извлекает более качественные трикластеры в терминах плотности, но тратит на это немного больше времени (см. рисунок). Стоит отметить, что метод NOAC принимает только один легко интерпретируемый параметр.

СПИСОК ЛИТЕРАТУРЫ

1. *Zaki M.J., Meira Jr W.* Data Mining and Machine Learning: Fundamental Concepts and Algorithms. Cambridge University Press, 2020.
2. *Madeira S.C., Oliveira A.L.* Biclustering algorithms for biological data analysis: a survey // IEEE/ACM Trans. on Comp. Biol. and Bioinf., IEEE. 2004. V. 1. No. 1. P. 24–45.
3. *Henriques R., Madeira S.C.* Triclustering algorithms for three-dimensional data analysis: a comprehensive survey // ACM Computing Surveys (CSUR), ACM. 2018. V. 51. No. 5. P. 1–43.
4. *Ignatov D.I., Gnatyshak D.V., Kuznetsov S.O., Mirkin B.G.* Triadic formal concept analysis and triclustering: searching for optimal patterns // Machine Learning, Springer. 2015. V. 101. No. 1. P. 271–302.
5. *Kaytoue M., Kuznetsov S.O., Macko J., Napoli A* Biclustering meets triadic concept analysis // Ann. Math. Artif. Intell., Springer. 2014. V. 70. No. 1. P. 55–79.
6. *Ganter B., Wille R.* Formal Concept Analysis: Mathematical Foundations. Berlin/ Heidelberg: Springer, 1999.

7. *Ignatov D.I., Kuznetsov S.O., Poelmans J.* Concept-based biclustering for internet advertisement // In proc. ICDMW, IEEE. 2012. P. 123–130.
8. *Lehmann F., Wille R.* A triadic approach to formal concept analysis // Conceptual Structures, LNCS, Springer. 1995. V. 954. P. 32–43.
9. *Voutsadakis G.* Polyadic Concept Analysis // Order, Springer Verlag. 2002. V. 19. No. 3. P. 295–304.
10. *Ganter B., Obiedkov S.* Implications in triadic formal contexts // In proc. International Conference on Conceptual Structures, Springer. 2004. P. 186–195.
11. *Ananias K.H.A., Missaoui R., Ruas P.H.B., Zarate L.E., Song M.A.J.* Triadic concept approximation // Information Sciences, Elsevier. 2021. V. 572. P. 126–146.
12. *Harper F.M., Konstan J.A.* The movielens datasets: History and context // ACM Transact. on Interactive Intell. Syst., ACM. 2015. V. 5. No. 4. P. 1–19.

Статья представлена к публикации членом редколлегии О.П. Кузнецовым.

Поступила в редакцию 20.01.2021

После доработки 07.02.2022

Принята к публикации 15.02.2022

© 2022 г. К.С. ЯКОВЛЕВ, канд. физ.-мат. наук (yakovlev@isa.ru),
Ю.С. БЕЛИНСКАЯ, канд. физ.-мат. наук (belinskaya.us@gmail.com),
Д.А. МАКАРОВ, канд. физ.-мат. наук (makarov@isa.ru),
(Федеральный исследовательский центр
"Информатика и управление" РАН, Москва),
А.А. АНДРЕЙЧУК (andreychuk@mail.com)
(Институт искусственного интеллекта AIRI, Москва;
Российский университет дружбы народов, Москва)

БЕЗОПАСНО-ИНТЕРВАЛЬНОЕ ПЛАНИРОВАНИЕ И МЕТОД НАКРЫТИЙ ДЛЯ УПРАВЛЕНИЯ ДВИЖЕНИЕМ МОБИЛЬНОГО РОБОТА В СРЕДЕ СО СТАТИЧЕСКИМИ И ДИНАМИЧЕСКИМИ ПРЕПЯТСТВИЯМИ[1]

Рассматривается задача управления движением колесного мобильного робота с дифференциальным приводом в среде со статическими и динамическими препятствиями. Предлагается многоэтапный подход к решению этой задачи, основанный на применении методов эвристического поиска для решения задачи планирования траектории и методов теории автоматического управления для следования по траектории. Результаты проведенных экспериментальных исследований (численного моделирования) свидетельствуют о том, что предложенный метод следования по траектории на порядок быстрее, чем один из наиболее распространенных в робототехнике законов управления — управление с прогнозирующими моделями (MPC, Model Predicitve Control), и способен обеспечивать высокое качество работы. При этом само планирование осуществляется за приемлемое время.

Ключевые слова: планирование траектории, мобильная робототехника, плоская система, динамические препятствия, примитивы движений.

DOI: 10.31857/S0005231022060083, **EDN:** ACWFDF

1. Введение

Движение к цели в среде со статическими и динамическими препятствиями — одна из фундаментальных проблем робототехники. Можно выделить два различных подхода к ее решению: реактивный и делиберативный. В рамках первого подхода (см., например, алгоритмы семейства BUG [1] или ORCA [2]) реализуются законы управления, направленные на избегание столкновений и продвижение к цели. При этом законы опираются лишь на локальные наблюдения агента (мобильного робота). В рамках второго

[1] Исходный код алгоритма планирования доступен по ссылке: `bit.ly/3mEBK59`. Видеодемонстрация доступна по адресу: `youtu.be/cYm3Q1tHRnU`.

подхода обычно происходит декомпозиция задачи навигации на планирование траектории и следование по ней. Алгоритмы планирования, такие как D*Lite [3] (для статических сред) или SIPP [4] (для сред с динамическими препятствиями), формируют желаемую траекторию, следование по которой обеспечивается с помощью методов теории управления, например с помощью управления с прогнозирующими моделями — MPC [5], часто применяющегося в мобильной робототехнике.

В данной статье предполагается, что управляемый мобильный агент — это колесный робот с дифференциальным приводом, а модель окружающего пространства (карта) известна заранее. Также предполагается, что у робота имеется отдельная система предсказания траекторий движения динамических препятствий (например, основанная на техническом зрении). Более того, предполагается, что карта и предсказанные траектории движения динамических объектов являются точными. Это весьма сильное допущение, которое, однако, позволяет разделить задачи построения модели окружающего мира и планирования в этой модели. В статье рассматривается вторая задача — планирование и исполнение плана в заранее известной (динамической) модели мира. Для ее решения представляется целесообразным использование делиберативного подхода, а именно: сначала строится траектория, которая избегает столкновения со статическими и динамическими препятствиями, затем осуществляется следование по ней.

При реализации такого подхода возникают следующие трудности. Во-первых, при планировании необходим учет временно́го измерения для того, чтобы избежать столкновений с динамическими препятствиями, т.е. планирование должно осуществляться не только в пространстве, но и во времени. Во-вторых, необходимо также учитывать кинематические и динамические ограничения рассматриваемого робота, для того чтобы впоследствии регулятор робота смог обеспечить движение по спланированной траектории с высокой точностью.

Для преодоления указанных трудностей в статье предлагается следующая схема решения задачи. Сначала происходит построение множества кинематически и динамически согласованных примитивов движения — коротких отрезков траектории, удовлетворяющих ограничениям на динамику движения объекта управления. Это построение использует свойство плоскостности системы, описывающей движение робота с дифференциальным приводом. Для построения примитивов используется метод, основанный на понятии накрытия [6], который позволяет свести задачу терминального управления (краевую задачу) к двум связанным задачам Коши, получение решений для которых — гораздо более простая задача, чем получение решения исходной краевой задачи напрямую. Кроме того, метод накрытий позволяет находить не только траекторию маневра, но и программное управление, реализующее движение по заданному примитиву (это используется в дальнейшем при разработке регулятора). Построенные примитивы движения интегрируются далее в алгоритм безопасно-интервального планирования SIPP [4], который в

явном виде позволяет оперировать интервалами времени и, таким образом, учитывать темпоральный аспект задачи планирования. Также при построении траектории учитываются ограничения на линейную скорость движения робота. Результатом работы планировщика является траектория, избегающая столкновений как со статическими, так и с динамическими препятствиями. Наконец, для следования по спланированной траектории предлагается оригинальный закон управления, который вновь опирается на свойство плоскостности системы движения робота и обеспечивает следование по траектории с высокой точностью.

В рамках проведенного экспериментального исследования (численного моделирования) показано, что *а*) предлагаемый алгоритм планирования способен строить траектории в средах с высоким числом динамических препятствий за приемлемое время (менее 1/4 секунды в тестах на стандартном современном (2021 г.) вычислителе с процессором Intel Core i5); *б*) предлагаемый регулятор на порядок быстрее, чем наиболее часто используемый в робототехнике регулятор MPC [5]; *в*) ошибка следования по траектории для предлагаемого регулятора ниже, чем ошибка MPC.

Новые научные результаты статьи состоят в следующем.

— Предлагается новый метод построения кинематически и динамически согласованных примитивов движения, учитывающий ограничения на динамику движения мобильного робота с дифференциальным приводом.

— Предлагается оригинальный алгоритм планирования, который использует построенные примитивы движения и опирается на подход безопасно-интервального планирования для построения траектории, избегающей столкновений как со статическими, так и с динамическими препятствиями.

— Предлагается оригинальный закон управления движением робота (регулятор) для следования по построенной траектории, который характеризуется высоким быстродействием и низкой ошибкой следования по траектории.

Дальнейший текст статьи структурирован следующим образом. В разделе 2 приводится обзор литературы, релевантной теме исследования. Раздел 3 содержит формальную постановку рассматриваемой задачи. Способ решения этой задачи описывается в общем виде в разделе 4 и далее конкретизируется в разделах 5–7. Раздел 8 посвящен экспериментальным исследованиям и их результатам. Заключительный раздел 9 содержит описание перспективных направлений дальнейших исследований.

2. Обзор литературы

2.1. Построение примитивов движения

Понятие примитивов движения достаточно часто возникает в литературе по планированию, когда речь идет о построении траекторий с помощью методов эвристического поиска [7] или методов, основанных на сэмплировании [8]. Характерным примером использования примитивов движения при планировании является публикация [9]. Стоит отметить, что на практике для

построения примитива движения необходимо решить задачу терминального управления или краевую задачу с двумя условиями (на начало и конец примитива). Для этого в [10] предлагается метод стрельбы (shooting method), который предполагает постоянное управляющее воздействие. Более универсальный подход предлагается в [11]. В [12] описывается подход к построению примитивов движения для плоских динамических систем с применением B-сплайнов. Авторы настоящей статьи также используют свойство плоскостности, однако построение примитива происходит за счет более универсального метода накрытий [6].

2.2. Планирование в среде с динамическими препятствиями

Часто эта нетривиальная задача сводится к постоянному перепланированию, при котором положения динамических препятствий фиксируются в момент очередного перестроения плана [13]. При этом на этапе планирования нет необходимости учета временной компоненты, что можно считать определенным преимуществом. С другой стороны, этот подход может не находить решения, хотя оно существует. Для непосредственного учета динамики окружающей среды на этапе планирование обычно вводят в рассмотрение дискретную шкалу времени и применяют методы эвристического поиска, которые расширяют состояния поиска соответствующими временными отметками [14]. Более универсальный подход, развивающий эту идею, предложен в [4], где представлен алгоритм безопасно-интервального планирования SIPP. В данной статье предлагается адаптация этого алгоритма, позволяющая конструировать траекторию из примитивов движения.

2.3. Следование по траектории

К настоящему моменту известно множество методов, предназначенных для выработки управляющих воздействий для обеспечения следования по заданной траектории, например: линеаризация обратной связи [15], метод обратного хода (backstepping) [16], управление с плавающим окном [17], управление с прогнозирующими моделями (MPC) [18] и др. Если управляемая система является плоской [19], то перспективным представляется создание законов управления, опирающихся на это свойство [20, 21]. Для подобных систем известен ряд подходов к построению законов управления [19, 22]. Например, распространен подход, при котором осуществляется линеаризация статической обратной связью и выбор траектории в виде полиномиальной зависимости от времени [22]. В данной работе подобный подход комбинируется с методом накрытий [23].

3. Постановка задачи

Рассмотрим колесный мобильный робот с дифференциальным приводом (см. рис. 1,a), динамическая модель которого имеет общий вид: $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$,

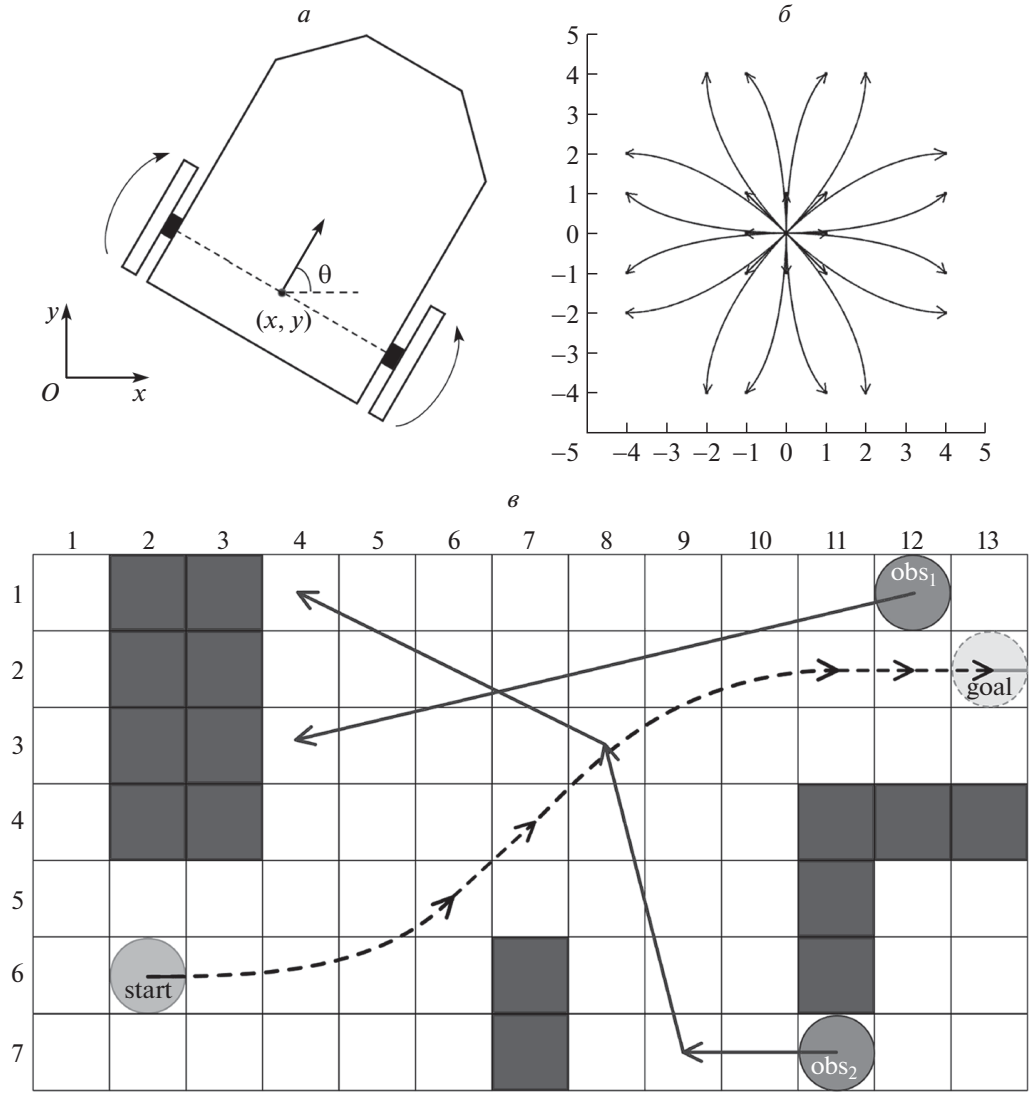


Рис. 1. *а* — Мобильный робот с дифференциальным приводом; *б* — Подмножество предлагаемых в статье примитивов движения робота; *в* — Пример траектории, избегающей столкновений со статическими и динамическими препятствиями. Анимация этого примера доступна по ссылке: `youtu.be/cYm3Q1tHRnU`.

где f – вектор-функция, $\mathbf{x}$ – состояние, $\mathbf{u}$ – управление, а $\dot{\mathbf{x}} = d\mathbf{x}/dt$ – производная по времени, задана, в частности, следующим образом [24]:

$$\dot{x} = v\cos\theta, \quad \dot{y} = v\sin\theta, \quad \dot{\theta} = \omega, \quad t \in \mathcal{T}. \tag{1}$$

Здесь $\mathcal{T} = [0, \infty)$ – время, (x, y) – положение центра колесной оси робота, θ – угол ориентации. Управление, $\mathbf{u} = (v, \omega)^{\mathrm{T}}$, состоит из линейной и угловой скоростей. Знак $^{\mathrm{T}}$ обозначает транспонирование.

Имеются также ограничения на максимальные скорости и ускорения как линейные, так и угловые. Множество допустимых управлений задается в виде

$$\mathbf{U} = \left\{ \mathbf{u} : |v| \leqslant v_{\max},\ \left|\frac{dv}{dt}\right| \leqslant a_{\max},\ |\omega| \leqslant \omega_{\max},\ \left|\frac{d\omega}{dt}\right| \leqslant \varepsilon_{\max} \right\}.$$

Рабочее пространство робота — это замкнутое подмножество в $\mathbb{R}^2$ (Евклидовом двумерном пространстве): $\mathcal{W} = \mathcal{W}_{free} \cup \mathcal{W}_{obs}$, где $\mathcal{W}_{free}$ – свободное пространство, $\mathcal{W}_{obs}$ – статические препятствия. Множество всех допустимых состояний робота, (x, y, θ), с учетом статических препятствий обозначим как $\mathcal{C}_{free}$. Помимо робота в среде перемещается фиксированное число динамических препятствий. Их траектории считаются известными, поэтому обозначим через $\mathcal{C}_{free}(t) \subset \mathcal{C}_{free}$ множество безопасных состояний робота в момент времени t, т.е. таких состояний, в которых не происходит столкновения ни со статическими, ни с динамическими препятствиями.

Решаемая задача формулируется следующим образом. Пусть задано начальное, $\mathbf{x}^0 = \mathbf{x}(0) \in \mathcal{C}_{free}(0)$, и целевое, $\mathbf{x}^f \in \mathcal{C}_{free}$, состояния робота. Необходимо найти такое управление $\mathbf{u}(\mathbf{x}(t), t),\ t \in [0, T_f]$, которое переводит (1) из $\mathbf{x}^0$ в $\mathbf{x}^f$, так что $\mathbf{x}(t) \in \mathcal{C}_{free}(t)\ \ \forall t \in [0, T_f]$ и $\mathbf{u} \in \mathbf{U}$. Заметим, что в данной статье предполагается, что после достижения целевого состояния робот не исчезает. Это обстоятельство накладывает следующее дополнительное требование: $\forall t > T_f : \mathbf{x}(t) \in \mathcal{C}_{free}(t)$. Другими словами, представляет интерес не просто сам факт достижения роботом цели, а достижение цели в такой момент времени, что робот может безопасно оставаться в целевом положении сколь угодно долго. Также заметим, что при решении поставленной задачи целесообразно минимизировать T_f, хотя формальное требование к достижению минимально возможного значения T_f в данной постановке отсутствует.

4. Общее описание предлагаемого подхода

В сложных средах с динамическими препятствиями решение поставленной задачи поиска допустимого управления с учетом ограничений напрямую не представляется эффективным с вычислительной точки зрения [25]. Поэтому целесообразным является применение декомпозиционного подхода, когда сначала ищется траектория (зависимость переменных состояния от времени), а затем управление строится целенаправленно для следования по этой траектории [12]. Соответственно в данной статье предлагается декомпозиция исходной задачи на следующие три подзадачи: построение выполнимых примитивов движения, планирование траектории, избегающей столкновения со статическими и динамическими препятствиями, в виде последовательности таких примитивов, следование по построенной траектории.

На первом шаге осуществляется построение примитивов движения, учитывающих динамическую модель рассматриваемой робототехнической системы, а также ограничения на управление: $\mathbf{u} \in \mathbf{U}$. При этом ограничения на состояния на данном этапе не учитываются. Формально примитивы движения представляются как зависимости переменных состояния от времени: $\bar{x}(t)$,

$\bar{y}(t)$, $\bar{\theta}(t)$. Неформально каждый примитив — это короткий (как в пространственном, так и в темпоральном измерениях) фрагмент траектории, учитывающий кинематические и динамические ограничения объекта управления (колесного робота). Примеры подобных примитивов показаны на рис. 1,*б*.

На втором шаге построенные примитивы используются в качестве базовых блоков при решении задачи планирования траектории, т.е. траектория конструируется из них. На этом этапе осуществляется учет ограничений на $\mathbf{x}$, т.е. траектория планируется таким образом, чтобы избегать столкновений как со статическими, так и с динамическими препятствиями. Также для этой цели при планировании используются неявно определенные действия ожидания. Результатом работы алгоритма планирования (планировщика) является последовательность согласованных в пространстве и времени примитивов (а также действий ожидания определенной продолжительности), т.е. желаемая траектория движения, которую обозначим как $\mathbf{x}^*(t) = (x^*(t), y^*(t), \theta^*(t))^{\mathrm{T}}$. Пример траектории, составленной из примитивов (и действий ожидания), показан на рис. 1,*в*. Анимация этого примера доступна по ссылке: `youtu.be/cYm3Q1tHRnU`.

На третьем шаге следование по траектории $\mathbf{x}^*(t)$ осуществляется за счет построения допустимого управления $\mathbf{u}(\mathbf{x}, t) \in \mathbf{U}$, которое обеспечивает асимптотическую стабильность системы (1) в окрестности $\mathbf{x}^*(t)$.

Далее опишем указанные шаги более подробно.

5. Построение примитивов

Известно, что система (1) является (*дифференциально*) *плоской* [19], т.е. для этой системы известен набор функций плоского выхода, причем все переменные состояния и управления могут быть выражены в терминах плоского выхода и конечного числа его производных. В рассматриваемом случае плоский выход имеет вид:

$$\zeta_1 = x, \quad \zeta_2 = y. \tag{2}$$

Действительно, все переменные состояния, x, y, θ, и управления, v, ω, могут быть выражены через функции (2) и их производные первого порядка следующим образом:

$$x = \zeta_1, \quad y = \zeta_2, \quad \theta = \arctan\frac{\dot{\zeta}_1}{\dot{\zeta}_2}, \quad v = \sqrt{\dot{\zeta}_1^2 + \dot{\zeta}_2^2}, \tag{3}$$

$$\omega = \frac{d}{dt}\left(\sqrt{\dot{\zeta}_1^2 + \dot{\zeta}_2^2}\right) = \frac{\dot{\zeta}_1\ddot{\zeta}_1 + \dot{\zeta}_2\ddot{\zeta}_2}{\sqrt{\dot{\zeta}_1 + \dot{\zeta}_2}}.$$

Вводя дополнительную переменную $\xi = v = \sqrt{\dot{\zeta}_1^2 + \dot{\zeta}_2^2}$, получаем расширенную систему [22]:

$$\dot{x} = \xi\cos\theta, \quad \dot{y} = \xi\sin\theta,$$

$$\dot\theta = \frac{1}{\xi}(\psi_1 \sin\theta - \psi_2 \cos\theta), \quad \dot\xi = \psi_1 \cos\theta + \psi_2 \sin\theta, \tag{4}$$

которая эквивалентна системе нормальной формы Бруновского второго порядка:

$$\ddot\zeta_1 = \psi_1, \quad \ddot\zeta_2 = \psi_2. \tag{5}$$

Обозначим через $\tilde{\mathbf{x}}$ расширенный вектор состояния $\tilde{\mathbf{x}} = (x, y, \theta, \xi)^{\mathrm{T}}$, а через ψ – новое управление: $\psi = (\psi_1, \psi_2)^{\mathrm{T}}$. Заметим, что система (1) плоская в области $v \neq 0$.

Будем использовать свойство плоскостности для построения примитивов движения. Примитив движения определим как решение следующей задачи терминального управления для (4):

$$x(0) = x_0, \quad y(0) = y_0, \quad \theta(0) = \theta_0, \quad \xi(0) = \xi_0 \neq 0, \tag{6}$$

$$x(t_f) = x_f, \quad y(t_f) = y_f, \quad \theta(t_f) = \theta_f, \quad \xi(t_f) = \xi_f \neq 0, \tag{7}$$

которая может быть переписана в терминах плоского выхода следующим образом:

$$\zeta_i(0) = \zeta_{i0}, \quad \dot\zeta_i(0) = \dot\zeta_{i0}, \tag{8}$$

$$\zeta_i(t_f) = \zeta_{if}, \quad \dot\zeta_i(t_f) = \dot\zeta_{if}, \quad i = 1, 2. \tag{9}$$

Граничные условия x_0, x_f, y_0, y_f, θ_0, θ_f, ξ_0, ξ_f, а также продолжительность t_f определяют конкретный примитив движения.

Для решения терминальной задачи (8) будет использоваться метод накрытий [23] для плоских систем [26]. Сначала находим r-замыкание — систему дифференциальных уравнений, все решения которой являются решениями изначальной системы. Поскольку рассматриваемая задача имеет 8 граничных условий, в качестве r-замыкания можно выбрать любую систему восьмого порядка [26]. Для простоты выберем систему

$$\zeta_i^{(4)} = 0, \quad i = 1, 2. \tag{10}$$

Для решения (10) нужно получить начальные условия на вторую и третью производные ζ_i. Для этого рассмотрим вспомогательные функции

$$p_i = \zeta_i - \frac{1}{2}(t_f - t)^2 \ddot\zeta_i - \frac{1}{3}(t_f - t)^3 \dddot\zeta_i, \quad i = 1, 2, \tag{11}$$

производные по времени которых с учетом (10) равны

$$\dot p_i = \dot\zeta_i + (t_f - t)\ddot\zeta_i + \frac{1}{2}(t_f - t)^2 \dddot\zeta_i, \tag{12}$$

$$\ddot p_i = 0, \quad i = 1, 2. \tag{13}$$

Теперь начальные условия на вторую и третью производные ζ_i могут быть вычислены с помощью следующего *алгоритма.*

Шаг 1. Вычисляем $p_i(t_f)$ и $\dot{p}_i(t_f)$, $i = 1, 2$, из соотношений (9) и (12) подстановкой $t = t_f$ и $\zeta_i(t_f) = \zeta_{if}$ соответственно.

Шаг 2. Решаем задачу Коши для системы дифференциальных уравнений (13) с начальными условиями $p_i(t_f)$, $\dot{p}_i(t_f)$, $i = 1, 2$, в сторону уменьшения времени. В результате получим зависимости от времени $p_i(t)$, $i = 1, 2$, $t \in \in [0, t_f]$.

Шаг 3. Вычисляем $p_i(0)$, $\dot{p}_i(0)$, $i = 1, 2$.

Шаг 4. Подставляем найденные значения в (11), (12), полагая $t = 0$, и разрешаем уравнения (11), (12) относительно $\ddot{\zeta}_i(0)$, $\dddot{\zeta}_i(0)$, $i = 1, 2$.

В результате будет получен полный набор начальных условий для (10), и можно решить задачу Коши для этого уравнения. Решение дает примитив движения для заданных граничных условий и продолжительности движения. Дифференцируя найденные функции $\zeta_i(t)$ по времени, получим программное управление, реализующее движение вдоль найденного маневра:

$$\psi_i = \ddot{\zeta}_i, \quad i = 1, 2. \tag{14}$$

5.1. Примитивы для реализации перемещения

Описанный выше алгоритм позволяет построить любой примитив для перемещения из заданной начальной точки $(x_0, y_0, \theta_0, v_0)$ в заданную конечную точку $(x_f, y_f, \theta_f, v_f)$, если начальная, v_0, и конечная, v_f, скорости не равны нулю. В настоящей статье граничные условия для примитивов перемещения выбираются следующим образом:

- $x_0 = 0$, $y_0 = 0$, а значения x_f, y_f — целые числа. Это позволяет ввести дискретизацию рабочего пространства таким образом, чтобы робот перемещался из центра одной ячейки в центр другой;
- Продолжительность примитива t_f выбирается вручную таким образом, чтобы она была как можно меньше при соблюдении ограничений на управление (на практике расчет примитива начинается с достаточно большого значения t_f, которое затем уменьшается с повторением процедуры расчета до тех пор, пока не будут нарушены ограничения на управление);
- $v_0, v_f \in [\sigma, v_d]$, где σ – небольшое положительное число, нужное для того, чтобы избежать неуправляемости расширенной системы при $v = 0$, и $v_d < v_{\max}$ – желаемая скорость.

  Полагая $v_0 = \sigma$ и $v_f = v_d$, имеем маневр разгона от полной остановки, а $v_0 = v_d$ и $v_f = \sigma$ – торможение до полной остановки.

  Параметры $v_0 = v_f = v_d$ задают равномерное движение.

  Наконец, параметры $v_0 = v_f = \sigma$ задают маневр, при котором робот начинает движение от остановки, далее разгоняется, затем тормозит и снова останавливается;

- $\theta_0 = k \cdot \frac{\pi}{4}$, $k = 0, \ldots, 7$ и $\theta_f \in [\theta_0 - \frac{\pi}{4}, \theta_0, \theta_0 + \frac{\pi}{4}]$. Таким образом, начальная и конечная ориентации примитивов принимают ограниченный набор значений с шагом 45°, и робот в процессе исполнения примитива может либо изменить ориентацию на 45°, либо оставить ее без изменений (при прямолинейном движении).

На рис. 1,*б* изображены в качестве примера проекции на плоскость Oxy примитивов разгона. Всего в данной статье было построено 96 различных примитивов перемещения. При построении примитивов использовались следующие ограничения на максимальную и желаемую скорость, а также на максимальное ускорение:

$$v_d = 1 \text{ м/с}, \quad v_{\max} = 1{,}2 \text{ м/с}, \quad a_{\max} = 1 \text{ м/с}^2, \tag{15}$$
$$\omega_{\max} = \frac{\pi}{2} \text{ 1/с}, \quad \varepsilon_{\max} = \frac{\pi}{2} \text{ 1/с}^2.$$

5.2. Примитивы для поворота на месте

Описанный выше подход нельзя использовать для конструирования примитивов, задающих поворот на месте, поскольку линейная скорость в этом случае равна нулю и система не является плоской. Для решения этой проблемы предлагается следующий подход. Пусть примитив поворота задан начальными и конечными условиями:

$$\theta(0) = \theta_0, \quad \frac{d\theta}{dt}(0) = 0, \qquad \theta(t_f) = \theta_f, \quad \frac{d\theta}{dt}(t_f) = 0. \tag{16}$$

Поскольку задано 4 граничных условия, можно выбрать зависимость $\bar{\theta}(t)$ от времени (т.е. примитив движения) как многочлен третьего порядка:

$$\bar{\theta}(t) = \bar{\theta}_0 + \bar{\theta}_1 t + \bar{\theta}_2 t^2 + \bar{\theta}_3 t^3, \quad \frac{d\bar{\theta}}{dt} = \bar{\theta}_1 + 2\bar{\theta}_2 t + 3\bar{\theta}_3 t^2.$$

Подставляя $t = 0$ и $t = t_f$ и учитывая (16), получаем систему линейных алгебраических уравнений четвертого порядка. Решение этой системы дает коэффициенты полинома $\bar{\theta}(t)$. Более того, производная этого многочлена задает программное управление, реализующее заданный примитив поворота:

$$\omega = \frac{d\bar{\theta}}{dt}, \quad v = 0. \tag{17}$$

Поскольку в статье полагается, что начальная и конечная ориентации примитива принимают ограниченный набор значений с шагом 45°, то достаточно задать повороты на месте на $k \cdot 45°$ градусов, где $k = 1, \ldots, 7$. Продолжительность каждого примитива поворота, t_f, выбирается аналогично тому, как выбирается продолжительность примитива перемещения.

6. Планирование траектории с примитивами движений

Задача планирования заключается в построении траектории, которая достигает целевого состояния робота из начального и при том избегает столкновений со статическими и динамическими препятствиями. Такую траекторию можно представить как последовательность действий и записать в виде: $\pi = (a_1, t_1), \ldots, (a_n, t_n)$, где a_i – это действие, а t_i – момент времени, в который это действие должно быть совершено. При этом каждое действие a_i является либо действием *движения*, соответствующим одному из описанных ранее примитивов, либо действием *ожидания.* Продолжительность каждого действия движения заранее известна, в то время как продолжительность действия ожидания может быть произвольной (расчет продолжительности действий ожидания — задача алгоритма планирования).

Напомним, что примитивы движения построены таким образом, что конечные положения, в которые они переводят робота, имеют целочисленные координаты. Используя эту особенность, рабочее пространство $\mathcal{W}$ может быть дискретизировано и представлено в виде графа регулярной декомпозиции (англ. grid), где каждая вершина соответствует некоторой области рабочего пространства [27], см. рис. 1,*в*. Соответственно каждая вершина графа (клетка) может быть либо заблокированной, либо незаблокированной. Ребра графа задаются имплицитно с помощью примитивов движения. Естественно, допускаются перемещения только между незаблокированными вершинами, при этом в процессе движения робот не должен задевать ни одну заблокированную клетку. Столкновения с динамическими препятствиями при нахождении в вершинах графа и движении по ребрам учитываются непосредственно в алгоритме планирования.

Для описания состояния робота при планировании, cfg, учитываются следующие параметры: положение, ориентация и скорость, т.е. $cfg = (x, y, \theta, v)$. Такое состояние называется конфигурацией. Для учета временно́й составляющей и избегания динамических препятствий используется подход безопасно-интервального планирования (SIPP, англ. Safe Interval Path Planning) [4].

В рамках этого подхода для каждой конфигурации, в которой роботом может выполняться действие ожидания, вводятся безопасные и конфликтные интервалы времени. Безопасный интервал для конфигурации cfg — это совокупность тех моментов времени, в течение которых робот может находиться в cfg без столкновения с динамическими препятствиями (о том, как рассчитывается безопасный интервал, будет сказано далее). Для построения траектории осуществляется эвристический поиск в пространстве состояний, состоящих из комбинаций конфигураций робота и соответствующих им безопасных интервалов, т.е. состояние эвристического поиска — это пара $(cfg, [t_1, t_2])$, где cfg – конфигурация, $[t_1, t_2]$ – безопасный интервал для cfg.

Процесс планирования аналогичен принципу работы алгоритма A* [28] и основан на итеративном переборе лучших состояний поиска и генерации их последователей (пока не будет рассмотрено состояние, соответствующее цели робота). Важной особенностью безопасно-интервального планирования явля-

ется процесс генерации потомков для рассматриваемого состояния поиска. В рамках этого процесса SIPP пытается достичь их в минимально возможное время (так называемый принцип "достигай как можно раньше", англ. — earliest arrival time), учитывая при этом возможные конфликты с динамическими препятствиями при исполнении движения. Другими словами, алгоритм либо сразу совершает действие движения, либо если это невозможно из-за динамических препятствий, осуществляет действие ожидания минимально возможной продолжительности (о том, как вычисляется эта продолжительность, будет сказано далее). Известно, что алгоритм SIPP обладает свойствами полноты и оптимальности, т.е. алгоритм гарантирует нахождение решения, если оно существует, более того возвращаемое решение гарантированно является оптимальным (с учетом заданной дискретизации рабочего пространства).

Далее приведем подробное описание реализации основных компонентов алгоритма SIPP, о которых было сказано выше: расчет безопасных интервалов, детектирование конфликтов и расчет минимального времени достижения состояния.

6.1. Модель столкновений

В этой статье предполагается, что столкновение между роботом и динамическим препятствием происходит, если расстояние между их центрами становится меньше, чем $r_a + r_o$, где r_a – это радиус безопасности для робота, r_o – для препятствия. Таким образом, можно считать, что робот и динамические препятствия моделируются дисками (кругами). Также предполагается, что траектория каждого движущегося препятствия известна планировщику и доступна в виде $\{(x_i, y_i, \theta_i, t_i)\}$, т.е. в виде последовательности конфигураций (с фиксированной частотой временной дискретизации траектории).

6.2. Безопасные интервалы

Для своей работы планировщику необходимо для каждой конфигурации (x, y, θ, v), в которой робот может выполнять действие ожидания, вычислять безопасный интервал. В рассматриваемом случае такие действия робот может выполнять только в центрах клеток графа регулярной декомпозиции. Обозначим их как $c = (x_c, y_c)$. Заметим далее, что следствием введенной выше модели столкновений является тот факт, что все конфигурации робота с центром в клетке c, т.е. конфигурации вида $(x_c, y_c, \cdot\,, \cdot)$, имеют одни и те же безопасные интервалы. Таким образом, необходимо рассчитывать безопасные интервалы лишь для клеток графа.

Для расчета безопасных интервалов клеток применяется следующий подход. Последовательно осуществляется перебор всех траекторий динамических препятствий, и для каждой такой траектории, заданной как $\{(x_i, y_i, \theta_i, t_i)\}$, и для каждого t_i вычисляется расстояние от (x_i, y_i) до центров ближайших клеток. Если для какой-либо клетки $c = (x_c, y_c)$ это расстояние меньше, чем $r_a + r_o$, то t_i принадлежит к небезопасному интервалу c, т.е. если

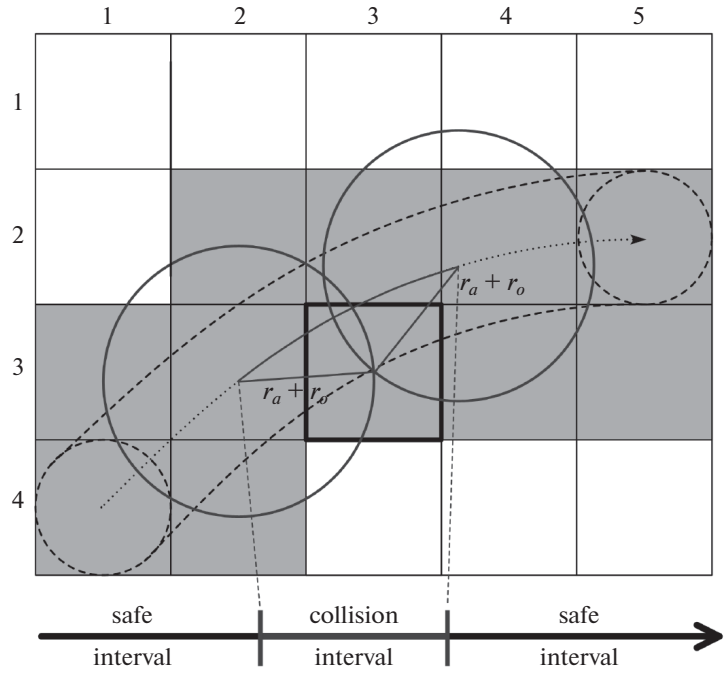


Рис. 2. Определение безопасных интервалов для конфигураций робота с $x = 3$, $y = 3$. Серым показаны клетки, которые задевает динамическое препятствие при следовании по криволинейной траектории, указанной точечной стрелкой (из клетки 1, 4 в клетку 5, 2). Обозначенный сплошной линией сегмент на этой траектории — тот сегмент, когда расстояние от центра препятствия до центра клетки меньше, чем $r_a + r_o$, т.е. границы этого сегмента определяют небезопасный интервал для клетки. Его инверсия задает набор из двух безопасных интервалов.

робот будет занимать эту клетку в момент t_i, — произойдет столкновение с движущимся препятствием. Группировка таких идущих подряд t_i дает полный небезопасный интервал, а инверсия всех небезопасных интервалов для клетки – безопасные. Иллюстрация этого процесса показана на рис. 2.

6.3. Расчет минимального времени достижения состояния поиска

Одним из основных принципов безопасно-интервального планирования как эвристического поиска является принцип достижения каждого создаваемого состояния поиска в минимально возможное время. Этот принцип гарантирует полноту и оптимальность алгоритма. Опишем реализацию принципа в рассматриваемом случае.

Пусть на текущей итерации поиска рассматривается состояние $n = (cfg, [t_1, t_2])$, где $cfg = (x, y, \theta, v)$ – конфигурация робота, $[t_1, t_2]$ – безопасный интервал. Известно, что робот достигает этого состояния в момент времени t_n. Пусть в этом состоянии применимо действие перемещения a, в результате которого робот может быть перемещен в состояние $n' = (cfg', [t'_1, t'_2])$.

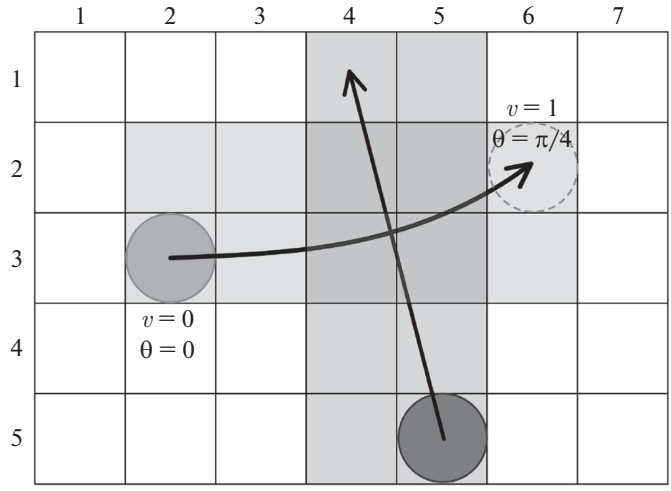


Рис. 3. Определение мест потенциальных столкновений робота с динамическим препятствием. Столкновение возможно только в клетках с координатами 4, 2; 4, 3; 5, 2 и 5, 3.

Согласно принципу "достигай как можно раньше" алгоритм должен применить действие a именно в момент t_n. Однако на практике это может привести к столкновению с динамическим препятствием. Для того чтобы правильно обрабатывать эту ситуацию, алгоритм планирования должен, во-первых, уметь детектировать такие столкновения, во-вторых, вычислять минимально возможное время ожидания в исходной конфигурации cfg и совершать действие a по истечении этого времени.

Для определения столкновений предлагается использовать подход, аналогичный описанному в [29]. В рамках этого подхода считается, что столкновение между движущимся препятствием и роботом происходит тогда, когда существует момент времени, в который и робот, и препятствие пересекают одну и ту же клетку графа регулярной декомпозиции. Иллюстрация этого положения показана на рис. 3. Формально же факт одновременного пересечения клетки агентом и препятствием можно сформулировать следующим образом. Пусть $l_a = [t_a, t'_a]$ и $l_o = [t_o, t'_o]$ – временные интервалы, в течение которых робот/движущееся препятствие пересекают одну и ту же клетку графа регулярной декомпозиции. Столкновение происходит, если $l_a \cap l_o \neq \varnothing$.

Для расчета непосредственно интервалов l_a и l_o используется та же процедура, что и для расчета безопасных интервалов состояний поиска (см. выше). Различие состоит в том, что в данном случае, если идет расчет l_o (l_a), используется формула $r_o + \sqrt{2}/2$ ($r_a + \sqrt{2}/2$), а не $r_a + r_o$. Модификация обусловлена тем, что необходимо вычислить интервал столкновения препятствия (робота) с клеткой безотносительно размера робота (препятствия), поэтому r_a (r_o) заменяется на $\sqrt{2}/2$, что в свою очередь равняется радиусу описанной (вокруг клетки) окружности. Заметим, что такой способ определения временного интервала столкновения клетки и движущегося препятствия (робота) является

консервативным, т.е. гарантируется, что в любой момент вне этого интервала столкновения не будет, но при этом обратное, в общем случае, неверно.

Итак, пусть теперь при попытке совершить действие a без задержки, т.е. в момент времени t_n, алгоритм определяет, что происходит столкновение в определенной клетке графа, т.е. $[t_a, t'_a] \cap [t_o, t'_o] \neq \varnothing$. Тогда перед совершением движения необходимо добавить задержку (действие ожидания), продолжительность которой составляет $\delta = t'_o - t_a$. Такая задержка гарантирует, что столкновения в указанной выше клетке более не будет. Таким образом, процесс вычисления минимально возможного времени ожидания перед совершением действия a состоит в последовательном переборе клеток, которые задевает примитив движения, соответствующий действию a, и применении в отношении каждой из таких клеток процедуры расчета задержки, описанной выше.

7. Следование по траектории

Построенные примитивы являются непрерывными по переменным x, y, θ, v, но разрывны по угловой скорости ω. Таким образом, при переключении примитивов возникает переходный процесс, и полученные ранее программные управления при повороте и перемещении должны быть заменены на обратную связь, которая компенсирует отклонения от желаемой траектории.

Для реализации следования по примитивам перемещения предлагается использование следующей обратной связи, заменяющей (14):

$$\psi_1 = \ddot{x}^* + (\lambda_1 + \lambda_2)(\dot{x} - \dot{x}^*) - \lambda_1\lambda_2(x - x^*), \tag{18}$$

$$\psi_2 = \ddot{y}^* + (\lambda_1 + \lambda_2)(\dot{y} - \dot{y}^*) - \lambda_1\lambda_2(y - y^*). \tag{19}$$

Здесь x^*, y^* – желаемые траектории для x, y соответственно, а $\lambda_1 < 0$, $\lambda_2 < 0$ – параметры, влияющие на длительность переходного процесса, которые подбираются эмпирически. Чем больше они по модулю, тем быстрее закон управления способен компенсировать отклонение, но это может потребовать большого расхода управления, и ограничения на управление могут быть нарушены. В численных экспериментах были использованы следующие значения этих параметров: $\lambda_1 = -4$, $\lambda_2 = -5$. Дополнительно при околонулевых скоростях ψ_i умножается на положительный коэффициент, который ограничивает рост угловой скорости и углового ускорения. Этот коэффициент уменьшается при увеличении линейной скорости v.

Для реализации примитивов поворота на месте программное управление (17) заменяется на следующую обратную связь:

$$v = 0, \quad \omega = \dot{\theta}^* + \lambda_1(\theta - \theta^*), \tag{20}$$

где θ^* – планируемая зависимость изменения значения θ от времени.

При реализации целой траектории происходит переключение между двумя типами регуляторов в зависимости от типа движения, которое по плану исполняет робот в текущий момент. Полученный таким образом закон

управления будем называть плоскостной динамической обратной связью по состоянию (flatness-based dynamic state feedback control, FBDF).

8. Численные эксперименты

Для оценки предлагаемых в статье методов и алгоритмов планирования и управления была проведена серия численных экспериментов. Сначала было произведено сравнение разработанного регулятора FDBF с наиболее часто используемым в робототехнике регулятором MPC на отдельных примитивах. Далее был выполнен анализ того, как быстро работает предлагаемый планировщик и как меняется скорость его работы при увеличении числа динамических препятствий. Далее произведено сравнение работы двух рассматриваемых законов управления FDBF и MPC при следовании по построенным траекториям.

8.1. Сравнение FDBF и MPC на отдельных маневрах

Численные эксперименты по сравнению регуляторов на отдельных примитивах производились на ПК с процессором Intel Core i5-8250U и 8 Гб ОЗУ под управлением ОС Windows 10.

Оба закона управления были численно реализованы в MATLAB R2020a (для реализации нелинейного MPC была использована функция `nlmpc`). Для выбора наилучших параметров MPC была проведена серия предварительных численных экспериментов, и в результате получены следующие параметры: интервал дискретизации (sample time) $Ts = 0{,}1$, горизонт предсказания (prediction horizon) $p = 10$, горизонт управления (control horizon) $c = 5$, весовая матрица коэффициентов отклонения по состоянию $Q_m = \mathrm{diag}(10, 10, 1, 1)$ для примитивов перемещения и $Q_r = \mathrm{diag}(10, 10, 1)$ для примитивов поворота на месте, весовая матрица коэффициентов перед управлением $R = \mathrm{diag}(1, 1, 1)$. Для оценки эффективности законов управления использовался квадратичный критерий оценки стоимости с весовыми матрицами R, Q_m или Q_r. В частности, для оценки эффективности управления для примитивов перемещения использовался критерий

$$Cost = \sum_{\alpha=1}^{N} Q_m(\tilde{\mathbf{x}}_\alpha - \tilde{\mathbf{x}}^*_\alpha)^2 + \sum_{\alpha=1}^{N} R\psi^2_\alpha,$$

где $\tilde{\mathbf{x}}_\alpha$ — расширенный вектор состояния на шаге t_α, $\tilde{\mathbf{x}}^*_\alpha$ — значение планируемого вектора состояния в момент времени t_α, ψ_α — вектор управления в расширенной системе в момент времени t_α, N — количество шагов дискретизации по времени. Для оценки эффективности исполнения примитивов поворота на месте использовался аналогичный критерий.

Для сравнения были выбраны четыре различных примитива перемещения и четыре различных примитива поворота на месте. Остальные типы примитивов могут быть получены аффинными преобразованиями восьми указанных примитивов. Для сравнения не использовались примитивы, которые

Таблица 1. Эффективность законов управления на отдельных маневрах. В первых двух столбцах указаны примитивы движения (каждая строка — отдельный примитив). В следующих столбцах — показатели качества работы регуляторов FBDF и MPC при отработке этих примитивов

Примитивы		FBDF		MPC	
Начало	Цель	Стоимость	Время, мс	Стоимость	Время, мс
$(0, 0, 0°, 1)$	$(1, 0, 0°, 1)$	0,141	5	0,093	586
$(0, 0, 0°, 1)$	$(4, 1, 45°, 1)$	0,295	15	0,274	2 320
$(0, 0, 45°, 1)$	$(1, 1, 45°, 1)$	0,094	5	0,073	845
$(0, 0, 45°, 1)$	$(2, 4, 90°, 1)$	0,186	17	0,177	2 385
$(0, 0, 0°, 0)$	$(0, 0, 45°, 0)$	0,322	8	0,313	799
$(0, 0, 0°, 0)$	$(0, 0, 90°, 0)$	0,911	10	0,897	1 111
$(0, 0, 0°, 0)$	$(0, 0, 135°, 0)$	1,673	12	1,657	1 336
$(0, 0, 0°, 0)$	$(0, 0, 180°, 0)$	2,572	17	2,555	2 545

включают разгон от нулевой скорости или торможение до полной остановки, поскольку в этом случае оба метода требуют переключения методов управления, что может исказить представление об их эффективности.

Для каждого маневра оценка качества закона управления проводилась с использованием подходящего квадратичного критерия (чем меньше значение функции стоимости, тем лучше), а также времени расчета управления для реализации заданного примитива (чем меньше время расчета, тем лучше). Результаты приведены в табл. 1. Видно, что предлагаемый закон управления рассчитывает управление на два порядка быстрее, чем MPC, при сравнимом качестве управления, несмотря на то что FBDF не оптимизирует функцию стоимости в отличие от MPC.

8.2. Планирование и следование вдоль траектории

Для проведения экспериментальных исследований использовались два графа регулярной декомпозиции размером 64×64 клеток, в которых каждая клетка соответствует области площадью 1 м^2. Первая карта, `empty64x64`, не содержит статических препятствий. Вторая карта, `rooms64x64`, представляет собой набор комнат, соединенных узкими проходами шириной 1 м (эта карта была взята из открытой коллекции карт, используемых для тестирования алгоритмов планирования [30]).

Робот и динамические препятствия моделировались диском радиуса 0,5 м. Число динамических препятствий варьировалось в диапазоне от 100 до 250. Для каждого динамического препятствия была случайным образом построена траектория, состоящая от последовательности прямолинейных движений. Для каждой карты и количества динамических препятствий было подготовлено по 100 заданий, различающихся траекториями динамических препятствий, а также их начальными и конечными положениями. Для каждого задания начальное положение робота находилось в левой верхней клетке, а целевое — в правой нижней (т.е. роботу необходимо было пересечь всю карту, чтобы достигнуть цели).

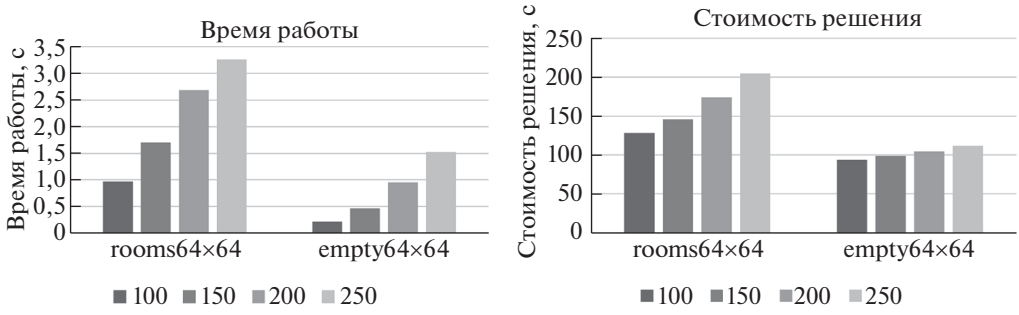


Рис. 4. Среднее время работы алгоритма планирования и средняя стоимость отыскиваемых решений (продолжительность спланированных траекторий) в зависимости от протестированной карты и числа динамических препятствий.

Численные эксперименты по планированию траектории производились на ПК с процессором Intel Core i5-10600k и 32 Гб ОЗУ под управлением ОС Windows 10.

Графики на рис. 4 демонстрируют среднее время работы алгоритма и среднюю стоимость найденных траекторий. В данном случае под стоимостью понимается время, необходимое для достижения целевого положения. Как можно заметить, с увеличением числа динамических препятствий увеличивается и время работы алгоритма. Однако рост — не экспоненциальный, а полиномиальный (близкий к линейному). Отдельно заметим, что на пустой карте при числе динамических препятствий, равном 100, планирование осуществляется за доли секунды (порядка 0,15 с). Также видно, что стоимость решения увеличивается с ростом числа динамических препятствий незначительно (особенно в случае пустой карты).

После того как траектории были построены, были использованы регуляторы MPC и FBDF для следования по ним. Для оценки точности следования по траектории было вычислено расстояние между запланированным и реальным положением робота в каждый момент времени с шагом 10 мс. График, показывающий отклонение на одной из построенных траекторий, показан на рис. 5. Здесь слева — отклонения положения робота при следовании по траектории от запланированной траектории, справа — внешний вид самой траектории (робот движется из левого верхнего угла в правый нижний, динамические препятствия не изображены). Заметные скачки у регулятора FBDF наблюдаются в тех случаях, когда происходит переключение между примитивами вращения и перемещения (такие места на траектории отмечены кругами на правой части рисунка), при этом регулятор способен быстро компенсировать эти отклонения. В целом, как видно на графике, отклонение от траектории для регулятора FBDF ниже, чем для MPC.

Для сравнения регуляторов на всех спланированных траекториях были рассчитаны значения среднеквадратичной ошибки (RMSE) и максимального отклонения от запланированной траектории. Соответствующие результаты представлены в табл. 2. Как показали результаты экспериментов, регулятор

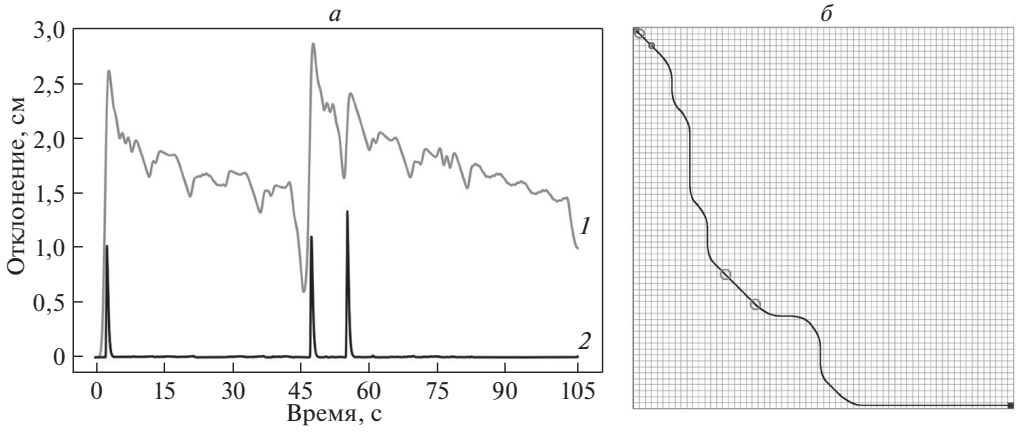


Рис. 5. Сравнение регуляторов FBDF и MPC при следовании по отдельной траектории. *а* — график ошибки следования. *1* — MPC, *2* — FBDF. По оси абсцисс — время (в секундах), по оси ординат — отклонение положения робота от запланированного в данный момент времени (в сантиметрах); *б* — сама (запланированная) траектория. Окружностями отмечены места стыковки маневров, при которых происходят скачки управления регулятора FBDF (пики на графике *а*).

FBDF имеет почти на порядок меньшее значение среднеквадратичной ошибки и вдвое меньшее максимальное отклонение. С увеличением числа динамических препятствий значения среднеквадратичной ошибки растут для обоих регуляторов. Этот рост обусловлен более длинными траекториями, а также наличием большего числа действий ожидания и вращений, что вынуждает регулятор чаще переключаться. Полученные траектории были также проверены на наличие конфликтов с динамическими препятствиями. В то время как спланированные траектории гарантированно являются неконфликтными, из-за наличия отклонений в процессе следования они все же могут возникать. В рассмотренных заданиях конфликтными оказались 15 траекторий, полученных с помощью регулятора MPC, и всего одна, полученная с помо-

Таблица 2. Среднеквадратичные ошибки (RMSE) и максимальные отклонения (MaxDev) при следовании по спланированным траекториям, усредненные по всем заданиям для фиксированного числа динамических препятствий. Единица измерения – сантиметр. Первый столбец — число динамических препятствий, последующие — количественные показатели качества следования по траекториям с заданными числом динамических препятствий для регуляторов MPC и FBDF

	MPC		FBDF	
Obs	RMSE, см	MaxDev, см	RMSE, см	MaxDev, см
100	1,012	3,741	0,104	1,408
150	1,093	4,172	0,120	1,407
200	1,143	4,286	0,130	1,409
250	1,187	4,018	0,138	1,410

щью FBDF. При этом во всех случаях расстояние между роботом и динамическим препятствием не превышало величину $r_a + r_o$ более чем на 1 см, где r_a, r_o – радиус робота/препятствия (равный 50 см). Таким образом, если для алгоритма планирования траектории искусственно увеличить радиус робота всего на 1 см (создать буфер безопасности), то следование по спланированным траекториям будет проходить без столкновений.

9. Заключение

В статье предложена комбинация методов планирования и управления для обеспечения безопасного движения мобильного робота с дифференциальным приводом в среде со статическими и динамическими препятствиями. В части управления предложено использование метода накрытий для плоских систем, который позволяет решать задачи построения кинематически и динамически допустимых примитивов движения, а также следования вдоль композиции примитивов (траектории). В части построения траектории предложено использование алгоритма безопасно-интервального планирования, в рамках которого построенные примитивы движения используются для конструирования траектории, избегающей столкновения как со статическими, так и с динамическими препятствиями.

В данной статье задача планирования в динамической модели мира рассматривалась отдельно от задачи построения таковой. Предполагалось, что имеющаяся модель полна и точна, т.е. точно известно расположение всех статических препятствий на карте и также точно известны (спрогнозированы) траектории движения динамических препятствий. Очевидно, что на практике получение подобных точных моделей весьма затруднительно, если вообще возможно. Таким образом, перспективным направлением дальнейших работ является исследование методов планирования и исполнения плана в условиях неточной модели, например, когда траектории движения динамических препятствий заданы вероятностными распределениями. Еще одним перспективным направлением видится разработка метода автоматической генерации произвольных примитивов движения и интеграция этого метода с алгоритмами планирования траектории, основанными на принципах случайного сэмплирования. Также представляет особый интерес проведение экспериментальных исследований указанных методов и алгоритмов на реальном роботе.

СПИСОК ЛИТЕРАТУРЫ

1. *McGuire K.N., de Croon G.C.H.E., Tuyls K.A.* Comparative Study of Bug Algorithms for Robot Navigation // Robotics and Autonomous Syst. 2019. V. 121. P. 103261.
2. *Berg J., et al.* Reciprocal n-body Collision Avoidance // Robotics Research. Berlin–Heidelberg: Springer, 2011. P. 3–19.
3. *Koenig S., Likhachev M.* D* Lite // 18th National Conf. on Artificial Intelligence. 2002. P. 476–483.

4. *Phillips M., Likhachev M.* SIPP: Safe Interval Path Planning for Dynamic Environments // 2011 IEEE Int. Conf. on Robotics and Automation. IEEE, 2011. P. 5628–5635.
5. *Mayne D.Q., et al.* Constrained Model Predictive Control: Stability and Optimality // Automatica. 2000. V. 36. No. 6. P. 789–814.
6. *Бочаров А.В., Вербовецкий А. М.* Симметрии и законы сохранения математической физики // М.: Факториал, 2005.
7. *Likhachev M., Ferguson D.* Planning Long Dynamically Feasible Maneuvers for Autonomous Vehicles // Int. J. of Robotics Research. 2009. V. 28. No. 8. P. 933–945.
8. *Sakcak B., et al.* Sampling-Based Optimal Kinodynamic Planning with Motion Primitives // Autonomous Robots. 2019. V. 43. No. 7. P. 1715–1732.
9. *Pivtoraiko M., Kelly A.* Generating Near Minimal Spanning Control Sets for Constrained Motion Planning in Discrete State Spaces // 2005 IEEE/RSJ Int. Conf. on Intelligent Robots and Syst. IEEE, 2005. P. 3231–3237.
10. *hwan Jeon J., Karaman S., Frazzoli E.* Anytime Computation of Time-Optimal Offroad Vehicle Maneuvers Using the RRT // 2011 50th IEEE Conf. on Decision and Control and Eur. Control Conf. IEEE, 2011. P. 3276–3282.
11. *Webb D.J., Van Den Berg J.* Kinodynamic RRT*: Asymptotically Optimal Motion Planning for Robots with Linear Dynamics // 2013 IEEE Int. Conf. on Robotics and Automation. IEEE, 2013. P. 5054–5061.
12. *Flores M.E., Milam M.B.* Trajectory Generation for Differentially Flat Systems via NURBS Basis Functions with Obstacle Avoidance // 2006 Amer. Control Conf. IEEE, 2006. P. 5769–5775.
13. *Rufli M., Ferguson D., Siegwart R.* Smooth Path Planning in Constrained Environments // 2009 IEEE Int. Conf. on Robotics and Automation. IEEE, 2009. P. 3780–3785.
14. *Silver D.* Cooperative Pathfinding // Proc. AAAI Conf. on Artificial Intelligence and Interactive Digital Entertainment. 2005. V. 1. No. 1. P. 117–122.
15. *Isidori A.* Nonlinear Control Systems. Berlin: Springer, 1995.
16. *Khalil H.K.* Nonlinear Systems (3rd ed.). Upper Saddle River. NJ: Prentice Hall, 2002.
17. *Utkin V.I.* Sliding Mode Control Design Principles and Applications to Electric Drives // IEEE Trans. on Industrial Electronics. 1993. V. 40. No. 1. P. 23–36.
18. *Yu S., et al.* MPC for Path Following Problems of Wheeled Mobile Robots // IFAC-PapersOnLine. 2018. V. 51. Np. 20. P. 247–252.
19. *Fliess M., et al.* Flatness and Defect of Non-Linear Systems: Introductory Theory and Examples // Int. J. Control. 1995. V. 61. No. 6. P. 1327–1361.
20. *Bascetta L., Arrieta I.M., Prandini M.* Flat-RRT*: A Sampling-Based Optimal Trajectory Planner for Differentially Flat Vehicles with Constrained Dynamics // IFAC-PapersOnLine. 2017. V. 50. No. 1. P. 6965–6970.
21. *Sahoo S.R., Chiddarwar S.S.* Mobile Robot Control Using Bond Graph and Flatness Based Approach // Procedia Computer Science. 2018. V. 133. P. 213–221.
22. *Четвериков В.Н.* Плоскостность динамически линеаризуемых систем // Диффер. уравнения. 2004. Т. 40. № 12. С. 1665–1674.
23. *Белинская Ю.С., Четвериков В.Н.* Метод накрытий для терминального управления с учетом ограничений // Диффер. уравнения. 2014. Т. 50. № 12. С. 1629–1639.

24. *Tang C.P.* Differential Flatness-Based Kinematic and Dynamic Control of A Differentially Driven Wheeled Mobile Robot // 2009 IEEE Int. Conf. on Robotics and Biomimetics (ROBIO). IEEE, 2009. P. 2267–2272.

25. *Andersson O., et al.* Receding-Horizon Lattice-Based Motion Planning with Dynamic Obstacle Avoidance // 2018 IEEE Conf. on Decision and Control (CDC). IEEE, 2018. P. 4467–4474.

26. *Belinskaya Y.S., Chetverikov V.N.* Covering Method for Point-To-Point Control of Constrained Flat Systems // IFAC-PapersOnLine. 2015. V. 48. No. 11. P. 924–929.

27. *Yap P.* Grid-Based Path-Finding // Conf. of the Canadian Society for Computational Studies of Intelligence. Berlin–Heidelberg: Springer, 2002. P. 44–55.

28. *Hart P.E., Nilsson N.J., Raphael B.* A Formal Basis for The Heuristic Determination of Minimum Cost Paths // IEEE Trans. Syst. Sci. and Cybernetics. 1968. V. 4. No. 2. P. 100–107.

29. *Cohen L., et al.* Optimal and Bounded-Suboptimal Multi-Agent Motion Planning // 12th Annual Sympos. on Combinatorial Search. 2019. P. 44–51.

30. *Sturtevant N.R.* Benchmarks for Grid-Based Pathfinding // IEEE Trans. Computational Intelligence and AI in Games. 2012. V. 4. No. 2. P. 144–148.

Статья представлена к публикации членом редколлегии О.П. Кузнецовым.

Поступила в редакцию 07.12.2021

После доработки 11.01.2022

Принята к публикации 26.01.2022

© 2022 г. О.П. КУЗНЕЦОВ, д-р техн. наук (olpkuz@yandex.ru)
(Институт проблем управления им. В.А. Трапезникова РАН, Москва)

ОБ УСЛОВИЯХ ПРОХОЖДЕНИЯ СИГНАЛА ЧЕРЕЗ ЦЕПЬ АСИНХРОННЫХ ПОРОГОВЫХ ЭЛЕМЕНТОВ[1]

Исследуется процесс распространения сигнала в цепи последовательно соединенных асинхронных пороговых элементов. Асинхронность элементов проявляется в том, что они могут иметь различные длительности переключения из пассивного состояния в активное и обратно. Элементы реактивны, т.е. возбуждаются в результате внешних воздействий, и становятся пассивными, когда внешние воздействия отсутствуют. Показано, что при различных соотношениях длительностей переходных процессов включения и отключения элементов длительность сигнала, проходящего по цепи, может сохраняться, увеличиваться или уменьшаться. В последнем случае сигнал через достаточно длинную цепь не проходит. Сформулированы условия прохождения сигнала через цепь.

Ключевые слова: асинхронный пороговый элемент, прохождение сигнала, переходные процессы.

DOI: 10.31857/S0005231022060095, **EDN:** ADFCXE

1. Введение

Настоящая статья продолжает исследования сетей из асинхронных пороговых элементов, начатые в [1], где элементами сети являлись биологические нейроны, а связи соответствовали химическим взаимодействиям между ними. В [2] предложено обобщение модели [1]: в нем узлы асинхронной сети рассматриваются как абстрактные пороговые элементы, а химические взаимодействия заменены многосортными сигналами. В моделях [1, 2] асинхронность возникает из-за того, что разные элементы имеют разные длительности переходных процессов, т.е. переключений из активного состояния в пассивное и обратно. Это приводит к конкуренции между элементами, хорошо известной в дискретной схемотехнике под названием состязаний (см., например, [3, 4]).

В различных областях важны разные аспекты асинхронности. В дискретной схемотехнике асинхронность порождает состязания, нарушающие нормальную работу схем. Поэтому в ней преобладает принудительная синхронизация переключений элементов. В управлении бизнесом [5] и программировании [6] асинхронный подход повышает эффективность, экономя время: основной процесс управления запускает параллельные процессы и движется

[1] Работа выполнена при частичной финансовой поддержке Российского фонда фундаментальных исследований (проект № 20-07-00190).

дальше, не дожидаясь их окончания. В нейробиологии активно обсуждаются асинхронные процессы распространения сигналов и их синхронизация в нервных системах [7, 8], а также использование асинхронного программирования для их моделирования [9–11].

В данной статье исследуется еще один аспект асинхронности: влияние соотношения длительностей включения и отключения элементов на прохождение сигнала по сети, даже когда элементы одинаковы, а параллельные процессы отсутствуют.

Результаты, изложенные в разделах 3, 4, опубликованы в [12] без доказательств. Ниже публикуются подробные доказательства теорем, приведенных в [12]; в разделе 5 впервые излагается новый метод вычисления протоколов прохождения сигнала по цепи.

2. Основные определения и постановка задачи

Асинхронный пороговый элемент[2] — это элемент с несколькими входами и одним выходом, который может находиться в двух состояниях – активном и пассивном. В активном состоянии элемент N_i генерирует сигнал мощности d_i.

Состояние $y_i(t)$ элемента N_i определяется значением его *потенциала* $U_i(t)$, изменяющегося в интервале $U_{i0} \le U_i(t) \le U_{\max}$, и *порогом* P_i, лежащим в том же интервале:

$$y_i(t) = \begin{cases} 1, & \text{если } U_i(t) \ge P_i; \\ 0 & \text{иначе.} \end{cases} \tag{1}$$

Помимо потенциала и порога, асинхронный элемент обладает другими параметрами — весами w_{ij} входов (i – номер элемента, j – номер входа) и двумя эндогенными скоростями изменения потенциала (v^0_{ien} для пассивного и v^1_{ien} для активного состояния). По определению $v^1_{\text{ien}} > v^0_{\text{ien}}$. Эндогенные скорости не зависят от внешних воздействий; это константы, характеризующие конкретный элемент.

Асинхронный элемент называется реактивным, если обе его эндогенные скорости отрицательны. Он активируется только при наличии достаточно сильных внешних воздействий; при их отсутствии потенциал реактивного элемента стремится к U_{i0}.

Элемент может интерпретироваться как логический элемент электронной схемы или электромагнитное реле, как нейрон в искусственной или биологической нейронной сети, или как агент в моделях социального поведения.

В дальнейшем будут рассматриваться *однородные асинхронные цепи* реактивных элементов, т.е. сети из реактивных элементов $N_1, \ldots, N_n$, в которых 1) выход элемента N_i соединен с единственным входом элемента N_{i+1},

[2] Асинхронные пороговые элементы впервые были рассмотрены в [1, 2]. В [1] они назывались нейронами, а в [2] — агентами.

Таблица 1

	P	$U_{\max}$	U_0	v^0_{en}	v^1_{en}	d	w
N_0						1,0	
N_i	0,5	0,9	0	−0,8	−0,5	1,0	1,0

$i = 1, \ldots, n$, и других связей нет[3]; 2) вход элемента N_1 и выход элемента N_n — внешние для цепи; 3) значения параметров для любого элемента цепи одинаковы, поэтому можно говорить о параметрах цепи. Пример таблицы, задающей параметры цепи, приведен в табл. 1. N_0 – внешний источник возбуждения.

Входное воздействие будем называть сигналом. Сигнал проходит через элемент, если через некоторое время после поступления сигнала на его вход элемент активируется и генерирует выходной сигнал. Сигнал поступает на вход N_i, пока активен N_{i-1}. Сила влияния сигнала характеризуется экзогенной скоростью $s_i(t)$ изменения потенциала N_i, которая пропорциональна мощности d входного сигнала и весу w входа: $s_i(t) = kwdy_{i-1}(t)$. В дальнейшем полагаем $k = 1$. Суммарная скорость v_i изменения потенциала элемента N_i является суммой экзогенной и эндогенной скоростей

$$v_i(t) = s_i(t) + v^0_{\text{en}} = wdy_{i-1}(t) + v^0_{\text{en}}, \tag{2}$$

если элемент пассивен (потенциал ниже порога), и

$$v_i(t) = s_i(t) + v^0_{\text{en}} = wdy_{i-1}(t) + v^0_{\text{en}}, \tag{3}$$

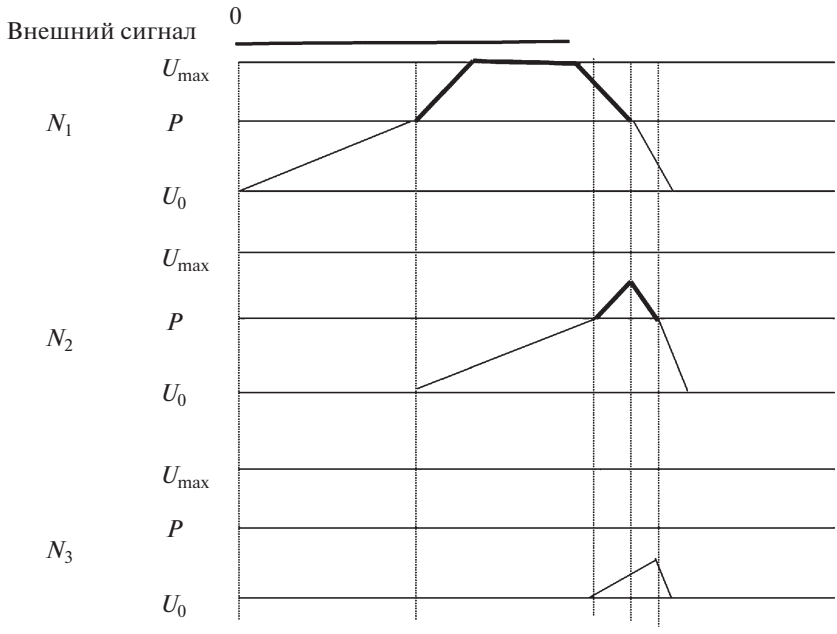
если элемент активен (потенциал выше порога).

Знак скорости означает направление изменения потенциала.

Событием в момент t называется изменение состояния любого элемента цепи, а также достижение U_{i0} или $U_{\max}$ в момент t. Событиям соответствуют точки на шкале непрерывного времени, разбивающие шкалу на отрезки — такты. Границы тактов нумеруются натуральными числами $0, 1, 2, \ldots$ и называются дискретными моментами времени. Такт t — это полуинтервал $[t, t + 1]$, длительность которого обозначается как $\tau(t)$. Последовательность событий на непрерывной временной шкале называется *протоколом* функционирования цепи. *Внешним состоянием цепи* называется вектор $Y(t) = (y_1(t), \ldots, y_n(t))$; *внутренним состоянием* — вектор значений потенциалов элементов $U(t) = (U_1(t), \ldots, U_n(t))$.

Функционирование сети из асинхронных элементов, т.е. его протокол, в общем случае вычисляется алгоритмом, описанным в [2]. Для случая асинхронных цепей этот алгоритм упрощается, поскольку сигналы односортны,

[3] В [1, 2] рассматриваются многосортные широковещательные связи: сигнал определенного сорта воспринимается всеми элементами, у которых есть входы, чувствительные к этому сорту. В данной работе все сигналы обычные односортные, а связи локальные, т.е. соединяющие ровно два элемента.



Рисунок

все связи имеют стандартный вид (N_i, N_{i+1}), а изменения потенциалов вычисляются по более простым формулам (2) и (3). Этот алгоритм приведен в [12]. В разделе 5 будет приведен еще более простой алгоритм, не требующий таблиц переходов и связанных с ними понятий, введенных в [2, 12].

В цепи из реактивных элементов нетривиальное поведение может возникнуть только в результате подачи внешнего сигнала на вход N_1 и последовательной активации следующих элементов. *Главная задача, которая исследуется в данной работе*: *при каких условиях сигнал пройдет через всю цепь, т.е. через некоторое время после подачи сигнала на вход* N_1 *появится сигнал на выходе* N_n?

Необходимое условие прохождения сигнала через элемент заключается в том, чтобы сила внешнего воздействия превосходила отрицательную эндогенную скорость, т.е. чтобы суммарная скорость была положительной. Однако этого условия недостаточно; нужно, чтобы сигнал длился достаточно долго, что видно из следующего примера.

Пример 1. На рисунке представлена временная диаграмма функционирования цепи, параметры которой заданы табл. 1, при длительности входного сигнала $Q_0 = 4{,}5$. Жирными линиями обозначены периоды активности элементов. Видно, что сигнал не проходит уже через третий элемент: длительности активности N_2 недостаточно, чтобы потенциал N_3 достиг порога и N_3 включился. Вертикальные линии соответствуют совпадающим событиям: например, включение N_1 означает начало генерации сигнала, входного для N_2, и, соответственно, начало роста потенциала N_2.

Рассмотрим структуру процесса функционирования элемента N_i. Цикл включения-отключения N_i, начинающийся с подачи на его вход входного сигнала, называется *полным*, если он состоит из последовательности пяти фаз (см. график N_1 на рисунке): 1) зарядки (роста потенциала от U_0 до P), 2) роста потенциала от P до $U_{\max}$; 3) пребывания потенциала в точке $U_{\max}$; 4) разрядки (уменьшения потенциала от $U_{\max}$ до P); 5) падения потенциала до U_0.

Обозначим длительности этих фаз через q_{1i}, q_{2i}, q_{3i}, q_{4i}, q_{5i}, где $i = 1, \ldots,$ n – номер элемента. Фазы 1 и 2 называются полными в данном цикле, если за ними следуют фазы 2 и 3 соответственно; фазы 4 и 5 полны, если им предшествуют фазы 3 и 4 соответственно. Если цепь однородна, а фазы 1, 2, 4, 5 полны, то их длительности выражаются формулами (4)–(7), содержащими только параметры, т.е. не зависят от конкретных элементов и времени. Поэтому индексы номеров элементов для полных фаз опущены. Длительность фазы 3 зависит от длительности входного сигнала и для разных элементов может быть различной. Так как скорости v^0_{en} и v^1_{en} отрицательны, для наглядности вместо v^0_{en} и v^1_{en} будем писать $-|v^0_{\text{en}}|$ и $-|v^1_{\text{en}}|$. Полагаем, что $U_0 = 0$ и $U_{\max} - P = lP$.

$$q_1 = \frac{P - U_0}{wd + v^0_{\text{en}}} = \frac{P}{wd - |v^0_{\text{en}}|}, \tag{4}$$

$$q_2 = \frac{U_{\max} - P}{wd + v^1_{\text{en}}} = \frac{lP}{wd - |v^1_{\text{en}}|}, \tag{5}$$

$$q_4 = \frac{U_{\max} - P}{|v^1_{\text{en}}|} = \frac{lP}{|v^1_{\text{en}}|}, \tag{6}$$

$$q_5 = \frac{P - U_0}{|v^0_{\text{en}}|} = \frac{P}{|v^0_{\text{en}}|}. \tag{7}$$

Для N_1 формулы (4) и (6) верны, только если мощность d_0 внешнего сигнала равна d. Однако именно это предположение $d_0 = d$ будет использоваться в дальнейшем. Оно не снижает общности исследования, поскольку начиная с выхода первого элемента по цепи в любом случае будет распространяться сигнал мощности d. Поэтому, если входной сигнал имеет другую мощность, полученные результаты будут верны для цепи, где N_1 является внешним источником, а сама цепь начинается с элемента N_1.

Дополнительные обозначения: Q_0 – длительность внешнего сигнала; Q_i, $i = 1, \ldots, n$, – длительность активности элемента N_i и, соответственно, длительность его выходного сигнала, t_{ji} – момент начала фазы j элемента N_i, t_{zi} – момент падения U_i до U_0. Для внешнего сигнала обозначим через $t_{00} =$ $= 0$ – момент его начала, t_{z0} – момент его окончания; t_{ji}, t_{00}, t_{z0} – точки на временной шкале, отмечающие моменты наступления событий; последовательность этих моментов на временной шкале — это протокол функционирования цепи. Некоторые события неизбежно совпадают: например, $t_{z0} = t_{41}$.

Для дальнейшего важно простое равенство:

$$Q_i = t_{5i} - q_{2i}. \tag{8}$$

Выходные сигналы элементов цепи (кроме элемента N_n) будем называть *внутренними.* Сигнал называется *длинным*, если $Q_i \geq q_1 + q_2$, *коротким*, если $q_1 < Q_i < q_1 + q_2$, *недостаточным*, если $Q_i \leq q_1$. Заметим, что а) такая классификация сигналов зависит от параметров сети, так как величины q_1, q_2, q_4 зависят от этих параметров; б) она относится не только ко внешнему сигналу, но и к внутренним сигналам цепи; в) если сигнал короткий, то длительности фаз 2 и 4 перестают быть постоянными; их заменяют величины q_{2i}, q_{4i}, которые зависят от длины входного сигнала.

Из диаграммы на рисунке видно, что внешний сигнал — длинный, входной сигнал для N_2 (длительность которого равна Q_1) — короткий, а входной сигнал для N_3 (его длительность равна Q_2) — недостаточный.

Справедливо следующее утверждение.

Лемма 1. Если $Q_i > q_1$, $i = 0, 1, \ldots, n-1$, то $Q_{i+1} = Q_i - q_1 + q_{4,i+1}$.

Действительно, согласно (8) $Q_i = t_{5i} - q_{2i}$; активация N_{n+1} наступает в момент $t_{2i} + q_1$ и заканчивается в момент $t_{5i} + q_{4,i+1}$. Отсюда

$$Q_{i+1} = t_{5i} + q_{4,i+1} - (t_{2i} + q_1) = Q_i - q_1 + q_{4,i+1}. \tag{9}$$

Если входной сигнал для N_{i+1} – длинный, то $q_{4,i+1} = q_4$.

Сигнал, при котором $q_1 = q_{41}$, будем называть *равновесным.* Как будет видно из дальнейшего, равновесным может быть как длинный (если $q_{4,i+1} = q_4$), так и короткий сигнал.

3. Случай длинного внешнего сигнала: $Q_0 > q_1 + q_2$.

В этом случае

$$Q_0 = q_1 + q_2 + \alpha_0, \quad \alpha_0 > 0, \tag{10}$$

и N_1 с момента $t = q_1 + q_2$ переходит в фазу 3; ее длительность $q_{13} = \alpha_0$. При этом фазы 1, 2, 4 элемента N_1 полны, и их длительности вычисляются по формулам (4)–(6). Определение длительностей этих фаз для следующих элементов цепи является одной из задач данной работы. Напомним, что Q_i – это длительность активности элемента N_i и, соответственно, длительность входного сигнала для N_{i+1}.

Теорема 1. Пусть на вход цепи с заданными параметрами подан сигнал длительности $Q_0 = q_1 + q_2 + \alpha_0$, $\alpha_0 > 0$. Тогда:

1. *Если $q_1 = q_4$, то $Q_{i+1} = Q_i$, $i = 1, \ldots, n$, т.е. сигнал пройдет по всей цепи с сохранением длительности.*

2. *Если $q_4 = q_1 + \beta$, $\beta > 0$, то при каждом прохождении через элемент сигнал удлиняется на одну и ту же величину β. Поэтому сигнал пройдет через цепь любой длины, причем приращение длительности сигнала на выходе цепи пропорционально длине цепи.*

3. *Если* $q_4 = q_1 - \gamma$, $\gamma > 0$, *то существует такое натуральное число* z_1, *зависящее от длительности входного сигнала, что, если длина цепи превосходит* z_1, *то сигнал через эту цепь не пройдет. Конкретнее: найдется такое натуральное число* k, *что если цепь содержит не менее* $k+1$ *элементов, то выходной сигнал* $(k+1)$*-го элемента будет либо недостаточным, либо коротким. В первом случае* $z_1 = k+1$; *во втором случае* $z_1 = k+l+1$, *где* $l > 0$.

Доказательство. 1. Если $q_1 = q_4$, то равенство $Q_{i+1} = Q_i$, $i = 1, \ldots, n$ следует из леммы 1.

2. Если $q_4 = q_1 + \beta$, то из (9) имеем $Q_{i+1} = Q_i + \beta$, т.е. при каждом прохождении через элемент сигнал удлиняется на β. Следовательно, приращение длительности сигнала на выходе пропорционально длине цепи.

3. Из (9) следует, что

$$Q_{i+1} = Q_i - q_1 + q_4 = Q_i - q_1 + q_1 + \gamma = Q_i + \gamma, \tag{11}$$

т.е. при каждом прохождении через элемент выходной сигнал укорачивается на одну и ту же величину γ. Однако это верно только до тех пор, пока выходной сигнал N_{i+1} остается длинным, т.е. период активности N_{i+1} содержит фазу[4] 3 и длительность фазы 4 равна q_4. В общем случае дело обстоит так: $Q_1 = q_2 + \alpha_0 + q_4$. Представим α_0 в виде $\alpha_0 = k\gamma + \delta$, где k – неотрицательное целое, $0 \le \delta < \gamma$. Пока $i \le k$, $Q_{i+1} = q_2 + \alpha_i + q_4$, где $\alpha_i = \alpha_0 - i\gamma$, откуда

$$Q_{k+1} = q_2 + \alpha_k + q_4 = q_2 + \alpha_0 - k\gamma + q_4 = q_2 + \delta + q_4. \tag{12}$$

Заменив в (12) q_4 на $q_1 - \gamma$, получим

$$Q_{k+1} = q_2 + \delta + q_1 - \gamma = q_1 + q_2 + \delta + \gamma.$$

Поскольку $\delta < \gamma$, то $Q_{k+1} < q_1 + q_2$, т.е. уже не является длинным. Для активации N_{k+2} нужно, чтобы $Q_{k+1} > q_1$. Поэтому, если $q_2 + \delta - \gamma \le 0$, выходной сигнал N_{k+1} недостаточен, N_{k+1} не активируется и $z = k+1$, что и требовалось доказать. В противном случае выходной сигнал N_{k+1} – короткий. Этот случай будет рассмотрен в следующем разделе.

Примечание. Утверждение 3 теоремы 1 предполагает фиксированную длительность входного сигнала и потому не противоречит следующему очевидному утверждению: для любой цепи фиксированной длины существует такая длительность внешнего сигнала, что сигнал через эту цепь пройдет. Для этого достаточно, чтобы внешний сигнал длился до момента, когда возбудится последний элемент цепи.

Чтобы получить формулировку теоремы в терминах параметров цепи, нужно заменить величины q_1 и q_4 на правые части формул (4) и (6) соответственно. В частности, равенство $q_1 = q_4$ после такой замены и простых преобразований принимает вид:

$$|v^1_{\text{en}}| = l(wd - |v^0_{\text{en}}|).$$

[4] На рисунке уже выходной сигнал N_1 является коротким для N_2.

Пример 2. Для входного сигнала в примере 1 $\alpha_0 = 1{,}2$. По формулам (4)–(6) получим $q_1 = 2{,}5$, $q_2 = 0{,}8$, $q_4 = 0{,}8$, т.е. имеет место случай 3 теоремы 1. Далее, $\gamma = q_1 - q_4 = 1{,}7$; $\alpha_0 < \gamma$ и $k = 0$. $Q_1 = q_2 + \alpha_0 + q_4 = 0{,}8 + 1{,}2 + 0{,}8 = 2{,}8$, т.е. $q_1 < Q_1 < q_1 + q_2$. Поэтому выходной сигнал $N_{i+1} = N_2$ короткий, что видно на рисунке. Завершение примера приведено в конце раздела 4.

4. Случай короткого внешнего сигнала: $Q_0 \le q_1 + q_2$

Из п. 3 теоремы 1 видно, что короткий сигнал может возникнуть не только на входе N_1, но и на входе любого элемента цепи. Однако рассмотрение общего случая потребовало бы слишком громоздких обозначений. Поэтому ограничимся случаем внешнего короткого сигнала; рассмотрение общего случая будет отличаться только обозначениями.

4.1. Точка равновесия

В случае короткого сигнала фаза 3 для N_1 отсутствует и

$$Q_0 = q_1 + \varepsilon_0, \quad 0 < \varepsilon_0 \le q_2. \tag{13}$$

Очевидно, что $q_{21} = \varepsilon_0$. Тогда

$$Q_1 = q_{21} + q_{41} = \varepsilon_0 + q_{41}. \tag{14}$$

За время $q_{21} = \varepsilon_0$ потенциал N_1 вырастет на величину $\Delta U_{21} \le lP$:

$$\Delta U_{21} = \varepsilon_0 \left(wd - |v_{\text{en}}^1|\right), \tag{15}$$

после чего внешний сигнал отключится и у N_1 начнется урезанная фаза 4. При этом

$$q_4 = \Delta U_{21} / \left|v_{\text{en}}^1\right| = \varepsilon_0 \left(wd - |v_{\text{en}}^1|\right) / |v_{\text{en}}^1|. \tag{16}$$

В дальнейшем формула $\left(wd - |v_{\text{en}}^1|\right) / |v_{\text{en}}^1|$ неоднократно понадобится. Поэтому введем обозначение $\left(wd - |v_{\text{en}}^1|\right) / |v_{\text{en}}^1| = F$ и перепишем (16) в виде

$$q_{41} = \varepsilon_0 F. \tag{16$'$}$$

Заметим, что $1 + F = 1 + \left(wd - |v_{\text{en}}^1|\right) / |v_{\text{en}}^1| = wd / |v_{\text{en}}^1|$.

Из (14) и (16′) следует, что

$$Q_1 = \varepsilon_0 + q_{41} = \varepsilon_0 + \varepsilon_0 F = \varepsilon_0 (1 + F). \tag{17}$$

Для того чтобы короткий сигнал прошел по всей цепи с сохранением длительности, т.е. чтобы выполнялось равенство $Q_{i+1} = Q_i$, $i = 1, \ldots, n$, как и в случае длинного сигнала, необходимо и достаточно выполнение равенства $q_1 = q_{4i}$. Однако если для длинного сигнала $q_{4i} = q_4$, т.е. зависит только от

параметров цепи, то при коротком сигнале, как видно из (16′), величина q_{4i} зависит еще и от длины сигнала, т.е. выполняется только при определенном значении ε_0, которое, в свою очередь, должно удовлетворять условию (13).

Выясним теперь, при каком значении ε_0 равенство $q_1 = q_{41}$ выполняется. Подставляя в него вместо q_1 и q_{41} их выражения из формул (4) и (16′) соответственно, получим

$$P/\left(wd - |v_{\text{en}}^0|\right) = \varepsilon_0 F,$$

откуда

$$\varepsilon_0 = P/\left(\left(wd - |v_{\text{en}}^0|\right) F\right). \tag{18}$$

Из условия (13) получим:

$$0 < P/\left(\left(wd - |v_{\text{en}}^0|\right) F\right) \le q_2. \tag{19}$$

Таким образом, справедливо следующее утверждение.

Лемма 2. Для любой цепи, параметры которой удовлетворяют условию (19)*, существует величина* $\varepsilon^* \le q_2$*, определяемая соотношением* (18)*, такая, что внешний сигнал длины* $q_1 + \varepsilon^*$ *проходит по цепи без изменения длительности, т.е. является равновесным.*

Для цепи, удовлетворяющей условию (19), момент $q_1 + \varepsilon^*$ назовем *точкой равновесия.*

Пример 3. Для сети с параметрами, заданными таблицей 1, условие (19) не выполняется. Действительно, для этой сети, как было показано в конце предыдущего раздела, $q_4 < q_1$, и, следовательно, $q_{4i} < q_1$. Поэтому условие (19) для этой сети выполняться не может.

Пример 4. Для сети с параметрами, заданными табл. 2, условие (19) выполняется. Для этих параметров $F = \left(wd - |v_{\text{en}}^1|\right)/|v_{\text{en}}^1| = 1/0{,}6$; подстановка в (18) дает $\varepsilon^* = \frac{0{,}5}{0{,}8F} = 0{,}375$. Поскольку по формуле (5) $q_2 = 0{,}5$, то $\varepsilon^* < q_2$.

В дальнейшем понадобится формула

$$q_{4i}/q_{2i} = F, \tag{20}$$

которая следует из (16′).

Поскольку при коротком сигнале величины q_{2i}, q_{4i}, Q_i зависят от ε, введем для них более подробные обозначения. Эти величины в точке равновесия

Таблица 2

	P	$U_{\max}$	U_0	v_{en}^0	v_{en}^1	d	w
N_0						1,6	
N_i	0,5	1	0	−0,8	−0,6	1,6	1,0

(если она существует) обозначим как q_{*21}, q_{*41}, Q_{*0}, а при $Q_0 = q_1 + \varepsilon_0 - x$, как q_{x21}, q_{x41}, Q_{x0}. В этих обозначениях

$$q_{*41} = q_1, \quad q_{*21} = \varepsilon^*. \tag{21}$$

Из (20) и (21) следует

$$q_1 = \varepsilon^* F. \tag{22}$$

Заметим, что равенство (22), предполагающее существование точки равновесия и, соответственно, справедливость равенства $q_1 = q_{4i}$ для любого i, следует из (16′) и (21).

4.2. Случай, когда внешний сигнал короче равновесного

В этом случае

$$Q_{\lambda 0} = q_1 + \varepsilon^* - \lambda, \quad \lambda > 0 \quad \text{и} \quad q_{\lambda 21} = \varepsilon^* - \lambda. \tag{23}$$

Из (20) имеем

$$q_{\lambda 41} = (\varepsilon^* - \lambda)F = q_1 - \lambda F; \quad Q_{\lambda 1} = q_{\lambda 21} + q_{\lambda 41} = q_1 + \varepsilon^* - \lambda - \lambda F$$

и с учетом (23)

$$Q_{\lambda 0} - Q_{\lambda 1} = \lambda F. \tag{24}$$

Таким образом, при прохождении через N_1 длительность сигнала уменьшается на величину λF. Если эта величина больше длительности фазы 2, т.е. $q_{\lambda 21} = \varepsilon^* - \lambda \le \lambda F$, то выходной сигнал N_1 становится недостаточным и распространение сигнала прекращается. В противном случае N_2 активируется и генерирует выходной сигнал.

Лемма 3. Пусть на вход цепи, имеющей точку равновесия, подан короткий сигнал, который короче равновесного (т.е. сигнал, при котором $q_1 > q_{41}$) и который проходит через p элементов. Тогда для любого $i = 1, \ldots, p-1$ справедливо неравенство

$$Q_{\lambda i} - Q_{\lambda, i+1} > Q_{\lambda, i-1} - Q_{\lambda i}. \tag{25}$$

Для доказательства леммы понадобится несколько простых соотношений, верных для любого $i < p$.

Во-первых, при коротком сигнале очевидно, что

$$q_{\lambda 2, i+1} = Q_{\lambda i} - q_1. \tag{26}$$

Во-вторых, так как $Q_{\lambda i} = q_{\lambda 2i} + q_{\lambda 4i}$, то с учетом (20) имеем $Q_{\lambda i} = q_{\lambda 2i} + q_{\lambda 2i} F$ или

$$Q_{\lambda i} = q_{\lambda 2i}(1 + F). \tag{27}$$

Из (27) следует, что $Q_{\lambda i} - Q_{\lambda, i+1} = (q_{\lambda 2i} - q_{\lambda 2, i+1})(1 + F)$.

Поэтому для доказательства неравенства (25) достаточно доказать неравенство

$$q_{\lambda 2i} - q_{\lambda 2,i+1} > q_{\lambda 2,i-1} - q_{\lambda 2i}. \tag{28}$$

Используя (26), в левой части (28) заменим $q_{\lambda 2i}$ и $q_{\lambda 2,i+1}$ на $Q_{\lambda,i-1} - q_1$ и $Q_{\lambda i} - q_1$ соответственно, а после сокращения q_1 воспользуемся (27). Получим $q_{\lambda 2i} - q_{\lambda 2,i+1} = Q_{\lambda,i-1} - Q_{\lambda i} = (q_{\lambda 2,i-1} - q_{\lambda 2i})(1 + F)$, откуда непосредственно следует справедливость (28). Тем самым лемма доказана.

Из леммы 3 следует, что:

— все разности $Q_{\lambda,i-1} - Q_{\lambda i}$ положительны, так как согласно (24) $Q_{\lambda 0} - Q_{\lambda 1} > 0$, а в дальнейшем они только увеличиваются; иначе говоря, величины $Q_{\lambda i}$ уменьшаются с возрастающей скоростью;

— поэтому найдется такое l, что выходной сигнал элемента N_l окажется недостаточным и не сможет активировать элемент N_{l+1}. Способ вычисления l легко извлечь из доказательства леммы 3.

Теперь можно завершить доказательство п. 3 теоремы 1.

Длительность короткого выходного сигнала элемента N_{k+1} имеет вид $Q_{k+1} = q_2 + q_{4,k+1}$. По условию п. 3 теоремы 1 параметры цепи таковы, что $q_1 > q_4$. Следовательно, $q_1 > q_{4,k+1}$, т.е. этот сигнал короче равновесного и удовлетворяет условиям леммы 3. Поэтому найдется такое l, что выходной сигнал элемента N_{k+l+1} окажется недостаточным. Тем самым доказательство теоремы 1 завершено.

Еще одно следствие леммы 3 касается цепей, для которых $q_4 = q_1$, но сигнал — короткий. В этом случае фаза 4 будет укороченной: $q_{41} < q_4 = q_1$, т.е. сигнал короче равновесного, и согласно лемме 3 по достаточно длинной цепи этого класса сигнал не пройдет.

4.3. Случай, когда внешний сигнал длиннее равновесного

В этом случае

$$Q_{\lambda 0} = q_1 + \varepsilon^* + \lambda, \quad \lambda > 0 \quad \text{и} \quad q_{\lambda 21} = \varepsilon^* + \lambda. \tag{29}$$

По лемме 2 имеем

$$q_{\lambda 41} = (\varepsilon^* + \lambda)F = q_1 + \lambda F; \quad Q_{\lambda 1} = q_{\lambda 21} + q_{\lambda 41} = q_1 + \varepsilon^* + \lambda + \lambda F$$

и с учетом (29)

$$Q_{\lambda 1} - Q_{\lambda 0} = \lambda F.^5 \tag{30}$$

[5] Это равенство совпадает с (23) с точностью до перемены знака в левой части. Все последующее доказательство леммы 4 также повторяет доказательство леммы 3 с точностью до перемены знака.

Таким образом, при прохождении через N_1 длительность сигнала увеличивается на величину λF. Если эта длительность больше длительности фазы 2, то выходной сигнал N_1 становится длинным. В противном случае справедлива

Лемма 4. Пусть на вход цепи, имеющей точку равновесия, подан короткий сигнал, который длиннее равновесного (т.е. сигнал, длительность которого имеет вид (29)*) и который проходит через l элементов, причем сигнал, генерируемый элементом N_l, остается коротким. Тогда для любого $i < l$ справедливо неравенство*

$$Q_{\lambda,i+1} - Q_{\lambda i} > Q_{\lambda i} - Q_{\lambda,i+1}. \tag{31}$$

Из (27) следует, что $Q_{\lambda,i+1} - Q_{\lambda i} = (q_{\lambda 2,i+1} - q_{\lambda 2i})(1 + F)$.

Поэтому для доказательства неравенства (30) достаточно доказать неравенство

$$q_{\lambda 2,i+1} - q_{\lambda 2i} > q_{\lambda 2i} - q_{\lambda 2,i-1}. \tag{32}$$

Используя (26), в левой части (33) заменим $q_{\lambda 2,i+1}$ и $q_{\lambda 2i}$ на $Q_{\lambda,i-1} - q_1$ и $Q_{\lambda i} - q_1$ соответственно, а после сокращения q_1 воспользуемся (27). Получим $q_{\lambda 2,i+1} - q_{\lambda 2i} = Q_{\lambda i} - Q_{\lambda,i-1} = (q_{\lambda 2i} - q_{\lambda 2,i-1})(1 + F)$, откуда непосредственно следует справедливость (31). Тем самым лемма доказана.

Из леммы 4 следует, что:

— все разности $Q_{\lambda i} - Q_{\lambda,i-1}$ положительны, так как согласно (30) $Q_{\lambda 1} - Q_{\lambda 0} > 0$, а в дальнейшем они только увеличиваются; иначе говоря, величины $Q_{\lambda i}$ увеличиваются с возрастающей скоростью;

— поэтому найдется такое l, что выходной сигнал элемента N_l окажется длинным и, следовательно, длительность фазы разрядки N_l будет равна q_4. Так как цепь имеет точку равновесия и потому $q_1 = q_{4i} < q_4$, то поведение сигнала при его прохождении через элементы $N_{l+1}, N_{l=2}, \ldots$ описывается случаем 2 теоремы 1: при прохождении сигнала через каждый элемент сигнал будет удлиняться на одну и ту же величину.

Способ вычисления l легко извлечь из доказательства леммы 4.

Из лемм 2, 3 и 4 непосредственно следует

Теорема 2. Для любой цепи, параметры которой удовлетворяют условию (19):

1) *существует точка равновесия: такая величина $\varepsilon^* \le q_2$, что внешний сигнал длительности $q_1 + \varepsilon^*$ проходит по цепи без изменения длительности;*

2) *если сигнал длительности $q_1 + \varepsilon$ короче равновесного, т.е. $q_1 < \varepsilon < \varepsilon^*$, то существует величина z_2 такая, что через цепь, содержащую не менее z_2 элементов, этот сигнал не пройдет*;

3) *если сигнал длительности $q_1 + \varepsilon$ длиннее равновесного, т.е. $\varepsilon^* < \varepsilon < q_2$, то существует величина r такая, что в цепи, содержащей не менее r эле-*

ментов, потенциал r-го элемента достигнет максимума; для r-го и последующих элементов будет выполняться случай 2 теоремы 1.

Вторая половина утверждения 3 теоремы следует из того, что условие (19) означает выполнение равенства $q_1 = q_{4i} < q_4$.

4.4. Случай, когда точка равновесия отсутствует

Точка равновесия существует, если выполняется равенство $q_1 = q_{4i}$. В этом случае оно выполняется для любого i. Отсутствие точки равновесия означает, что при любой длительности короткого сигнала, т.е. сигнала, длительность которого имеет вид (13), равенство $q_1 = q_{4i}$ не выполняется ни для какого i. Это возможно только в двух случаях: либо $q_{4i} = q_4$, т.е. сигнал должен быть длинным (п. 1 теоремы 1), либо $q_1 > q_4$ (п. 3 теоремы 1). Второй случай означает, что для любого короткого сигнала существует такое натуральное k, что через цепь длины не меньше k сигнал не пройдет.

4.5. Итоговый результат

Общая картина прохождения сигнала по однородной цепи, которая следует из результатов пп. 3, 4, представлена в табл. 3, где все цепи разбиты на три типа, для которых $q_1 = q_4$, $q_1 < q_4$, $q_1 > q_4$. Случаи, представленные клетками таблицы, в дальнейшем будем кодировать строкой и столбцом: например, случай длинного сигнала для типа 1 кодируется как Т1Д, а случаи короткого сигнала — как Т1К, Т2К1, Т2К2, Т2К3.

Таблица 3

Сигнал	Тип 1: $q_1 = q_4$	Тип 2: $q_1 < q_4$	Тип 3: $q_1 > q_4$
Длинный	Проходит с сохранением длительности	Проходит с увеличением длительности	Укорачивается; не проходит при длине цепи $n < z_1$
Короткий	Укорачивается; не проходит при длине цепи $n < z_3$	Есть точка равновесия: 1) равновесный сигнал проходит с сохранением длительности; 2) сигнал короче равновесного не проходит при длине цепи $n < z_2$; 3) сигнал длиннее равновесного проходит с увеличением длительности	
Недостаточный	Не проходит через первый элемент		

Наличие в цепи неоднородных элементов существенно повышает возможности сохранения длительности внешнего сигнала. Например, если начальный отрезок цепи состоит из $r_1 < z_1$ элементов типа 3 (см. табл. 3), то длинный

сигнал проходит этот отрезок, хотя и укорачивается. Если за этим отрезком последует отрезок цепи, состоящий из элементов типа 2, то существует такое число r_2 элементов этого отрезка (способ его вычисления можно извлечь из доказательства п. 3 теоремы 1), что на выходе $(r_1 + r_2)$-го элемента длительность сигнала будет не меньше длительности внешнего сигнала.

5. Метод вычисления поведения асинхронной цепи

Информации, представленной табл. 3, недостаточно для того, чтобы для класса цепей с заданными параметрами вычислить протокол конкретной цепи этого класса. Здесь опишем метод вычисления протокола функционирования цепи, который опирается на два специфических свойства однородных асинхронных цепей:

1) каждый элемент активируется не более одного раза;

2) если на входе элемента N_{i+1} сигнал появился, то это произошло в момент

$$t_{1i} = (i-1)q_1 < q_4. \tag{33}$$

Равенство (33) следует из того очевидного факта, что в момент окончания фазы 1 элемента N_{i-1}, т.е. его включения, на входе элемента N_i появляется сигнал и запускается его фаза 1. Поэтому моменты $t_{1,i-1}$ и t_{1i} отстоят друг от друга на величину q_1.

Для описания метода введем ряд новых понятий.

Статическим профилем Pr_i элемента N_i называется вектор (q_1, q_2, q_4, q_5). Эти числа определяются параметрами элемента и вычисляются по формулам (4)–(7). Напомним, что длина τ_i входного сигнала для N_i – это длительность активности N_{i-1}, т.е. $\tau_i = Q_{i-1}$.

Динамический профиль $Pr_i(t_{1i})$ элемента N_i – это вектор $Pr_i\,(t_{1i}) = (t_{1i}, t_{2i}, t_{3i}, t_{4i}, t_{5i})$, который с учетом (33) можно записать как $Pr_i\,(t_{1i}) = ((i-1)q_1, t_{2i}, t_{3i}, t_{4i}, t_{5i})$. Отсутствие какой-либо фазы и соответственно момента ее наступления будем обозначать знаком $\sim$.

Динамическим профилем цепи из n однородных элементов с заданными параметрами называется множество динамических профилей ее элементов. Поскольку динамический профиль элемента привязан к временной шкале, то динамический профиль однородной цепи однозначно соответствует временной диаграмме функционирования цепи, пример которой приведен на рисунке.

Предлагаемый метод заключается в вычислении протокола функционирования цепи путем последовательного (начиная с $Pr_i(t_{11})$) вычисления динамических профилей элементов цепи и расположения событий, содержащихся в этих профилях, т.е. моментов $t_{ji}(j-1,\ldots,5;\ i=1,\ldots,n)$ на временной шкале. Он описывается следующим образом.

Пусть заданы параметры цепи, включая число n ее элементов, и длина Q_0 входного сигнала.

1. Вычисляем величины q_1, q_2, q_4, q_5, F.

2. Если $Q_{i-1} > q_1 + q_2$, то сигнал длинный, переход к п. 3; если $q_1 < Q_{i-1} < q_1 + q_2$, то сигнал короткий, переход к п. 4; если $Q_{i-1} < q_1$, то сигнал недостаточный, переход к п. 5.

3. Длинный входной сигнал имеет вид $Q_{i-1} = q_1 + q_2 + \alpha_{i-1},\ \alpha_{i-1} > 0$. Компоненты $Pr_i(t_{1i})$ вычисляются следующим образом:

$t_{1i} = (i-1)q_1;$

$t_{2i} = (i-1)\,q_1 + q_1 = iq_1;$

$t_{3i} = t_{2i} + q_2 = iq_1 + q_2;$

$t_{4i} = t_{3i} + \alpha_{i-1} = iq_1 + q_2 + \alpha_{i-1};$

$t_{5i} = t_{4i} + q_4 = iq_1 + q_2 + \alpha_i + q_4;$

$t_{zi} = t_{5i} + P/v^0_{\text{en}}.$

$Q_i = t_{5i} - t_{2i}.$

Если $i < n$, увеличиваем i на 1 и переходим к п. 2; если $i = n$, переходим к п. 6.

4. Короткий входной сигнал имеет вид $Q_{i-1} = q_1 + \varepsilon_{i-1}$, $0 < \varepsilon_{i-1} \le q_2$.

$t_{1i} = (i-1)q_1;$

$t_{2i} = (i-1)\,q_1 + q_1 = iq_1;$

$t_{3i} = \sim;$

$t_{4i} = t_{2i} + q_{2i} = iq_1 + \varepsilon_{i-1};$

$t_{5i} = t_{4i} + q_{4i} = iq_1 + \varepsilon_{i-1} + \varepsilon_{i-1}F = iq_1 + \varepsilon_{i-1}\,(1+F) =$

$= iq_1 + \varepsilon_{i-1} wd/|v^1_{\text{en}}|$ (см. (16)).

$t_{zi} = t_{5i} + P/v^0_{\text{en}}.$

$Q_i = t_{5i} - t_{2i}.$

Если $i < n$, увеличиваем i на 1 и переходим к п. 2; если $i = n$, переходим к п. 6.

5. Недостаточный входной сигнал имеет вид $Q_{i-1} = \omega < q_1$.

$t_{1i} = (i-1)q_1;$

$t_{2i} = t_{3i} = t_{4i} = \sim;$

$t_{5i} = (i-1)\,q_1 + \omega;$

$t_{zi} = t_{5i} + \omega(wd - |v^0_{\text{en}}|)/|v^0_{\text{en}}|$ (вычисляется аналогично (16)).

Переходим к п. 6.

6. Упорядочиваем вычисленное множество чисел $\{t_{ji}\}$ по возрастанию.

7. Конец алгоритма.

Для проверки вычислений полезно отметить, что некоторые события должны совпадать. В частности, $t_{2,i-1} = t_{1i}$, $t_{5,i-1} = t_{4i}$.

Пример 5. Вычислим этим методом протокол цепи, заданной табл. 1. Для этой цепи

$$q_1 = 2{,}5; \quad q_2 = 0{,}8; \quad q_4 = 0{,}8; = 2{,}5; \quad q_5 = 0{,}625.$$

$$F = \frac{\left(wd - |v_{\text{en}}^1|\right)}{|v_{\text{en}}^1|} = \frac{0{,}5}{0{,}5} = 1.$$

Если длительность входного сигнала равна 4,5, то

$q_{31} = Q_0 - q_1 - q_2 = 4{,}5 - 2{,}5 - 0{,}8 = 1{,}2;$

$Q_1 = q_2 + q_{31} + q_4 = 0{,}8 + 1{,}2 + 0{,}8 = 2{,}8.$

$Q_1 < q_1 + q_2 = 2{,}5 + 0{,}8,$ поэтому сигнал для N_2 будет коротким.

Вычислим динамические профили элементов.

1. Для N_1: $i = 1$.

$Q_0 = 4{,}5 = q_1 + q_2 + 1{,}2.$ Сигнал длинный и $\alpha_0 = 1{,}2.$

$t_{11} = (i-1)\, q_1 = 0;$

$t_{21} = (i-1)\, q_1 + q_1 = q_1 = 2{,}5;$

$t_{31} = t_{21} + q_2 = 2{,}5 + q_2 = 3{,}3;$

$t_{41} = t_{31} + \alpha_0 = 3{,}3 + \alpha_0 = 4{,}5;$

$t_{51} = t_{41} + q_4 = 4{,}5 + 0{,}8 = 5{,}3;$

$$t_{z1} = t_{51} + \frac{P}{|v_{\text{en}}^0|} = 5{,}3 + \frac{0{,}5}{0{,}8} = 5{,}925.$$

$Q_1 = t_{51} - t_{21} = 2{,}8;$

$Pr_1\,(t_{11}) = (0;\ 2{,}5;\ 3{,}3;\ 4{,}5;\ 4{,}5;\ 5{,}3).$

2. Для $N_2 : i = 2$.

$Q_1 = 2{,}8 = q_1 + 0{,}3.$ Сигнал короткий и $\varepsilon_1 = q_{22} = 0{,}3.$

$t_{12} = (i-1)\, q_1 = 2{,}5;$

$t_{22} = (i-1)\, q_1 + q_1 = 2q_1 = 5{,}0;$

$t_{32} = \sim;$

$t_{42} = t_{22} + q_{22} = t_{22} + \varepsilon_1 = 5{,}3;$

$t_{52} = t_{42} + q_{42} = 5{,}3 + \varepsilon_1 F = 5{,}6;$

$$t_{z2} = t_{52} + \frac{P}{|v_{\text{en}}^0|} = 5{,}6 + \frac{0{,}5}{0{,}8} = 6{,}225.$$

$Q_2 = t_{52} - t_{22} = 5{,}6 - 5{,}0 = 0,6.$ $Q_2 < q_1$; сигнал недостаточный.

3. Для $N_3 : i = 3$.

$$t_{13} = (i-1)\, q_1 = 5{,}0;$$
$$t_{23} = t_{33} = q_{43} = \sim;$$
$$t_{53} = t_{13} = Q_2 = 5{,}0 + 0{,}6 = 5{,}6.$$
$$t_{z3} = t_{53} + Q_2 \frac{wd - |v_{\text{en}}^0|}{|v_{\text{en}}^0|} = 5{,}6 + 0{,}6 \cdot \frac{0{,}2}{0{,}8} = 5{,}75.$$

На этом вычисление динамических профилей заканчивается. Представим их в виде табл. 4:

Таблица 4

Элемент	t_{1i}	t_{2i}	t_{3i}	t_{4i}	t_{5i}	t_{zi}
N_1	0	2,5	3,3	4.5	5,3	5,925
N_2	2,5	5,0	*	5,3	5,6	6,225
N_3	5,0	*	*	*	5,6	5,75

Расположение событий на единой шкале времени представлено в табл. 5.

Таблица 5

События	t_{11}	t_{21}, t_{12}	t_{31}	t_{41}	t_{22}, t_{31}	t_{51}, t_{42}	t_{52}	t_{z1}	t_{z2}	t_{z3}
Значения	0	2,5	3,3	4,5	5,0	5,3	5,6	5,925	6,225	5,75

6. Заключение

Естественным продолжением этого исследования является рассмотрение сетей произвольной топологии.

Полученные результаты можно использовать в нейробиологии при анализе распространения сигналов в нервных системах [1, 8, 13], в исследованиях распространения влияний в социальных сетях [14, 15], а также в пороговых моделях социального поведения [16–20].

СПИСОК ЛИТЕРАТУРЫ

1. *Кузнецов О.П., Баженков Н.И., Болдышев Б.А. и др.* Асинхронная дискретная модель химических взаимодействий в простых нейронных системах // Искусственный интеллект и принятие решений. 2018. № 2. С. 3–20.
2. *Кузнецов О.П.* Асинхронные многосортные системы // АиТ. 2021. № 2. С. 132–148.
3. *Brzozowski J.A.* Topics in Asynchronous Circuit Theory // Recent Advances Formal Languages Appl. 2006. V. 25. P. 11–42.

4. *Sparsø J.* Introduction to Asynchronous Circuit Design. TU of Denmark, 2020.
5. *Chandy K.M.* Event-Driven Applications: Costs, Benefits and Design Approaches. California Institute of Technology, 2006.
6. *Davies Alex.* Async in C# 5.0. O'Reilly, 2012.
7. *Buzsáki G.* Neural Syntax: Cell Assemblies, Synapsembles, and Readers // Neuron. 2010. V. 68. No. 3. P. 362–385.
8. *Feinerman O., Segal M., Moses E.* Signal Propagation Along Unidimensional Neuronal Networks // J. Neurophysiolog. 2005. V. 94. No. 5. P. 3406–3416.
9. *D'Haene M., Schrauwen B., Van Campenhout J., Stroobandt D.* Accelerating Event-Driven Simulation of Spiking Neurons with Multiple Synaptic Time Constants // Neural Comput. 2008. V. 21. P. 1068–1099. https://doi.org/10.1162/neco.2008.02-08-707
10. *Naveros F., Garrido J.A., Carrillo R.R., Ros E., Luque N.R.* Event- and Time-Driven Techniques Using Parallel CPU-GPU Co-processing for Spiking Neural Networks // Frontiers Neuroinform. 2017. V. 11. Article 7. P. 1–22.
11. *Pecevski D., Kappeland D., Jonke Z.* Nevesim: Event-Driven Neural Simulation Framework with a Python Interface // Frontiers Neuroinform. 2014. V. 8. P. 70.
12. *Kuznetsov O.P.* Signal Spreading Through a Chain of Asynchronous Threshold Elements // Lecture Notes in Networks and Systems. Proceedings of the Fifth International Scientific Conference "Intelligent Information Technologies for Industry" (IITI'21). 2021. V. 330. P. 24–34.
13. *Николас Дж., Мартин Р., Валлас Б., Фукс П.* От нейрона к мозгу. Пер. с англ. Изд. 5-е, стереотип. М.: УРСС: ЛЕНАНД, 2019.
14. *Jackson M.O.* Social and Economic Networks. Prinston Univer. Press, 2008.
15. *Kempe D., Kleinberg J., Tardos E.* Maximizing the Spread of Influence through a Social Network // Theory of Computing. 2015. V. 11. No. 4. P. 105–147.
16. *Schelling T.* Dynamic Models of Segregation // J. Math. Sociol. 1971. V. 1. P. 143–186.
17. *Granovetter M.S.* Threshold Models of Collective Behavior // Amer. J. Sociolog. 1978. V. 83. No. 6. P. 1420–1443.
18. *Li Z., Tang X.* A Study of Collective Action Threshold Model Based on Utility and Psychological Theories // Lecture Notes in Computer Science. 2012. V. 7669. P. 474–482.
19. *Бреер В.В.* "Модели толерантного порогового поведения (от Т. Шеллинга – к М. Грановеттеру)" // Проблемы управления. 2016. № 1. Р. 11–20.
20. *Zhilyakova L., Gubanov D.* Double-Threshold Model of the Activity Spreading in a Social Network: The Case of Two Types of Opposite Activities // Proceedings of the 11th IEEE International Conference on Application of Information and Communication Technologies AICT2017. 2017. V. 2. P. 267–270.

Статья представлена к публикации членом редколлегии Д.Е. Пальчуновым.

Поступила в редакцию 08.12.2021

После доработки 10.01.2022

Принята к публикации 26.01.2022

© 2022 г. И.С. ПРОСКУРКИН, канд. физ.-мат. наук (megavolt007@mail.ru),
В.К. ВАНАГ, д-р. физ.-мат. наук (vvanag@kantiana.ru)
(Центр нелинейной химии, Балтийский федеральный университет им. Иммануила Канта, Калининград)

СЛУЧАЙНОЕ ПРИНЯТИЕ РЕШЕНИЙ В СЕТЯХ ИМПУЛЬСНО СВЯЗАННЫХ СПАЙКОВЫХ ОСЦИЛЛЯТОРОВ[1]

Теоретически и экспериментально исследована иерархическая сеть импульсно связанных спайковых микроосцилляторов (МО), способная случайным образом реагировать на внешний сигнал. Сеть включает в себя блоки Антенна, Центральный Генератор Ритмов (ЦГР) и блок Принятия Решений (ПР). Внешний сигнал вызывает в Антенне противофазные или синфазные колебания входящих в Антенну микроячеек. ЦГР также обладает этими двумя колебательными модами. Какую моду принять блоку ЦГР в ответ на появление колебательной моды в Антенне, решает блок ПР. Благодаря своей конфигурации блок ПР принимает это решение случайно. В качестве МО используются микросферы с колебательной реакцией Белоусова–Жаботинского. Импульсные связи между МО осуществляются сфокусированными на МО лучами света.

Ключевые слова: спайковые микроосцилляторы, импульсные связи, сети, принятие решений, случайный выбор, реакция Белоусова–Жаботинского.

DOI: 10.31857/S0005231022060101, **EDN:** ADGWTB

1. Введение

Изучение сетей импульсно связанных спайковых осцилляторов приобрело в последние годы особую актуальность [1–11]. В качестве таких осцилляторов рассматриваются математические модели, электрические схемы и даже автоколебательные химические реакции [12–14]. Для изучения сетей спайковых[2] химических микроосцилляторов мы используем известную автокаталитическую реакцию Белоусова–Жаботинского (БЖ) [15, 16]. Реакция БЖ представляет собой окисление, как правило, малоновой кислоты (МА) броматом в кислой среде, причем этот процесс катализируется ионами металлов

[1] Данное исследование было поддержано из средств программы стратегического академического лидерства “Приоритет 2030” Балтийского федерального университета им. И. Канта.

[2] Спайком БЖ реакции называется резкий переход системы из состояния с преимущественно восстановленным катализатором в состояние с преимущественно окисленным катализатором и обратно — из окисленного в восстановленное состояние. Такой быстрый переход обеспечивается автокатализом.

или такими металло-комплексами, как ферроин, $Fe(phen)_3^{2+}$, или $Ru(bpy)_3^{2+}$, где phen – это 1,10-фенантролин, а bpy – это 2,2′-бипиридин. Динамика БЖ реакции подобна динамике нейронов. Поэтому построение сетей химических микроосцилляторов позволяет выявлять и исследовать свойства, присущие нейросетям мозга.

К настоящему времени нами исследована относительно небольшая иерархическая сеть импульсно связанных химических осцилляторов (от 30 до 50 микроосцилляторов), способная демонстрировать адаптивное поведение [17]. Сеть импульсно связанных БЖ-осцилляторов может обладать мультиритмичностью [18]. Эффекту мультиритмичности (или мультистабильности) даже посвящен специальный выпуск журнала Chaos [19]. Этот эффект мы используем для создания мультиритмичного Центрального Генератора Ритмов (ЦГР), контролирующего различные состояния (или моды) системы. Переключения между этими состояниями осуществляются специальными импульсами, которые формируются специальным блоком Принятия Решений (ПР) [20].

Помимо функциональных блоков ПР и ЦГР наша иерархическая сеть содержит два блока “Ридер” и блок Антенна. Антенна принимает внешние сигналы, которые ассоциируются Антенной с какой-либо модой, способной в ней возникнуть. Ридеры считывают текущие ритмы блоков ЦГР и Антенны и посылают информацию на блок ПР. Он в свою очередь сравнивает информационные сигналы от двух Ридеров и в случае несовпадения ритма ЦГР с ритмом Антенны посылает специальные импульсы на ЦГР, которые переключают его текущий ритм в другой ритм, аналогичный ритму Антенны. Используемый нами термин “иерархическая сеть” подразумевает, что одни блоки сети (например, Антенна) управляют динамикой других блоков сети (например, ЦГР). Полученные ранее результаты [17, 20] наметили перспективы создания сетей, способных выполнять более сложные задачи.

2. Принципиальное описание сети

В данной статье мы представляем новую схему иерархической нейроподобной сети БЖ осцилляторов и возбудимых ячеек (рис. 1), в которой старый ПР блок разделен на две части. Одну часть мы назвали блоком “Исполнитель” (ячейки № 9–№ 12), а для другой части сохранили старое название ПР (ячейки № 13, № 14, “Pro” и “Contra”). Но теперь новый блок ПР действительно принимает решения, а не выполняет команды, как это происходило в старом блоке ПР. Функции же исполнения команд отданы блоку “Исполнитель”. Для простоты мы упростили блоки ЦГР (осцилляторы № 1 и № 2) и Антенна (возбудимые ячейки № 5 и № 6). Теперь каждый из этих блоков состоит всего из двух ячеек и может иметь только два ритма: синфазные и противофазные колебания. Соответственно, и Ридеры этих блоков (“P” и P_A) состоят только из двух возбудимых ячеек (Ридер “P” состоит из возбудимых ячеек № 3 и № 4, а ридер P_A представлен возбудимыми ячейками № 7 и № 8). Ячейки блока ЦГР (а также ячейки № 13 и № 14) находятся в колебательном

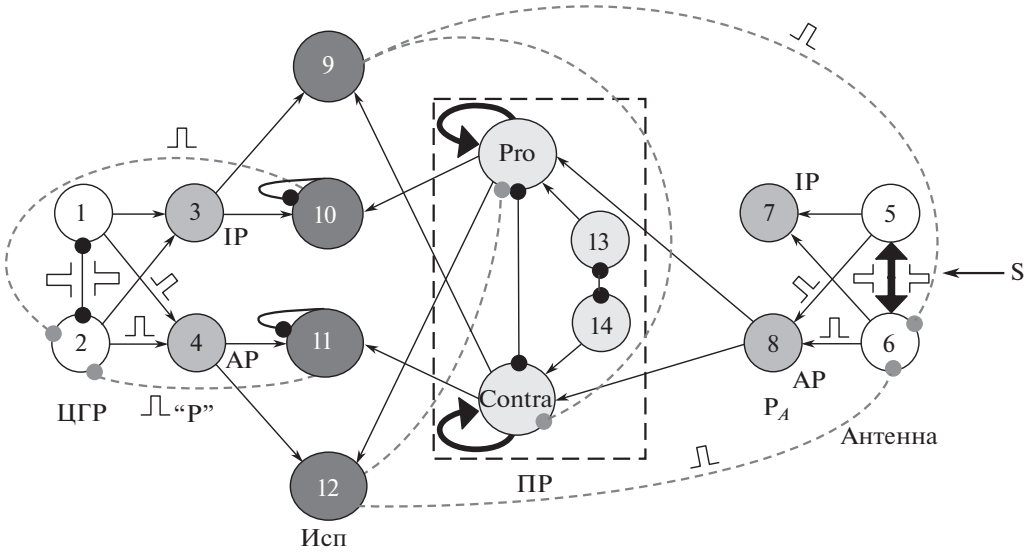


Рис. 1. Схема иерархической сети импульсно связанных спайковых осцилляторов и возбудимых ячеек, в которой реализуется случайный тип принятия решения. Обозначения: ЦГР – центральный генератор ритмов; Исп – блок Исполнитель; ПР – блок Принятия Решений. Черными жирными стрелочками отмечены возбуждающие импульсы с амплитудой I, тонкими черными стрелочками – импульсы с амплитудой $I/2$. Связи с круглыми наконечниками – это ингибиторные импульсы. Черными сплошными и серыми пунктирными линиями обозначены ингибиторные связи внутри одного блока и между разными блоками соответственно. Подписи у ячеек Ридеров соответствуют определяемым ими ритмам, для IP (inphase) – синфазный, а для AP (antiphase) – противофазный. Чтобы не перегружать рисунок, в схеме прорисованы лишь те связи, которые необходимы для ее работы в случае возникновения в Антенне противофазного ритма.

режиме, а все остальные ячейки — в возбудимом состоянии с одинаковым порогом возбуждения.

Как правило, в живых системах принятие решения является сложным процессом, в который вовлечена обработка многих сигналов из разных частей мозга. Однако в некоторых случаях решение необходимо принять очень быстро, не имея возможности на анализ всей информации. В принятии решения в таком случае, как правило, присутствует случайный выбор [21]. На рис. 1 как раз и представлена сеть, предназначенная для случайного (в значении "равновероятного") принятия решений.

Блок ЦГР может находиться в синфазном (IP) или противофазном ритме (AP). При поступлении внешнего импульса S на Антенну в ней также возникает IP или AP ритм, а Ридер P_A определяет возникший ритм (= моду). Если в Антенне возбудилась AP (IP) мода, то активируется ячейка № 8 (№ 7), которая с периодом противофазных или синфазных колебаний в Антенне (в зависимости от того, какая мода возбудилась в Антенне при приходе сигнала S)

посылает импульсы на блок ПР, точнее, на ячейки “Pro” и “Contra”. Для активации ячеек “Pro” и “Contra”, а также ячеек № 8, № 7, № 3 и № 4 на них должны одновременно прийти два импульса амплитудой $I/2$. Под словом “одновременно” понимается такая последовательность импульсов, профили которых в значительной степени пересекаются на временно́й шкале в момент их прихода на ячейку. В этом случае суммарная амплитуда возбуждающего сигнала достигает порогового значения I, при котором возбудимая ячейка генерирует спайк.

Чтобы каждая ячейка Ридера P_A активировалась в ответ на определенный ритм, два импульса от ячеек Антенны приходят на ячейки Ридера с разными фазами. На ячейку № 7 импульсы от ячеек № 5 и № 6 приходят без временны́х задержек. В этом случае ячейка № 7 возбуждается, если в Антенне установилась IP мода. На ячейку № 8 импульсы от ячеек № 5 и № 6 приходят с разностью фаз, равной половине периода противофазных колебаний T_{AP}. Поэтому ячейка № 8 возбуждается, если в Антенне установилась AP мода. Аналогичным образом работает Ридер “P” блока ЦГР.

Случайность принятия решения заключается в случайной активации ячеек “Pro” или “Contra”. Если случайно активируется ячейка “Pro”, то в блоке ЦГР должен установиться такой же внутренний ритм, как и в Антенне. И наоборот, если случайно активируется ячейка “Contra”, то в блоке ЦГР должен установиться ритм, противоположный ритму Антенны. Блок Исполнитель выполняет всю “техническую” работу по установлению нужного ритма в ЦГР. Ячейки блока Исполнитель активируются только при одновременном приходе на них двух импульсов: одного от одной из ячеек Ридера “P”, а другого от одной из ячеек блока ПР. Если одна из ячеек № 10 или № 11 активировалась, то Исполнитель отправляет переключающий сигнал на блок ЦГР, изменяя его ритм на ритм, аналогичный или противоположный ритму Антенны, в зависимости от того, активна ячейка “Pro” или “Contra”.

В качестве примера работы блока Исполнитель рассмотрим случай, когда активны ячейки “Pro” и № 3. Одновременный приход двух импульсов от ячеек “Pro” и № 3 на ячейку № 10 возбуждает ее. В активированной ячейке № 10 рождается спайк и ячейка № 10 отправляет на ячейку № 2 блока ЦГР переключающий ингибиторный импульс. Этот импульс переключает синфазную моду ЦГР в противофазную, а родившаяся в ЦГР AP мода регистрируется ячейкой № 4 Ридера “P”. Так как переключающий импульс, отправленный на ячейку № 2, имеет временну́ю задержку, превышающую периоды ритмов в блоках ЦГР и Антенна, то за это время задержки ячейка № 10 может сгенерировать второй спайк, что нарушает правильную работу сети. Поэтому для ячейки № 10 (это же относится и к ячейке № 11) дополнительно введено импульсное самоингибирование, что позволяет ячейке № 10 оставаться какое-то время после первого спайка нечувствительной к возбуждающим импульсам.

Случайный выбор принятия решения реализуется через совпадение двух независимых событий, а именно: прихода на ячейки “Pro” и “Contra” двух импульсов возбуждения от других, не синхронизованных между собой осцилля-

торов, имеющих разные частоты. В качестве одного осциллятора выступает одна из ячеек Ридера P_A (например, ячейка № 8, если в Антенне установилась АР мода). А в качестве второго осциллятора выступает один из осцилляторов пары импульсно связанных ингибиторной связью осцилляторов № 13 и № 14. Благодаря правильно подобранным параметрам ингибиторной связи эти осцилляторы колеблются в противофазе. Осциллятор № 13 посылает импульсы на ячейку “Pro”, а осциллятор № 14 — на ячейку “Contra”, которые приходят на эти ячейки в противофазе, т.е. в разные моменты времени. Второй импульс, необходимый для активации ячеек “Pro” и “Contra”, поступает от осциллятора № 8 (в случае АР моды в Антенне). Заметим, что осциллятор № 8 начинает работать в случайный момент времени, определяемый случайным моментом прихода на Антенну внешнего импульса S. Поэтому одновременный приход двух возбуждающих импульсов на ячейку “Pro” или “Contra” является случайным событием.

Если одна из ячеек “Pro” или “Contra” возбудилась (что равносильно принятию решения), то нужно эту возбужденную ячейку перевести в колебательную моду, которая должна существовать некоторое время до момента исполнения принятого решения. Поэтому мы вводим для ячеек “Pro” и “Contra” дополнительные возбуждающие импульсы самоактивации с амплитудой I, которые периодически вызывают спайки в этих ячейках после первой их активации. Период самоактивации ячеек “Pro” и “Contra”, который определяется временем задержки импульса самоактивации, подобран так, чтобы он был приблизительно равен периоду синфазных колебаний в блоке Антенна. Если возбудилась одна из ячеек “Pro” или “Contra”, то активация второй ячейки должна стать невозможной. Для выполнения этого условия мы вводим между ячейками “Pro” и “Contra”двустороннюю импульсную ингибиторную связь большой амплитуды [22].

После того как все необходимые переключения завершены и случайно принятое решение выполнено, наша сеть должна вернуться в исходное состояние, в котором она была бы готова принимать новые случайные решения, т.е. все ячейки кроме ячеек блоков ЦГР, “Р” и ячеек № 13 и № 14 должны оказаться в стационарном возбудимом состоянии. Эту работу выполняют ячейки № 9 и № 12, посылая ингибиторные импульсы на ячейки “Pro” и “Contra” блока ПР и на ячейку № 6 блока Антенна. Ингибиторные импульсы можно посылать и на ячейку № 5 блока Антенна, но это не имеет значения.

Возможен также вариант, когда возбуждаются ячейки “Pro” или “Contra”, а изначальный ритм блока ЦГР уже аналогичен или противоположен, соответственно, ритму Антенны. В этом случае переключения моды ЦГР не происходит. Вместо этого на Антенну и блок ПР отправляются от ячеек № 9 или № 12 ингибиторные импульсы обратных связей, которые возвращают эти блоки в их исходное состояние.

Таким образом, сеть импульсно связанных ячеек, представленная на рис. 1, получая внешний сигнал S, случайным образом решает, установить ли в ЦГР ритм, аналогичный или противоположный ритму Антенны.

3. Эксперимент

В качестве БЖ микроосцилляторов и возбудимых микроячеек мы используем микросферы ионообменной смолы Dowex® 50WX2 с иммобилизованным в них светочувствительным катализатором БЖ реакции, $Ru(bpy)_3^{2+}$, так называемые БЖ микросферы [23]. Метод приготовления микросфер Dowex 50WX2 с иммобилизованным $Ru(bpy)_3^{2-}$ приведен в ссылке [24]. БЖ микросферы, имеющие средний диаметр около 100 микрометров (см. рис. 2), помещаются в обращенную микроэмульсию AOT [24], наполненную всеми реагентами БЖ реакции, но без катализатора. Для приготовления AOT микроэмульсии [24] в качестве непрерывной гидрофобной фазы использовался тетрадекан. Концентрация $Ru(bpy)_3^{2+}$в микросферах была высчитана равной 3 мМ. Водные нанокапли AOT микроэмульсии с БЖ реагентами постоянно подпитывают БЖ микросферы в силу диффузионного обмена содержимым между нанокаплями и БЖ микросферами.

При достаточно большом расстоянии между БЖ микросферами они ведут себя независимым образом. Мы помещали БЖ микросферы в реактор (см. рис. 2), в качестве которого служит склеенная из предметных стеклышек прямоугольная ванночка глубиной 1 мм. После наполнения БЖ микроэмульсией и БЖ микросферами ванночка накрывалась оптическим стеклом, чтобы предотвратить возможное испарение тетрадекана.

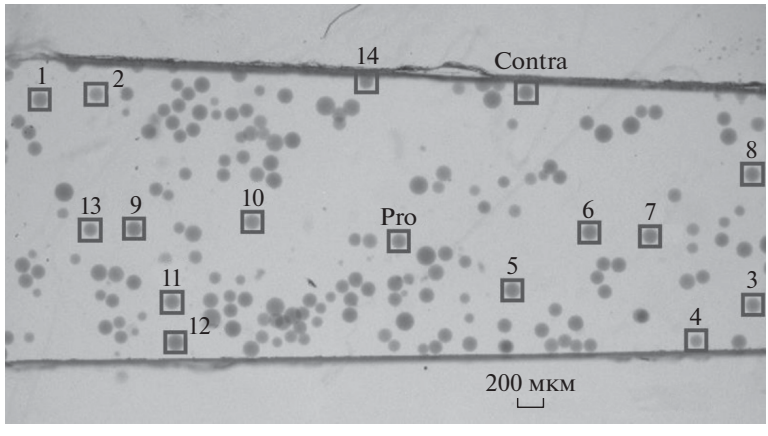


Рис. 2. Снимок используемых в эксперименте БЖ микросфер, находящихся в реакторе. Квадратными рамками отмечены границы областей облучения актиничным светом всех БЖ микросфер, используемых для реализации сети, представленной на рис. 1. Номера рядом с БЖ микросферами соответствуют номерам ячеек на схеме рис. 1. Не отмеченные номерами БЖ микросферы во время эксперимента находятся в колебательном состоянии и никак не влияют на динамику исследуемых БЖ микросфер из-за их удаленного расположения, исключающего действие диффузионных связей. Две длинные черные линии на рисунке являются “верхней” и “нижней” границами реактора (правая и левая границы находятся за пределами рисунка).

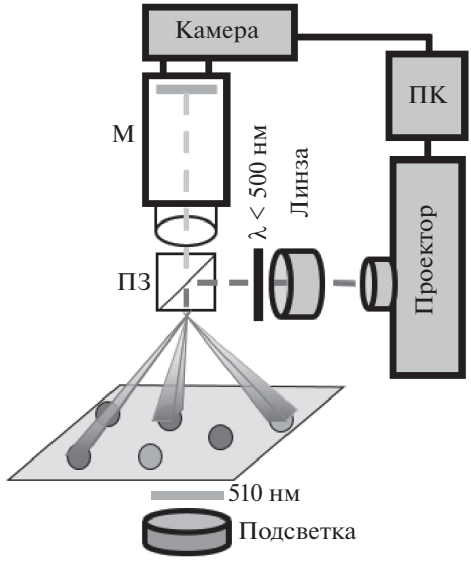


Рис. 3. Схема лабораторной установки. М – микроскоп; ПЗ – полупрозрачное зеркало с соотношением отражение/пропускание, равным 50/50; ПК – персональный компьютер. Серыми пластинками отмечены интерференционные фильтры с длиной волны $\lambda = 510$ нм и шириной пропускания $\Delta\lambda = 10$ нм. Черной пластинкой отмечен коротковолновый фильтр с границей отрезания на длине волны $\lambda = 500$ нм. Серыми и светло-серыми шариками представлены БЖ микросферы в реакторе. Проектор преобразует динамическую маску, контролируемую персональным компьютером, в лучи актиничного света, а линза фокусирует эти лучи на микросферы. Максимально возможная интенсивность актиничного света $I_{\text{макс}} = 2{,}07$ клк (килолюкса). Анализирующий свет подсветки проходит сквозь микросферы и регистрируется камерой. Резкое увеличение яркости микросферы отмечается программой LabVIEW как спайк осциллятора.

Микроэмульсия AOT описывается двумя физическими параметрами: $\omega = [H_2O]/[AOT]$ и объемной долей нанокапелек φ_d [25]. Величина ω определяет радиус водного ядра нанокапелек R_w (нм) приблизительно как $R_w \cong 0{,}17\omega$. Для нашего случая $\omega = 10$, а $\varphi_d = 0{,}45$. Концентрации БЖ реагентов в водной псевдофазе AOT микроэмульсии: $[MA] = 0{,}36$ М, $[H_2SO_4] = 0{,}5$ М, $[NaBrO_3] = 0{,}27$ М, $[NaBr] = 0{,}07$ М, где МА — это малоновая кислота. Эти концентрации поддерживают в БЖ микросферах колебания с периодом около 40 с, которые являются устойчивыми на протяжении не менее 100 мин.

С помощью видеокамеры, оборудованной на окуляре стереомикроскопа, регистрируется изменение яркости проходящего сквозь БЖ микросферы света с длиной волны $\lambda = 510$ нм (применяется интерференционный фильтр) (см. рис. 3). Свет этой длины волны не влияет на динамику фоточувствительной БЖ реакции.

Для создания импульсных связей мы используем актиничный свет с длиной волны $\lambda < 500$ нм (используется коротковолновый фильтр). В условиях нашей реакции импульсы такого света увеличивают период БЖ колебаний, т.е. являются ингибиторными импульсами. Увеличение амплитуды импульса (интенсивности освещенности) приводит к большему ингибированию. Сфокусированные лучи актиничного света, падающие на индивидуальные микросферы, создаются мощной линзой и проектором, в котором устанавливается нужная геометрическая маска, контролируемая компьютером. Необходимые области пикселей в маске принимают определенную степень яркости в нужные моменты времени, т.е. маска является динамической. Компьютерная программа, разработанная авторами в среде LabVIEW, анализирует состояние осцилляторов и контролирует все параметры импульсных связей, такие как амплитуда, длительность и задержка импульса, и создает динамическую маску, которая проецируется линзой и полупрозрачным зеркалом (ПЗ) на микросферы.

Исходно все БЖ микросферы находятся в колебательном состоянии при выбранных концентрациях БЖ реагентов. Для создания из них возбудимых ячеек мы используем постоянное облучение светом определенной интенсивности I, при которой БЖ микросферы ингибируются и переходят в возбудимое состояние. Известно, что если на короткий промежуток времени Δt прервать это ингибиторное облучение, то возбудимая ячейка даст спайк [26]. Таким образом, импульсное прекращение постоянного облучения используется нами в качестве возбуждающего импульса — это так называемый “отрицательный” импульс. Для реализации возбуждающих импульсов разной амплитуды мы понижаем интенсивность ингибиторного освещения возбудимых ячеек на амплитуду I для случая, если к этой ячейке на рис. 1 идет жирная черная стрелочка, и на амплитуду $I/2$, если к этой ячейке на рис. 1 идет тонкая черная стрелочка. Если интенсивность освещения возбудимой ячейки понижается до нуля (что соответствует амплитуде отрицательного возбуждающего импульса равной I), то такая ячейка дает спайк в случае продолжительности этого импульса не менее $\Delta t = 10$ с, но для получения стабильного результата выбрано $\Delta t = 15$ с. Приход на ячейку одного отрицательного возбуждающего импульса с амплитудой $I/2$ и длительностью $\Delta t = 15$ с понижает порог возбудимости ячейки. Поэтому для ее активации достаточно прихода второго импульса, который пересекается во времени с первым уже не в течение 10–15 с, а в течение $\Delta t_x = 5$ с (или больше).

Любой импульс в нашей сети, вышедший из одной ячейки и идущий к другой ячейке, характеризуется тремя параметрами: (1) время задержки τ между моментом рождения импульса (= момент времени появления спайка в ячейке) и моментом прихода этого импульса к адресату, т.е. к другой ячейке, (2) длительность импульса Δt, (3) амплитуда импульса I^*. Ингибиторная связь, например, между ячейками № 1 и № 2 создается положительными импульсами света со следующими параметрами: $\tau = 0$ с, $\Delta t = 3$ с и $I^* = 0{,}94$ клк. При этих параметрах устанавливаются устойчивые проти-

вофазные или синфазные колебания в зависимости от начальных условий. Напомним, что колебательные ячейки № 1 и № 2 не освещаются постоянным светом. Если из ячейки выходит несколько импульсов, то к параметрам импульсов добавляется подстрочный индекс "n_m", где n – это номер ячейки, из которой выходит импульс, а m – это номер ячейки, к которой устремляется импульс.

4. Результаты эксперимента

Используя БЖ-микроосцилляторы, показанные на рис. 2, нашу программу по управлению импульсами света (написанную в среде LabVIEW) и установку, схема которой показана на рис. 3, мы экспериментально воссоздали иерархическую сеть спайковых микроосцилляторов и возбудимых микроячеек, представленную на рис. 1. Работа этой сети в виде динамики ячеек всех ее блоков представлена на рис. 4 для случая, когда случайно активировалась ячейка "Pro".

Исходно ритм блока ЦГР является синфазным в рассматриваемом примере, и ячейка № 3 периодически генерирует спайки благодаря одновременно приходящим на нее импульсам от осцилляторов № 1 и № 2. Ячейки № 13 и № 14 блока ПР постоянно колеблются в противофазе, что достигается обоюдными ингибиторными импульсами со следующими параметрами: $\tau_{13\_14} = 0$ с, $\Delta t_{13\_14} = 3$ с и $I^*_{13\_14} = 0{,}94$ клк. Все остальные ячейки сети исходно находятся в возбудимом стационарном состоянии. Взаимные ингибиторные импульсы между ячейками Pro и Contra характеризуются следующими параметрами: $\tau = 0$ с, $\Delta t = 70$ с и $I^* = 1{,}88$ клк.

В случайный момент времени на Антенну поступает внешний сигнал S (черная вертикальная стрелочка на рис. 4) и активирует одну из ее ячеек (№ 5 или № 6), запуская в Антенне противофазный ритм. Ячейка № 8 Ридера P_A начинает генерировать спайки, регистрируя тем самым противофазный ритм Антенны (кинетика ячейки № 8 представлена пунктирной кривой на панельке "P", P_A рис. 4). Ячейки № 8 и № 13 отправляют возбуждающие импульсы на ячейку "Pro", а ячейки № 8 и № 14 — на ячейку "Contra". В момент $t = 1650$ с два импульса случайно приходят на ячейку "Pro" одновременно и возбуждают ее. Это означает, что блок ПР случайно выбрал решение "Pro".

Далее в момент времени $t \cong 1820$ с возбуждающие импульсы от ячеек № 3 и "Pro" одновременно поступают на ячейку № 10 и активируют ее. Она в свою очередь с временной задержкой τ_{10_2} отправляет переключающий ингибиторный импульс на ячейку № 2 блока ЦГР. Величина τ_{10_2} выбрана равной 60 с в силу того, что при такой задержке переключающий импульс приходит на ячейку № 2 в тот момент, когда фаза синфазных колебаний φ_{IP} находится в той области, в которой синфазный режим весьма чувствителен к внешним ингибиторным импульсам и может легко переключиться в противофазный режим. В нашем случае $\varphi_{IP} \cong 0{,}7$ (фаза меняется от нуля до единицы). Если фаза φ_{IP} будет еще ближе к единице, то чувствительность еще больше повышается, но стабильность переключения понижается. На рис. 4 видно, что пе-

реключающий импульс переводит блок ЦГР в противофазную моду в момент времени $t \cong 1900$ с. Параллельно с отправкой переключающего импульса на ячейку № 2 ячейка № 10 самоингибируется.

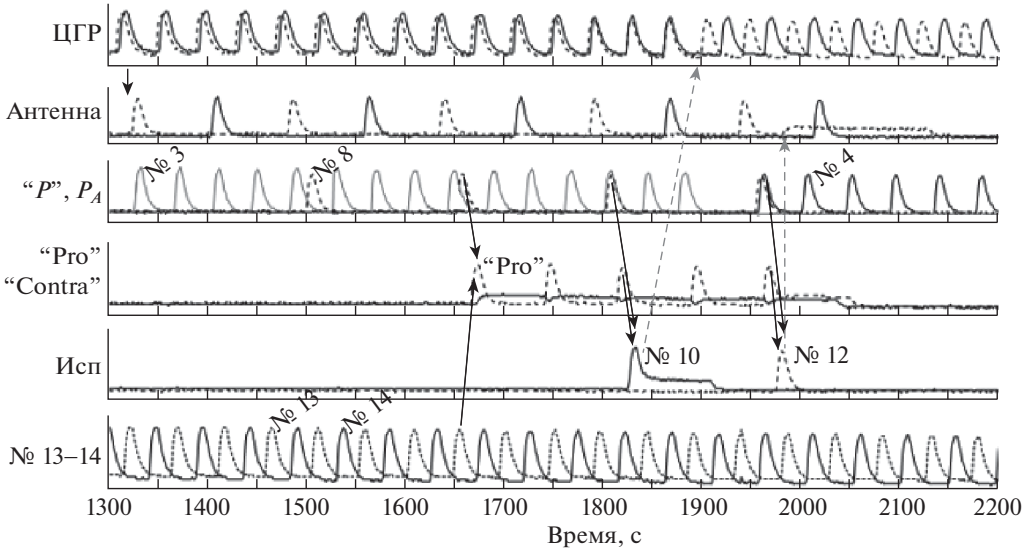


Рис. 4. Динамика ячеек сети при случайной активации ячейки “Pro”. Черными сплошными линиями со стрелочками отмечены возбуждающие импульсы, а серыми пунктирными — ингибиторные. Интенсивность постоянного освещения для всех возбудимых ячеек $I = 0{,}19$ клк. Параметры импульсов. Ингибиторные импульсы:

$$\tau_{10\_2} = 60 \text{ с},\ \tau_{11\_2} = 30 \text{ с},\ \Delta t_{10\_2} = \Delta t_{11\_2} = 3 \text{ с},\ I^*_{10\_2} = I^*_{11\_2} = 0{,}94 \text{ клк};$$

$$\tau_{12\_6} = \tau_{9\_6} = 0 \text{ с},\ \Delta t_{12\_6} = \Delta t_{9\_6} = 150 \text{ с},\ I^*_{12\_6} = I^*_{9\_6} = 1{,}88 \text{ клк},$$

$$\tau_{10\_10} = \tau_{11\_11} = 0 \text{ с},\ \Delta t_{10\_10} = \Delta t_{11\_11} = 60 \text{ с},\ I^*_{10\_10} = I^*_{11\_11} = 1{,}88 \text{ клк}.$$

Возбуждающие импульсы:

$$\tau_{5\_6} = \tau_{6\_5} = 60 \text{ с},\ \Delta t_{5\_6} = \Delta t_{6\_5} = 15 \text{ с},\ \tau_{1\_3} = \tau_{2\_3} = \tau_{5\_7} = \tau_{6\_7} = 0 \text{ с},$$

$$\Delta t_{1\_3}\tau_{8\_\text{Pro}} = \tau_{8\_\text{Contra}} = 0 \text{ с},\ \Delta t_{8\_\text{Pro}} = \Delta t_{8\_\text{Contra}} = 15 \text{ с};$$

$$\tau_{13\_\text{Pro}} = \tau_{14\_\text{Contra}} = 0 \text{ с},\ \Delta t_{14\_\text{Contra}} = \Delta t_{13\_\text{Pro}} = 15 \text{ с};$$

$$\tau_{\text{Pro}\_\text{Pro}} = \tau_{\text{Contra}\_\text{Contra}} = 60 \text{ с},\ \Delta t_{\text{Pro}\_\text{Pro}} = \Delta t_{\text{Contra}\_\text{Contra}} = 15 \text{ с};$$

$$\tau_{\text{Pro}\_10} = 0 \text{ с},\ \Delta t_{\text{Pro}\_10} = 15 \text{ с},\ \tau_{\text{Pro}\_12} = 0 \text{ с},\ \Delta t_{\text{Pro}\_12} = 15 \text{ с},$$

$$\tau_{\text{Contra}\_11} = 0 \text{ с},\ \Delta t_{\text{Contra}\_11} = 15 \text{ с},\ \tau_{\text{Contra}\_9} = 0 \text{ с},\ \Delta t_{\text{Contra}\_9} = 15 \text{ с},$$

$$\tau_{3\_10} = 0 \text{ с},\ \Delta t_{3\_10} = 15 \text{ с},\ \tau_{3\_9} = 0 \text{ с},\ \Delta t_{3\_9} = 15 \text{ с},\ \tau_{4\_11} = 0 \text{ с},$$

$$\Delta t_{4\_11} = 15 \text{ с},\ \tau_{4\_12} = 0 \text{ с},\ \Delta t_{4\_12} = 15 \text{ с}.$$

В блоке ЦГР период синфазных колебаний $T_{\text{IP}} = 38$ с, а период противофазных колебаний $T_{\text{AP}} = 44$ с. Период противофазных колебаний в Антенне $T_{\text{AP}} = 151$ с. Период противофазных колебаний в паре осцилляторов № 13–№ 14 $T_{13\_14} = 48$ с.

Если ячейка № 4 определила новый противофазный ритм ЦГР, она отправляет возбуждающий импульс на ячейку № 12 блока Исполнитель. В момент времени $t \cong 1970$ с одновременно с импульсом от ячейки № 4 на ячейку № 12 приходит импульс от ячейки "Pro", вызывая в ней спайк. Возбужденная ячейка № 12 отправляет ингибиторный импульс на ячейку № 6 Антенны. Этот импульс переводит все ячейки Антенны в стационарное возбудимое состояние, в котором Антенна готова принимать новые внешние сигналы. Ячейка № 12 посылает также ингибиторный импульс на ячейку "Pro", переводя ее в возбудимое состояние, что позволяет блоку ПР вновь принимать случайные решения.

Аналогичный эксперимент при случайной активации ячейки "Contra" также дал ожидаемый результат.

Оценим теперь численно вероятность активации ячеек "Pro" и "Contra" при случайном приходе внешнего сигнала S на Антенну. Для этого мы промоделировали работу нашей сети с момента t_{S} случайного прихода единичного импульса S и до момента активации одной из ячеек "Pro" или "Contra". Условие "единичного импульса" означает, что в Антенне возбуждается противофазная мода и, следовательно, в Ридере $\mathrm{P_A}$ всегда будет возбуждаться ячейка № 8, настроенная на эту моду. Так как исходно все ячейки Антенны и Ридера $\mathrm{P_A}$ находятся в стационарном состоянии, то "случайность" прихода сигнала S надо соотносить с начальной фазой φ колебаний пары осцилляторов № 13–№ 14.

Если t_{S} – это время прихода внешнего импульса S на одну из ячеек Антенны, то $t_{\mathrm{S}} + \tau_{\mathrm{S\_8}}$ – это время первой активации ячейки № 8 ($= t_8^{(0)}$), где временной интервал $\tau_{\mathrm{S\_8}}$ складывается из времени, необходимого для активации второй ячейки Антенны, и из времени, требуемого на приход двух импульсов от ячеек Антенны на ячейку № 8. Конкретная величина времени $\tau_{\mathrm{S\_8}}$ зависит от того, на какую ячейку, № 5 или № 6, приходит сигнал S. Из рис. 4 можно видеть, что время $\tau_{\mathrm{S\_8}} \cong 190$ с. Однако знание точного значения величины $\tau_{\mathrm{S\_8}}$ не требуется для определения вероятности активации ячеек "Pro" и "Contra". Моменты первого и последующих спайков ячейки № 8, $t_8^{(n)}$, определяются как:

$$t_8^{(n)} = t_8^{(0)} + T_{\mathrm{AP}} \times n, \tag{1}$$

где n – целое число от нуля до бесконечности, $T_{\mathrm{AP}} = 151$ с (см. рис. 4). Предположим, что в момент времени t_{S} пара осцилляторов № 13–№ 14 находится в произвольной фазе φ_0, где фаза φ меняется от 0 до 1, а осциллятор № 13 производит спайк при $\varphi = 0$ и $\varphi = 1$ (при $\varphi = 1$ фаза обнуляется до $\varphi = 0$), а осциллятор № 14 производит спайк при $\varphi = 1/2$ (так как осцилляторы № 13 и № 14 колеблются в противофазе). Тогда спайки в ячейке № 13 происходят в следующие моменты времени $t_{13}^{(m)}$:

$$t_{13}^{(m)} = t_{\mathrm{S}} + (1 - \varphi_0 + m) \times T_{\mathrm{13\_14}}, \tag{2}$$

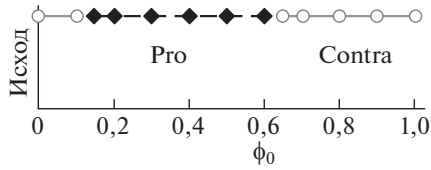


Рис. 5. Зависимость исхода активации ячеек “Pro” и “Contra” от начальной фазы колебаний φ_0 пары осцилляторов № 13–№ 14. Черный цвет соответствует активации ячейки “Pro”, а серый — активации ячейки “Contra”.

где m – целое число от 0 до бесконечности, а период T_{13_14} мы положили равным 48 с, как было измерено в наших экспериментах. Моменты спайков в ячейке № 14, $t_{14}^{(m)}$, можно задать через моменты $t_{13}^{(m)}$:

$$t_{14}^{(m)} = t_{13}^{(m)} - T_{13\_14}/2. \tag{3}$$

Так как $\tau_{13\_\mathrm{Pro}} = \tau_{8\_\mathrm{Pro}} = 0$ с, то мы полагаем, что ячейка “Pro” должна активироваться, если моменты времени $t_{13}^{(m)}$ и $t_8^{(n)}$ совпадают при каких-то целых n и m с точностью до времени перекрытия импульсов активации, причем мы учитываем только наименьшие значения n и m. Так как для активации ячейки “Pro” импульсы возбуждения могут перекрываться всего лишь на время Δt_x (= 5 с), то для активации ячейки “Pro” должно выполняться следующее неравенство:

$$\left| t_{13}^{(m)} - t_8^{(n)} \right| < \Delta t - \Delta t_x, \tag{4}$$

где $\Delta t \equiv \Delta t_{8\_\mathrm{Pro}} = \Delta t_{13\_\mathrm{Pro}}$ (= 15 с). Если же выполняется условие:

$$\left| t_{14}^{(m)} - t_8^{(n)} \right| < \Delta t - \Delta t_x, \tag{5}$$

причем раньше, чем условие (4), то должна активироваться ячейка “Contra”.

Так как выражение $\left| t_{13}^{(m)} - t_8^{(n)} \right|$ в (4) после подстановки в него выражений (1) и (2) преобразуется к виду $|(1 - \varphi_0 + m) \times T_{13\_14} - \tau_{\mathrm{S}\_8} - T_{\mathrm{AP}} \times n|$, в котором уже нет времени t_S, но есть начальная фаза φ_0 (то же самое происходит и с разностью $(t_{14}^{(m)} - t_8^{(n)})$, то при моделировании мы изменяли фазу φ_0 от нуля до единицы. Заметим, что случайный момент времени t_{S} определяет начальную фазу φ_0. На рис. 5 мы отмечаем черным цветом и ромбами исход опыта, если активировалась ячейка “Pro”, и серым цветом и кружочками, если активировалась ячейка “Contra”. Дли́ны черной и серой линий оказались равны друг другу, что доказывает, что активация ячеек “Pro” и “Contra” происходит равновероятно. Такой же равновероятный исход должен быть и при полностью случайной активации ячеек “Pro” и “Contra”. Однако наша “случайность” весьма специфична, а именно, существуют две компактные зоны фаз φ_0, в которых исход события предопределен, а вероятность попадания в одну из этих зон одинакова.

5. Заключение

В данной статье мы продемонстрировали способ случайного принятия решения сетью импульсно связанных спайковых осцилляторов. Этот способ, основанный на случайном совпадении во времени фаз или импульсов от осцилляторов с разной частотой, не является, конечно, единственным. Можно, например, использовать для принятия решения локальную подсеть из двух-трех осцилляторов, которая дает хаотические колебания [27]. Такой способ, вероятно, привел бы к более выраженной "случайности" при принятии решения. Однако нашей целью является разработка такого блока принятия решений, который сможет осуществлять выбор решения не только случайным образом, но и целенаправленно, учитывая полученный ранее опыт при принятии решений. Для этих целей мы планируем разработать блок памяти и синхронизировать его работу с другими блоками сети. Рассмотренный же в данной статье метод случайного принятия решений является лишь первым шагом на пути создания "разумного" микроробота, который не всегда работает по заданной программе.

Отметим, что полное математическое описание единичного БЖ микроосциллятора дается четырьмя дифференциальными уравнениями [18, 20]. Импульсная связь с временной задержкой между двумя микроячейками также описывается четырьмя дифференциальными уравнениями. Однако использовать столь точный математический аппарат для описания происходящих в сети событий совсем не обязательно. БЖ осцилляторы можно заменить фазовыми осцилляторами [18], а импульсное воздействие описывать при помощи кривых переустановки фазы [26, 28]. Возбудимые ячейки легко заменяются триггерными элементами с задаваемым порогом возбудимости. Временные задержки можно реализовать при помощи химических (или электрохимических) волн, распространяющихся с конечной скоростью по узким каналам между заданными микроячейками [29]. Такие замены говорят о том, что предложенный нами метод случайного принятия решений может быть универсален и применим к другим спайковым сетям.

Мы благодарим И.Л. Мальфанова за приготовление БЖ микросфер и микроэмульсии.

СПИСОК ЛИТЕРАТУРЫ

1. *Klinshov V.V.* Collective Dynamics of Networks of Active Units with Pulse Coupling: Review // Izvestiya Vysshikh Uchebnykh Zavedeniy-Prikladnaya Nelineynaya Dinamika. 2020. V. 28. P. 465–490.
Клиньшов В.В. Коллективная динамика сетей активных элементов с импульсными связями: Обзор // Изв. высш. уч. заведений. Прикладная нелинейная динамика. 2020. Т. 28. № 5. С. 465–490.
2. *Buzsaki G., Freeman W.* Editorial overview: Brain rhythms and dynamic coordination // Current Opinion in Neurobiology. 2015. V. 31. P. V–IX.
3. *Neves F.S., Timme M.* Reconfigurable Computation in Spiking Neural Networks // IEEE Access. 2020. V. 8. P. 179648–179655.

4. *Ryu H., Campbell S.A.* Stability, Bifurcation and Phase-Locking of Time-Delayed Excitatory-Inhibitory Neural Networks // Mathematical Biosciences and Engineering. 2020. V. 17. P. 7931–7957.
5. *Liang X., Zhang X.C., Xia J., Ezawa M., Zhao Y.L., Zhao G.P., Zhou Y.* A Spiking Neuron Constructed by the Skyrmion-Based Spin Torque Nano-oscillator // Appl. Phys. Lett. 2020. V. 116. #122402.
6. *Spaeth A., Tebyani M., Haussler D., Teodorescu M.* Spiking Neural state Machine for Gait Frequency Entrainment in a Flexible Modular Robot // Plos One. 2020. V. 15. P. e0240267.
7. *Kadhim K.L., Hermundstad A.M., Brown K.S.* Structured Patterns of activity in Pulse-coupled Oscillator Networks with Varied Connectivity // Plos One. 2021. V. 16. P. e0256034.
8. *Bazhanova M.V., Krylova N.P., Kazantsev V.B., Khramov A.E., Lobov S.A.* Synchronization in a Network of Spiking Neural Oscillators with Plastic Connectivity // Radiophysics and Quantum Electronics. 2020. V. 63. P. 298–309.
Бажанова М.В., Крылова Н.П., Казанцев В.Б., Храмов А.Е., Лобов С.А. Синхронизация в сети импульсных нейронных генераторов с пластичными связями // Изв. вузов. Радиофизика. 2020. Т. 63. № 4. С. 330–343.
9. *Afifurrahman, Ullner E., Politi A.* Collective Dynamics in the Presence of Finite-width Pulses // Chaos. 2021. V. 31. #043135.
10. *Galinsky V.L., Frank L.R.* Collective Synchronous Spiking in a Brain Network of Coupled Nonlinear Oscillators // Phys. Rev. Lett. 2021. V. 126. #158102.
11. *Gast R., Knosche T.R., Schmidt H.* Mean-field Approximations of Networks of Spiking Neurons with Short-term Synaptic Plasticity // Phys. Rev. E. 2021. V. 104. #044310.
12. *Kuramoto Y.* Phase- and Center-manifold Reductions for Large Populations of Coupled Oscillators with Application to Non-locally Coupled Systems // Int. J. Bifurcation and Chaos. 1997. V. 7. P. 789–805.
13. *Velichko A., Putrolaynen V., Belyaev M.* Higher-order and Long-range Synchronization Effects for Classification and Computing in Oscillator-based Spiking neural Networks // Neural Computing & Applications. 2021. V. 33. P. 3113–3131.
14. *Taylor A.F., Tinsley M.R., Wang F., Huang Z.Y., Showalter K.* Dynamical Quorum Sensing and Synchronization in Large Populations of Chemical Oscillators // Science. 2009. V. 323. P. 614–617.
15. *Belousov B.P.* Collection of Short Papers on Radiation Medicine. Moscow: Medgiz, 1959. P. 145–152.
Белоусов Б.П. В кн. Сборник рефератов по радиационной медицине за 1958 г. М.: Изд-во Медгиз, 1959. С. 145.
16. *Zhabotinsky A.M.* Periodic liquid phase reactions // Proc. Acad. Sci. USSR. 1964. V. 157. P. 392–395.
Жаботинский А.М. Периодические окислительные реакции в жидкой фазе // Докл. АН СССР. 1964. Т. 157. № 2. С. 392–395.
17. *Proskurkin I.S., Smelov P.S., Vanag V.K.* Experimental Verification of an Optochemical “neurocomputer” // Phys. Chem. Chem. Phys. 2020. V. 22. P. 19359–19367.
18. *Vanag V.K., Smelov P.S., Klinshov V.V.* Dynamical Regimes of Four Almost Identical Chemical Oscillators Coupled via Pulse Inhibitory Coupling with Time Delay // Phys. Chem. Chem. Phys. 2016. V. 18. P. 5509–5520.

19. *Feudel U., Pisarchik A.N., Showalter K.* Multistability and Tipping. From Mathematics and Physics to Climate and Brain-Minireview and Preface to the Focus Issue // Chaos. 2018. V. 28. #033501.

20. *Smelov P.S., Proskurkin I.S., Vanag V.K.* Controllable Switching between Stable Modes in a Small Network of Pulse-coupled Chemical Oscillators // Phys. Chem. Chem. Phys. 2019. V. 21. P. 3033–3043.

21. *Кузнецов О.П.* Ограниченная рациональность и принятие решений // Искусственный интеллект и принятие решений. 2019. № 1. С. 3–15.

22. *Mysore S.P., Kothari N.B.* Mechanisms of competitive selection: A canonical neural circuit framework // Elife. 2020. V. 9. P. e51473.

23. *Taylor A.F., Tinsley M.R., Showalter K.* Insights into Collective cell Behaviour from Populations of Coupled Chemical oscillators // Phys. Chem. Chem. Phys. 2015. V. 17. P. 20047–20055.

24. *Mallphanov I.L., Vanag V.K.* Distance Dependent Types of Coupling of Chemical Micro-oscillators Immersed in a Water-in-Oil Microemulsion // Phys. Chem. Chem. Phys. 2021. V. 23. P. 9130–9138.

25. *De T.K., Maitra A.* Solution behaviour of Aerosol OT in non-polar solvents // Adv. Colloid Interface Sci. 1995. V. 59. P. 95–193.

26. *Proskurkin I.S., Vanag V.K.* New Type of Excitatory Pulse Coupling of Chemical Oscillators via Inhibitor // Phys. Chem. Chem. Phys. 2015. V. 17. P. 17906–17913.

27. *Stankevich N., Volkov E.* Evolution of Quasiperiodicity in Quorum-sensing Coupled Identical Repressilators // Chaos. 2020. V. 30. #043122.

28. *Canavier C.C., Achuthan S.* Pulse Coupled Oscillators and the Phase Resetting curve // Math. Biosci. 2010. V. 226. P. 77–96.

29. *Safonov D.A., Vanag V.K.* Oscillatory Microcells Connected on a Ring by Chemical Waves // Chaos. 2021. V. 31. #063134.

Статья представлена к публикации членом редколлегии О.П. Кузнецовым.

Поступила в редакцию 26.11.2021

После доработки 17.01.2022

Принята к публикации 26.01.2022

© 2022 г. А.Я. ФРИДМАН, д-р техн. наук (fridman@iimm.ru)
(Институт информатики и математического моделирования Кольского научного центра РАН, Апатиты)

ОПЫТ ИНТЕЛЛЕКТУАЛИЗАЦИИ МЕТОДОВ СИТУАЦИОННОГО МОДЕЛИРОВАНИЯ ДИСКРЕТНЫХ НЕСТАЦИОНАРНЫХ ПРОСТРАНСТВЕННЫХ ОБЪЕКТОВ[1]

Рассматривается применение идей искусственного интеллекта в задачах ситуационного моделирования и управления дискретными пространственными динамическими системами. Представлен комплекс методов синтеза и количественного сопоставления альтернативных структур и сценариев развития объекта моделирования, синтезированных с учетом предпочтений лица, принимающего решения. В качестве примера реализации указанных методов описана ситуационная система моделирования, ядром которой служат концептуальная модель предметной области и интегрированные с ней экспертная и геоинформационная системы. Модель формализована в парадигме семиотической формальной системы, что позволило дать строгие определения основным понятиям ситуационного управления Д.А. Поспелова, отсутствующие в прототипах, и за счет этого обеспечить унифицированную обработку разнородной информации, автоматизировать детальный анализ консистентности модели и ее оперативную модификацию.

Ключевые слова: промышленно-природный комплекс, концептуальная модель предметной области, структурный синтез, ситуационное управление, интеллектуальная прикладная система.

DOI: 10.31857/S0005231022060113, **EDN:** ADHKCB

1. Введение

В статье рассматривается задача интеллектуализации методов исследования сложных динамических пространственных систем — экосистем, технологий добычи полезных ископаемых, административных территорий и т.п. Они названы автором промышленно-природными комплексами (ППК) и включают в себя промышленные подсистемы, чьи режимы работы нужно оптимизировать по некоторому критерию качества, и природные компоненты, задающие ограничения на работу технических объектов.

[1] Работа выполнена в рамках государственного задания ИИММ КНЦ РАН (НИР «Разработка теоретических и организационно-технических основ информационной поддержки управления жизнеспособностью региональных критических инфраструктур Арктической зоны Российской Федерации», проект № FMEZ-2022-0023).

Актуальность темы связана с порожденным развитием Интернета вещей (IoT) повышением доступности большого количества разнородных временных рядов, содержащих значения различных характеристик составных частей ППК, а также с ростом риска возникновения и масштабов ущерба от комплексных (зависимых) отказов, вызванных усложнением и взаимопроникновением различных ППК, сочетанием случайных факторов и несанкционированных воздействий, в частности, хакерства.

При компьютерных исследованиях сложных систем обычно переходят либо к моделированию средних значений характеристик (энтропии, математических ожиданий и других статистических свойств) [1, 2], либо к анализу их графовых свойств, например, достижимости тех или иных состояний, с помощью сетей Петри [3] или Е-сетей [4]. Если требуется оценивать количественные параметры более детально, необходимо вернуться к причинно-следственным моделям, в той или иной степени отражающим реальные потоки информации, материалов, сигналов (далее будем называть их *ресурсами*), которые циркулируют в системе. Для сложных систем обычно не удается найти достаточно адекватную аналитическую модель, поэтому используют имитационные модели, изоморфные (в заданном масштабе) реальному объекту. С такой целью часто применяют модели системной динамики (например, [5]) или их аналоги, но они не рассчитаны на объекты моделирования с иерархической структурой, вследствие чего быстро теряют консистентность при росте сложности реальной системы.

Построение иерархий возможно по нескольким принципам, основные их них — разделение общей функции системы на подфункции, организационная или структурная декомпозиция и декомпозиция по состояниям. Программное обеспечение для реализации перечисленных видов декомпозиции поддерживается соответственно методами SADT [6], DFD [7] и STD [8]. Для исследования сложных систем необходимо организовать сочетание и совместное применение всех указанных видов декомпозиции, чего не позволяет ни один из упомянутых программных продуктов. Преодолеть эту проблему предполагалось с помощью объектно-ориентированного языка программирования UML [9], но и он не свободен от недостатков, главный — необходимость писать все программные модули в одной среде программирования и встраивать их в диаграмму классов, что затрудняет применение ранее созданных модулей и не допускает применения интеллектуальных процедур обработки информации.

Судя по известным автору публикациям, наиболее близка к описанной в настоящей работе ситуационной системе моделирования (ССМ) [10, 11] интеллектуальная гибридная система поддержки принятия решений РДО («Ресурс-Действие-Объект», см., например, [12]), предназначенная для моделирования сложных систем, функционирование которых требуется представить в форме продукционных правил. Как и в ССМ, в РДО объект моделирования рассматривается как гибкая дискретная неупреждаемая система, непрерывные параметры дискретизируются с шагом, позволяющим достаточ-

но точно описать их изменения, используется дискретно-событийный подход к имитации [13]. Допускается построение иерархических моделей и подключение внешних имитаторов работы составных частей системы. В плане рассматриваемой здесь задачи моделирования ППК недостатки РДО состоят в следующем:

— основным механизмом моделирования служит логический вывод над системой продукций, что снижает эффективность моделирования при росте сложности модели;

— не обеспечивается удобное сопоставление альтернативных вариантов реализации исследуемого объекта;

— иерархия модели не согласуется с организационной структурой объекта моделирования, она либо структурирует продукционные правила с целью повышения скорости логического вывода, либо порождается системой классов в рамках объектно-ориентированного подхода;

— не предусмотрены средства учета географических характеристик исследуемого объекта, которые весьма существенны для ППК.

Общий недостаток известных автору систем моделирования для целей исследования ППК — отсутствие в них средств полноценной реализации всех аспектов ситуационного подхода, наиболее перспективного для решения поставленной задачи.

В связи с изложенным далее описана ССМ, не имеющая отмеченных выше недостатков, которая разработана при участии и под руководством автора в парадигме теории иерархических систем М. Месаровича [14] и теории ситуационного управления Д.А. Поспелова [15] и отличается от аналогов и прототипов в следующем:

— ядро ССМ составляет открытая для оперативной модификации иерархическая ситуационная концептуальная модель (СКМ) ППК, отражающая его организационную структуру и взаимодействия между составными частями. Эта модель позволяет учитывать все вышеупомянутые типы декомпозиции ППК, осуществлять детальный контроль консистентности при создании и корректировках модели, поддерживать современный сценарный подход к имитации, автоматически генерировать исполнительную среду для выполнения вычислительных экспериментов, интегрировать в ССМ подмодели различной природы, включая специализированную экспертную и геоинформационную системы;

— для СКМ конкретизирован ситуационный подход к моделированию, в том числе предложены формальные определения ситуации и сценария, отсутствующие у прототипов, что дало возможность корректно формировать, исследовать и сравнивать ситуации в статике и динамике (в имитационном режиме), реализовать все основные аспекты ситуационного управления [15] и традиционную для интеллектуализированных систем технологию «быстрого прототипа», а также выявлять и синтезировать наиболее предпочтительные для лица, принимающего решения (ЛПР), альтернативы структуры ППК,

обеспечивать детальный контроль корректности модели на всех этапах ее работы;

— в модели не используется нечеткая логика, примененная в ряде реализаций метода ситуационного управления, например, [16]. Это позволяет исследовать ППК с целью поиска так называемых «узких мест», которые порождают наиболее опасные зависимые отказы, поскольку такие отказы появляются при определенных (детерминированных) сочетаниях параметров частей ППК. Кроме того, появляется возможность строить модели нештатных и чрезвычайных ситуаций как расширение модели нормального функционирования ППК, используя один и тот же аппарат.

Повышение мощности вычислительных систем, развитие многопроцессорных методов обработки информации и Интернета вещей, соответствующий рост объема доступной информации о параметрах ППК и т.д., — все это создает новые возможности применения причинно-следственных моделей для исследования сложных систем [17], поскольку лишь такие модели позволяют выявлять зависимые отказы.

2. Синопсис ситуационной системы моделирования

Модель ППК [10, 11] формируется из трех типов сущностей (элементов): объектов $o_i \in O$ $(i = \overline{1, N_O})$, процессов $p_j \in P$ $(j = \overline{1, N_P})$ и ресурсов (данных) $r_k \in R$ $(k = \overline{1, N_R})$ (прописными буквами обозначены множества элементов СКМ). Конструирование модели начинается с построения дерева *объектов* — составных частей ППК, в результате объекты распределяются по L уровням иерархии управления:

$$O = \{o_i\} = \left\{ {}_{\alpha}o^{\gamma}_{\beta_\alpha} \right\} = \bigcup_{\alpha=1}^{N_L} O_\alpha, \tag{1}$$

где $\alpha = \overline{1, N_L}$ – индекс уровня, на котором находится некоторый объект;

$\beta_\alpha = \overline{1, N_\alpha}$ – место объекта на уровне;

$\gamma = \overline{1, N_{\alpha-1}}$ – номер объекта, управляющего данным на предыдущем уровне.

Иерархия (1) отражает организационную структуру ППК, а также пространственное расположение объектов: при разработке модели хотя бы один объект в каждой ветви дерева привязывается к карте с помощью встроенной геоинформационной системы (ГИС). Так устанавливается и поддерживается взаимно-однозначное соответствие между концептуальной схемой и географическими характеристиками ППК, что обеспечивает равноправное использование последних на всех стадиях моделирования. Взаимодействия объектов задаются посредством назначения *ресурсов* — любых информационных или материальных потоков/сигналов, которыми обмениваются объекты. Ресурсы соответствуют переменным в уравнениях модели. Трансформации ресурсов внутри объектов формализуются связыванием каждого из них с рядом *процессов* получения некоторого множества выходных ресурсов из множества

входных ресурсов. При создании модели в нее вводятся структурные альтернативы реализации ППК либо декомпозицией объектов на подобъекты с назначением отношения ИЛИ, либо указанием альтернатив обмена ресурсами (наборами ресурсов) для каких-либо объектов, а процессы приписываются определенным объектам, формируя основные отношения СКМ:

— дерево объектов $H = \bigcup\limits_{\alpha=1}^{N_{L-1}} H_\alpha$, где $H \subseteq O_{\alpha-1} \times B(O_\alpha)$; ($B$ – разбиение множества);

— отношение порождения объектами входных ресурсов процесса $PO \subseteq P \times$ $\times B(O)$;

— отношение порождения процессами выходных ресурсов объекта $OP \subseteq$ $\subseteq O \times B(P)$.

В процессе имитации каждое выполнение одного из процессов переводит модель из текущего состояния в другое. Для компьютерной спецификации этой процедуры любой элемент модели атрибутируется *исполнителем*, определяющим способ реализации этого элемента. У процессов исполнитель организует доступ к имитатору процесса, для ресурсов — задает способ генерации последовательностей значений ресурса, у объектов — формирует их графическое представление. Исполнители процессов могут иметь любую природу (логические, аналитические и др.), но должны моделироваться разностным уравнением. В современной терминологии исполнители процессов реализуют их цифровых двойников, а цифровой двойник объекта получается агрегированием цифровых двойников приписанных к нему процессов с возможностью учета пространственных параметров этого объекта.

Атомарная информация в ССМ — *факт*. Это имя и список значений одного ресурса. Факты делятся на «факты за», в которых перечисляются возможные значения ресурса, и «факты против», задающие недопустимые значения ресурса:

$$\begin{gathered}\text{«факты за»: } r_k \in R_k^+, \text{ где } R_k^+ \subset R_k, \\ \text{и «факты против»: } r_k \notin R_k^-, \text{ где } R_k^- \subset R_k,\end{gathered} \tag{2}$$

а R_k, $k = \overline{1, K}$ – области значений (*домены*) соответствующих ресурсов, дискретные или дискретизированные, поскольку ППК исследуются в классе дискретных систем.

Исходная ситуация есть список фактов вида (2), задаваемых пользователем при постановке задачи сеанса моделирования и трактуемых системой как текущая цель анализа модели. *Полная ситуация* формируется автоматически дополнением исходной ситуации до связного фрагмента модели, содержащего все ресурсы исходной ситуации, и рассматривается как область интересов ЛПР в этом сеансе. Корневой объект такого фрагмента называется *объектом принятия решения* и определяет организационный уровень, которому соответствует данная полная ситуация; она может включать несколько возможных структур ППК из-за введенных ранее альтернатив. Каждая такая безыз-

быточная (не содержащая вариантов) альтернативная структура называется *достаточной ситуацией* (ДС), поскольку соответствует конкретному варианту реализации полной ситуации, из которой она получена, и формирует корректное задание на имитацию. Некоторая достаточная ситуация является *управляющей*, если требует перехода из текущего класса ситуаций в другой путем выбора альтернативной ДС. В таком случае ССМ предлагает ЛПР предпочтительную структуру из нового выбранного им класса следующим образом. Для статического сопоставления ДС с другими ДС того же объекта принятия решений каждому объекту приписывается обобщенный критерий качества (ОКК) [18], представляющий собой среднеквадратическое значение взвешенных относительных отклонений скалярных критериев качества этого объекта от их номинальных значений, задаваемых вышестоящим ЛПР. Он же задает допуски на отклонения скалярных критериев от их номинальных значений, обратные значения этих допусков используются как коэффициенты важности (веса) некоторого скалярного критерия, что согласуется со здравым смыслом. Нулевое значение ОКК возможно только при идеальном функционировании объекта (полном совпадении всех критериев с пожеланиями ЛПР верхнего уровня); ОКК будет равен 1, когда все критерии находятся на грани допусков, и существенно превысит единицу, если режим работы объекта станет недопустимым [19]. Это позволяет быстро выявить источник проблем, найдя самый нижележащий объект с большим значением ОКК. Удельная чувствительность ОКК к изменениям скалярных критериев используется в СКМ для вычисления обобщенных затрат на удовлетворение этого критерия и дает основу для сопоставления ДС в статике, т.е. *классификации ситуаций.* В один класс относятся те ДС, в которых один и тот же скалярный критерий вносит минимальный вклад в ОКК, в пределах одного класса ситуация тем лучше, чем меньше вклад этого критерия в ОКК. При необходимости каждую ДС можно более детально анализировать в имитационном режиме, для чего СКМ автоматически синтезирует соответствующую вычислительную сеть, обеспечивая корректное подключение исполнителей задействованных элементов модели и синтез баз данных для хранения исходных данных и результатов имитации. *Сценарием* в ССМ называется последовательность ДС для одного и того же объекта принятия решений, рассмотренная в одном сеансе имитации при одних исходных данных.

3. Методы искусственного интеллекта в ССМ

Главным средством интеллектуализации процедур моделирования в ССМ служит встроенная статическая продукционная экспертная система (ЭС) с комбинированным (двунаправленным) выводом и двумя решателями — детерминированным и вероятностным. Наряду с СКМ и ГИС, ЭС является базовой подсистемой ССМ, применяемой на всех этапах моделирования и решающей следующие основные задачи:

— доопределение, классификация и обобщение ситуаций [15] с целью выявления наиболее эффективных структур ППК в каждом классе ситуаций.

Результаты этих процедур сохраняются в виде правил ЭС ССМ и используются во всех процедурах обработки ситуаций;

— реализация исполнителей процессов и ресурсов, еще не имеющих программ имитации, с целью построения «быстрого прототипа» модели. Если ЭС назначена исполнителем некоторого ресурса, все его возможные значения должны вычисляться в следствиях продукций ЭС. Если ЭС играет роль имитатора какого-либо процесса, все значения всех его входных ресурсов должны использоваться в предпосылках продукций, а все значения всех его выходных ресурсов — определяться в следствиях продукций;

— сопровождение пространственно-временных функций [20], используемых для учета и корректировки географических характеристик объектов; в некоторых из запросов к ГИС применяются вероятностные форматы данных. В левые части продукций ЭС ССМ можно включать следующие функции этого типа: *в_течение*, выполняющая ретроспективную выборку данных за определенный интервал времени; *соседние*, отбирающая объекты по географическому положению; *сходные*, возвращающая объекты, которые имеют один и тот же заданный набор характеристик; и *ближайший*, определяющая объект, наиболее близкий к заданным координатам. Поскольку все пространственно-временные функции возвращают логическую переменную, показывающую успешность запроса, допускается однократное вложение различных функций, когда результат запроса к «внутренней» функции используется как условие запуска «внешней»;

- анализ чувствительности результатов моделирования к вариациям исходных данных и параметров ППК;

- контроль корректности и разрешимости модели в стиле семиотической формальной системы [15, 20] путем анализа типов и категорий элементов модели, правильности сечений специальных частичных функций, определенных на отношениях СКМ, и предложений вычислимости элементов модели. Частичные функции обозначаются теми же символами, которые использованы в названиях отношений, но строчными буквами, а их сечения по некоторому элементу или подмножеству областей определения — такими же строчными, но жирными, символами (пример см. ниже, теоремы 1 и 2). Разрешимость проверяется в двух аспектах: потенциальная достижимость всех элементов СКМ при завершении каждой итерации ее конструирования и изменения, а также при анализе ситуаций — с целью проверки возможности построения минимального фрагмента СКМ, который описывает текущую полную ситуацию с учетом всех содержащихся в ней альтернатив.

3.1. Семиотический подход к моделированию ППК

Формальная система, описывающая *ядро* ССМ (СКМ + ЭС + ГИС), включает:

— *алфавит* из: символов, образующих имена элементов модели; функциональных и предикатных символов, обозначающих операции, отношения и связи этих элементов; специальных и синтаксических символов;

— *множество формул*, включающих символы алфавита и соотношения для вычисления функций и множеств, определенных в схеме СКМ;

— *выражения вычислимости* для всех элементов модели;

— *аксиомы вычислимости* всех ресурсов, внешних по отношению к ядру ССМ;

— *правила вывода*: следование с равенством

$$F_1, F_1 = F_2, \quad F_2 \Rightarrow F_3 \mid\!\!- F_3 \tag{3}$$

и непосредственное следование

$$F_1, F_1 \Rightarrow F_2 \mid\!\!- F_2. \tag{4}$$

В (3) и (4) $F_{(*)}$ — любые формулы, построенные по правилам этой формальной системы.

Концептуальная модель анализируется «снизу-вверх» циклами из следующих шагов.

Шаг 1. По правилу (4) выводятся все следствия из формул и аксиом.

Шаг 2. По правилам (3), (4) выводятся все следствия из аксиом и формул из шага 1.

Шаг 3. По предложениям вычислимости расширяется список вычислимых объектов.

Ситуационная концептуальная модель относится к семиотическим моделям [15], поскольку процедуры обработки ситуаций (доопределение исходной ситуации до полной; классификация ситуаций по ОКК и когнитивная классификация, представленная далее в разделе 3.3; обобщение ситуаций по прагматическим признакам) используют три соответствующие группы логико-трансформационных правил, которые хранятся в ЭС ССМ и оперативно меняют состав модели.

Применение унифицированных списковых форматов для представления ресурсов различных типов (числовых, строковых и др.) дает возможность вычислять одни и те же параметры аналитическими, логическими процедурами, а также средствами ГИС, что относится к элементам новизны описываемой системы моделирования.

Существенная часть контроля корректности СКМ — проверка условий, гарантирующих иерархичность структуры модели и, в частности, обеспечивающие выполнение принципа приоритета действия [14] для иерархических структур. Рассмотрим их несколько подробнее. Дадим их определения, а затем контролируемые соотношения (теоремы, доказательства которых [11] опущены для экономии места).

Анализ корректности структуры модели основан на отношениях порождения.

Определение 1. Отношение $OO \subseteq O \times B(O)$, *которое сопоставляет каждый объект модели всем другим объектам, чьи выходные ресурсы* (*без*

промежуточных объектов, непосредственно) подаются на входы данного объекта, называется отношением порождения объектов.

Теорема 1. $\boldsymbol{oo}(o_i) = \boldsymbol{po}(\boldsymbol{oa}(o_i))\backslash\{o_i\}$,

где символ $\backslash$ *– операция взятия разности множеств; отношение PO введено выше, а отношение* $OA \subseteq O \times B(P)$ *связывает объект со множеством его внутренних процессов.*

Определение 2. Отношение $PP \subseteq P \times B(P)$*, которое сопоставляет каждый процесс модели всем другим процессам, чьи выходные ресурсы (без промежуточных процессов, непосредственно) подаются на входы данного процесса, называется отношением порождения процессов.*

Теорема 2. $\boldsymbol{pp}(p_i) = \boldsymbol{op}(\boldsymbol{po}(p_i))\backslash\{oa^{-1}(p_i)\}$.

Соотношения теорем 1 и 2 проверяются для поиска циклов по ресурсам. Цикличность не всегда означает некорректность в СКМ, вызванные этим конфликты определяются в ходе имитации. Например, в приложениях, касающихся экономических связей между объектами модели, цикличность взаимодействий относится к их нормальным и часто встречающимся свойствам. Поэтому циклы, найденные с применением теорем 1 и 2, выдаются для утверждения либо запрета пользователю, который учитывает семантику ППК. Автоматически запрещаются только пути между элементами альтернативных ДС, а также объектами-синонимами и объектами из их множеств подчиненности.

Кроме ограничений, содержащихся в теоремах 1 и 2, при контроле СКМ проверяются соотношения другого типа, связанные с корректностью структуры иерархии. В частности, принцип приоритета действия [14] запрещает передачу материальных ресурсов от суперобъектов их подобъектам, поскольку подобные связи «обходят» межуровневые процессы настройки и агрегирования данных, что способно нарушить координируемость элементов иерархии. Тот же принцип ограничивает отношения следования между параметрами межуровневых процессов. Еще одна группа ограничений связана с выбором того или иного типа декомпозиции в иерархии (многоэшелонной, многослойной или стратифицированной [14]). Все структурные ограничения проверяются в СКМ с помощью транзитивных замыканий отношений порождения (см. определения 1 и 2) OO^* и PP^*. Так как процессы определяют динамику моделирования и порождения ресурсов в СКМ, с каждым процессом связана процедура контроля, обеспечивающая корректность работы модели в ходе имитации. Этот набор процедур контроля основан на новом определении *шаблона процесса* как подструктуры модели. Шаблон дает концептуальное описание преобразования множества входных ресурсов от других объектов внутри того объекта, к которому приписан процесс-владелец шаблона.

Структура схемы шаблона та же, что и у схемы модели в целом [11]:

$$S_{\text{Ш}} =< O_{\text{Ш}}, P_{\text{Ш}}, H_{\text{Ш}}, OP_{\text{Ш}}, PO_{\text{Ш}} >, \tag{5}$$

где $O_{\text{Ш}} \subset O$, $P_{\text{Ш}} \subset P$, $H_{\text{Ш}} \subset H$, $OP_{\text{Ш}} \subseteq O_{\text{Ш}} \times B(P_{\text{Ш}}) \subset OP$, $PO_{\text{Ш}} \subseteq P_{\text{Ш}} \times \times B(O_{\text{Ш}}) \subset PO$.

Теорема 3. $O_{Ш} = \boldsymbol{po}(\boldsymbol{pp}(p_i))$;

$$P_{Ш} = (\boldsymbol{pp}(p_i)) \cup (\boldsymbol{oa}(\boldsymbol{po}(p_i)) \cap (\boldsymbol{pp}^*(p_i)) \cup (\boldsymbol{pp}^*)^{-1}(p_i)).$$

Теоремы 1–3 позволили доказать следующие утверждения, значительно упростившие процедуры анализа СКМ при ее разработке и оперативной модификации.

Теорема 4. Ситуационная концептуальная модель разрешима, если:

— она корректна по правилам назначения имен и отношений;

— разрешим каждый специфицированный в модели шаблон для внутренних процессов и процессов агрегирования.

Теорема 5. Если успешно проведены все процедуры контроля, то:

— ситуационная концептуальная модель консистентна (*связна и непротиворечива*);

— возможен автоматический синтез исполнительной среды для имитации.

Далее проверяются свойства межуровневых процессов, необходимые для корректности иерархии объектов (согласованности и координируемости взаимодействий объектов модели). Процедура проверки основана на отношениях следования, в том числе заданных неявно.

На этой стадии анализа модели исходят не от объектов или процессов, как на вышеописанных этапах, а от конкретного ресурса (настроечного параметра или выходного значения функции качества какого-либо объекта). Цель проверки — запрет циклов по таким ресурсам, поскольку они относятся к информационным ресурсам, и их вычисление не итеративно. Здесь также используются сечения транзитивных замыканий отношения PP (см. определение 2), начиная от того процесса p_i, на выходе которого имеется данный ресурс. На каждой итерации вычисления сечений расширяется множество процессов P_{k+1}, для которых строятся эти сечения отношения порождения процессов (см. теорему 2): в него включаются процессы, найденные на предыдущей итерации,

$$P_0(p_i) = \{p_i\};$$
$$P_{k+1}(p_i) = P_k(p_i) \cup \boldsymbol{pp}(P_k(p_i)). \tag{6}$$

Вычисление сечений (6) прекращается и декларируется конфликт, если на каком-либо шаге k множество $P_k(p_i)$ содержит процессы, соответствующие условию констатации конфликта, которое установлено заранее. Если конфликт не возникает, расчет (6) продолжается до совпадения множеств $P_{k+1}(p_i)$ и $P_k(p_i)$, что считается (для выбранного ресурса) отсутствием конфликтов.

Доказано, что принцип приоритета действий [14] выполняется, если не диагностировано *основное условие конфликта*, которое описывает на языке СКМ отсутствие иерархических связей по материальным ресурсам, идущих вдоль связей подчиненности:

$$(7) \quad (\exists p_m \in P_{k+1}(p_i)) \left(\left(oa^{-1}(p_m) \in O_\alpha \right) \wedge \left(oa^{-1}(p_i) \in O_\beta \right) \wedge (\beta < \alpha) \right).$$

Это необходимое условие корректности иерархии.

Дополнительные условия конфликта не обязательны, их можно задавать для реализации различных типов иерархий:

— «иерархия неравенства», в которой запрещен обмен между объектами одного уровня иерархии, при этом в (7) вместо строгого неравенства ставится нестрогое:

$$(8) \quad (\exists p_m \in P_{k+1}(p_i)) \left(\left(oa^{-1}(p_m) \in O_\alpha \right) \wedge \left(oa^{-1}(p_i) \in O_\beta \right) \wedge (\beta \leq \alpha) \right);$$

— «строгая вертикаль», запрещающая связь объектов различного подчинения, когда множества (6) могут содержать только процессы из таких пар объектов, один из которых входит во множество подчиненности другого, т.е. является его предком или потомком в иерархии (1). В подобной структуре дополнительное условие конфликта имеет вид:

$$(9) \quad (\exists p_m \in P_{k+1}(p_i)) \left(\left(oa^{-1}(p_m) \notin \underline{\underline{\boldsymbol{h}}} \left(oa^{-1}(p_i) \right) \right) \right),$$

где $\underline{\underline{\boldsymbol{h}}}(*)$ – множество подчиненности объекта(ов), специфицированного(ых) в скобках;

— «стратифицированная иерархия», где обмен информацией допустим только для объектов на соседних с исходным уровнях (их количество задается параметром $\Delta\alpha$), тогда в (6) допускаются только процессы из подобных объектов. Дополнительное условие конфликта в таком случае аналогично (7), но с другим неравенством:

$$(10) \quad (\exists p_m \in P_{k+1}(p_i)) \left(\left(oa^{-1} \right)(p_m) \in O_\alpha \right) \wedge \left(oa^{-1}(p_i) \in O_\beta \right) \wedge (\beta > \alpha + \Delta\alpha).$$

Условия (7)–(10) (если это не ведет к противоречиям) можно назначать совместно, они позволили доказать описательные возможности СКМ в виде следующей теоремы.

Теорема 6. Все основные типы иерархий, описываемые теорией [15] (*эшелоны, страты, слои*), *реализуемы в структурах вышеописанной ситуационной модели.*

3.2. Управление выводом в СКМ

С учетом специфики решаемых задач и форматов представления информации для экспертной системы ССМ разработана оригинальная динамическая стратегия управления комбинированным логическим выводом, учитывающая внутреннюю структуру знаний для ускорения вывода. Моделирование ведется в универсуме $U = R_1 \times R_2 \times \ldots \times R_k$. Рассмотрим для примера

детерминированный вывод, поскольку он применяется в режиме имитации, т.е. наиболее часто. Здесь перебор вариантов можно сократить, «сужая» домен ресурса при появлении в БД ЭС «факта против», например, на l-м шаге вывода:

$$R_k^{l+1} = R_k^l \backslash R_k^-, \tag{11}$$

поэтому далее принято, что в БД есть только «факты за» из (2).

Детерминированные продукции в экспертной системе ССМ имеют вид

$$P_s: \ A_s \supset C_s, \quad s = \overline{1,S}, \tag{12}$$

где A_s и C_s – соответственно антецедент и консеквент продукции, представляющие собой конъюнкции конечного количества фактов вида (2). Если какие-либо из них имеют формат «факта против», они для удобства сравнения с фактами в БД при выводе преобразуются в формат «фактов за» вычислением дополнения запрещаемого ими подмножества значений ресурса относительно текущего значения его домена. Цель вывода по структуре аналогична A_s и C_s в (12):

$$G_l = G_{1l} \wedge G_{2l} \wedge \cdots \wedge G_{ml} \wedge \cdots \wedge G_{Ml}, \tag{13}$$

где G_{ml}, $m = \overline{1,M}$, имеют вид (2).

Общая идея управления выводом аналогична методам наискорейшего спуска: при неоднозначности выбора продолжать вывод для тех фактов и продукций из конфликтного множества (при прямом выводе) или конъюнктов цели (при обратном выводе), которые в максимальной степени снижают уровень недоопределенности текущего состояния вывода.

Экспертная система начинает работать в режиме обратного вывода, его особенность в ССМ состоит в том, что с C_s сравниваются не все G_m, а только содержащие переменные, о которых есть факты (2) в БД ЭС. Если таковых нет, ЭС переходит в режим прямого вывода. При этом эвристическая стратегия ускорения вывода реализуется путем разрешения конфликта выбором из конфликтного множества

$$CS_l = \{P_{1l}, P_{2l}, \ldots, P_{sl}, \ldots, P_{Nl}\} \tag{14}$$

той продукции, после применения которой останется неопределенным минимальное количество сочетаний значений «общих» переменных (ресурсов), содержащихся как в консеквенте этой продукции, так и в конъюнктах текущей цели (13). Для каждого из таких ресурсов вычисляются разности множеств значений R_k^+ в консеквенте продукции и соответствующем конъюнкте цели, а затем рассчитываются мощности полученных разностей. Выполняется та продукция из (14), для которой минимальна мощность декартова произведения этих разностей для входящих в нее «общих» ресурсов.

Если значения ресурсов в домене упорядочены (например, это дискретные или дискретизированные значения числового ресурса), то в соотношениях (2)

и (12) можно использовать знаки «предшествует» и «следует за», аналогичные отношениям «больше» и «меньше» в арифметике. Такие знаки позволяют ускорить сокращение доменов ресурсов (11) и, соответственно, укоротить цепочки вывода.

Тот же подход применим к вероятностному выводу, предусмотренному в ЭС ССМ, здесь он не приводится из-за громоздкости математических формулировок.

По известным причинам (отсутствие числовых функций качества интеллектуальных систем) нет возможности дать аналитическую оценку эффективности рассмотренной стратегии комбинированного вывода, в тестовых прогонах результат достигался на 10–40% быстрее, а для упорядоченных доменов выигрыш достигал 70%.

3.3. Когнитивная классификация ситуаций в СКМ

Кроме организации работы встроенной ЭС, методы искусственного интеллекта используются в ССМ для решения одной из базовых задач ситуационного управления [15] — задачи классификации ситуаций. Проблема в том, что результаты классификации с помощью представленного выше числового ОКК зависят не только от самой структуры объекта, но и от значений ресурсов. В ходе классификации ситуаций по ОКК строится основное отображение ситуационного управления, в котором каждому классу ситуаций (все ситуации этого класса считаются эквивалентными) сопоставляется некоторое управляющее воздействие. Как уже говорилось, в ССМ таким воздействием является изменение структуры ППК путем его перевода из текущей ДС в одну из ДС (назовем ее желаемой ситуацией) нового класса, выбранного по предпочтениям ЛПР. Рекомендовать ЛПР в качестве желаемой имеющуюся в архиве ситуаций ССМ оптимальную (по ОКК) ситуацию из нового класса не представляется разумным, поскольку она была выявлена при определенных значениях ресурсов, не (или очень редко) совпадающих с их текущими значениями. Поэтому требуется строить основное отображение, учитывая только структурные особенности ДС и абстрагируясь от значений ресурсов. С такой целью предложено применить в ССМ когнитивный подход [21] к построению классов, где экземпляры каждого класса не равноправны, как в «классической» классификации на основе отношения эквивалентности, но делятся на более или менее типичные по степени сходства с наиболее типичным представителем класса — *прототипом.* Построение основного отображения, инвариантного к значениям ресурсов (*когнитивного отображения ситуаций*), выполняется следующим образом. При классификации ситуаций по ОКК в архиве сохраняется не только оптимальная ДС каждого класса (она, очевидно, есть прототип в этом классе), но и несколько близких к ней (субоптимальных) ДС, у которых значение ОКК несущественно (на 10–15%) ниже, чем у прототипа. Затем вычисляется семантическое расстояние [22] между прототипами и субоптимальными ДС разных классов, и когнитивное отображение формируется из тех пар ситуаций, семантическое расстояние между кото-

рыми минимально. Полученное отображение не только перестает зависеть от значений ресурсов, но и позволяет минимизировать изменения текущей структуры, чтобы уменьшить возмущения в реальной системе при переводе ППК из одного класса ситуаций в другой (подобные задачи возникают при минимизации ripple effect [23] в цепях поставок). Осталось лишь разработать подходящую меру семантической близости ДС с учетом их иерархической структуры.

Таких мер известно достаточно много (например, [22]), в основном это *геометрические меры*, где экземпляр объекта формализуется точкой в пространстве свойств, и собственно *семантические метрики* для иерархических структур (онтологий, таксономий и т.п.), использующие расстояние между вершинами графа объекта. Последние не годятся в рассматриваемой здесь задаче, так как требуется иметь интегральную оценку различий между ДС, не связанную непосредственно с маршрутами между вершинами графа модели. Поэтому для ССМ предлагается модифицировать нормализованную модель Тверски [24], где расстояние между объектами a и b дается формулой

$$S(a,b) = \frac{f(A \cap B)}{f(A \cap B) + \alpha f(A \backslash B) + \beta f(B \backslash A)}. \tag{15}$$

В (15) A и B – множества имен свойств объектов; f – знакоположительная функция; α и β – положительные веса, назначаемые свойствам, которые одинаковы и различны у оцениваемых объектов. Обычно f – мощность множества-аргумента. Непосредственно модель (15) неприменима в ССМ, поскольку не позволяет учесть иерархичность структуры ситуации. Очевидно, в иерархии различия нижележащих уровней должны меньше влиять на расстояние между двумя ситуациями, чем различия на верхних уровнях. Поэтому расстояние между двумя иерархическими структурами предложено вычислять на базе модифицированной *иерархической модели Тверски*:

$$S(a,b) = \frac{1}{N_L} \sum_{\alpha=1}^{N_L} \frac{1}{\alpha} S_\alpha(a_\alpha, b_\alpha), \tag{16}$$

где S_α определяются согласно (15) для подмножеств свойств объектов на уровне α.

Поскольку (16) есть линейная свертка нескольких нормализованных моделей Тверски, она сохраняет свойства функций меры. Взвешивая слагаемые в (16), можно вводить экспертные предпочтения по уровням иерархии.

4. Приложения системы ситуационного моделирования

К настоящему моменту исследованы следующие области применения ССМ [11, 20]: улучшение процесса переработки минерального сырья на основе технологического графа для хибинских руд сложного вещественного состава; ретроспективный анализ состояния лесных экосистем с целью мониторинга

их загрязнения в условиях техногенного воздействия; оценка удароопасности горного массива в зоне горных работ; моделирование среднесрочных задач диспетчера в энергетике; выбор технологий обогащения сложного минерального сырья; технико-экономическая оценка рудных месторождений; исследование безопасности электроэнергетических систем путем защиты электрических сетей от перенапряжений. Приведем их краткое описание.

Задача оптимизации процесса переработки минерального сырья решалась в основном с помощью СКМ, отображающей технологический граф добычи и комплексной переработки руды как совокупности различных схем переработки со связями по их входным и выходным продуктам. Эти схемы включаются в граф согласно стадии освоения, которая должна соответствовать заданному периоду планирования. Приложение ССМ разрабатывалось для апатито-нефелиновой руды при долгосрочном планировании и формировала план переработки (сетевой график) — длительности и интенсивности применения технологий из заданного набора технологических схем — для Хибинского месторождения по критерию минимума суммарных затрат либо максимуму прибыли с учетом имеющихся ограничений на ресурсы и спрос по продуктам переработки. ГИС в этой задаче не применялась, перебор вариантов шел по ветвям технологического графа.

Система ретроспективного анализа вреда антропогенных выбросов по состоянию лесных массивов отображала динамику ущерба от выбросов в промышленных районах Мурманской области (города Никель и Мончегорск) на топографической основе путем сравнения параметров хвойных деревьев из загрязненных районов с биологически схожими деревьями из чистых зон (Кандалакшский заповедник). Разработанное приложение ССМ показало ухудшение ситуации после перехода металлургического производства в Мончегорске с местного на более загрязненное серой норильское сырье. В решении этой задачи участвовали только геоинформационная и экспертная система ССМ.

Наиболее успешной была апробация ССМ для проблемы прогнозирования горных ударов на Юкспорском руднике АО «Апатит». В приложении ССМ рудное тело разбивалось на условные элементы — параллелепипеды, текущее состояние каждого из них оценивалось вектором результатов прогноза по данным нескольких (от 3 до 9) независимых частных методик, список которых задавали маркшейдеры. Задачу комплексного прогноза решала ЭС ССМ путем логического вывода на вероятностных правилах, разработанных с участием экспертов. В течение восьми лет это приложение было штатной частью служб Кировского рудника и позволило вдвое снизить количество «ложных тревог», когда работа на участке останавливалась, но горный удар не происходил.

В каждом из указанных приложений апробировались различные аспекты представленной здесь информационной технологии, однако, к сожалению автора, до сих пор не удалось исследовать все основные идеи разработанного подхода в одном приложении; это предполагается сделать в будущем.

В целом, приложения ССМ позволили сократить сроки принятия решения при изменении ситуации и повысить объективность решений из-за улучшения информационной обеспеченности ЛПР.

5. Заключение

Рассмотрены некоторые аспекты разработки ситуационного подхода к моделированию сложных динамических пространственно-распределенных комплексов, преимущественно использующие методы искусственного интеллекта. Подробно рассмотрены динамическая стратегия управления детерминированным комбинированным выводом в ЭС ССМ и когнитивный метод классификации ситуаций, минимизирующий возмущения при необходимости перевода ППК из текущего класса ситуаций в выбранный ЛПР новый класс, для которого предпочтительная структура реализации ППК выбирается системой моделирования по результатам классификации ситуаций.

Изложенное показывает, что ССМ относится к прикладным системам искусственного интеллекта и позволяет строить гибкие системы моделирования промышленно-природных комплексов различного масштаба. В настоящее время просматриваются два основных направления развития этой системы. Первое — разработка исследовательских информационно-аналитических ситуационных систем моделирования, подведомственных региональным либо муниципальным (в больших городах) властям и ориентированных на аккумулирование разнородных знаний о характеристиках ППК на основе пространственной привязки такой информации. Цель функционирования подобных моделей заключается в профилактике нештатных и чрезвычайных ситуаций на базе предсказательной аналитики безопасности модификации ППК и включения в него новых компонентов с использованием имитаторов (цифровых двойников) подсистем ППК. Второе направление — создание систем поддержки принятия повторяющихся решений в специализированных ситуационных центрах с помощью проблемно-ориентированных ССМ, которые автоматизированно генерируются на базе исследовательских ССМ и предоставляют персоналу ситуационного центра альтернативу проведению экспертных советов в регионах, где их затруднительно или невозможно организовать.

Возможность полноценной реализации предложенного подхода к моделированию требует дополнения рождающегося IoT новой компонентой — Интернетом моделей вещей (Internet of Models of Things – IoMoT), т.е. переходом к Интернету вещей и моделей — Internet of Things and Models (IoTaM). Новая компонента Интернета должна обеспечивать:

— разработку для появляющихся устройств и систем их локализованных ситуационных моделей (цифровых двойников), унифицированных по интерфейсам и форматам представления входной и выходной информации, и сертификация таких моделей;

— создание системы поиска существующих моделей по их концептуальному описанию, с целью минимизации дублирования разработок;

— организацию возможности глобального применения этих моделей по принципам, сходным с GRID-технологиями, что также даст возможность решать вопросы авторских прав разработчиков и, если это требуется, обеспечивать режимность локальных моделей.

Автор признателен О.П. Кузнецову и рецензентам работы за детальный анализ текста при подготовке к публикации, что позволило существенно улучшить качество материала.

СПИСОК ЛИТЕРАТУРЫ

1. *Попков Ю.С.* Теория макросистем. Равновесные модели. М.: Либроком, 2013.
2. *Fishman G.S.* Monte Carlo: Concepts, Algorithms, and Applications. Springer, 1996.
3. *Котов В.Е.* Сети Петри. М.: Наука, 1984.
4. *Шаньгин В.Ф., Костин А.Е., Илюшечкин В.М. и др.* Программирование микропроцессорных систем. М.: Высш. шк., 1990.
5. *Graham Alan K.* Parameter Estimation in System Dynamics Modeling / Jorgen Randers (ed.). Elements of the System Dynamics Method. Portland, 1980. P. 143–161.
6. *Nam Pyo Suh.* Axiomatic Design – Advances and Applications. Chapter 5. New York: Oxford University Press, 2007. P. 239–298.
7. *Коваленко В.В.* Проектирование информационных систем. М.: Форум, 2011.
8. Joe Celko's Trees and Hierarchies in SQL for Smarties. The Morgan Kaufmann Series in Data Management Systems. Second Edition. Elsevier Inc., 2012.
9. *Hunt J.* The Unified Process for Practitioners: Object-oriented Design, UML and Java. Springer, 2000.
10. *Фридман А.Я., Курбанов В.Г.* Ситуационное моделирование надежности и безопасности промышленно-природных систем // Информационно-управляющие системы. 2014. № 4(71). С. 1–10.
11. *Фридман А.Я., Курбанов В.Г.* Формальная концептуальная модель промышленно-природного комплекса как средство управления вычислительным экспериментом // Труды СПИИРАН. 2014. № 6(37). С. 424–453.
12. *Ясиновский С.И.* Интеллектуальная гибридная система поддержки принятия решений РДО: вчера, сегодня, завтра // Мягкие измерения и вычисления. 2018. № 12(13). С. 33–50.
13. *Месарович М., Такахара Я.* Общая теория систем: математические основы. М.: Мир, 1978.
14. *Месарович М., Мако Д., Такахара И.* Теория иерархических многоуровневых систем. М.: Мир, 1973.
15. *Поспелов Д.А.* Ситуационное управление: теория и практика. М.: Наука, 1986.
16. *Борисов В.В., Зернов М.М.* Реализация ситуационного подхода на основе нечеткой иерархической ситуационно-событийной сети // Искусственный интеллект и принятие решений. 2009. № 1. С. 17–30.
17. *Микони С.В., Соколов Б.В., Юсупов Р.М.* Квалиметрия моделей и полимодельных комплексов. М.: РАН, 2018.
18. *Салуквадзе М.Е.* Задачи векторной оптимизации в теории управления. Тбилиси: Мецниереба, 1975.

19. *Фридман А.Я.* Координация и планирование управлений в локально организованных иерархических системах // Шестая Международная конференция «Системный анализ и информационные технологии» САИТ-2015 (15–20 июня 2015 г., г. Светлогорск, Россия): Тр. конф. В 2 т. Т. 1. М.: ИСА РАН, 2015. С. 115–124.
20. *Фридман А.Я.* Интерпретация концептуальной модели пространственного динамического объекта в классе формальных систем // Вестник КНЦ РАН. 2015. № 4(23). С. 100–112.
21. *Кузнецов О.П.* Когнитивная семантика и искусственный интеллект // Искусственный интеллект и принятие решений. 2012. № 4. С. 32–42.
22. *Крюков К.В., Панкова Л.А., Пронина В.А. и др.* Меры семантической близости в онтологии // Control Sciences. № 5. 2010. С. 2–14.
23. *Yuhong Li et al.* Ripple Effect in the Supply Chain Network: Forward and Backward Disruption Propagation, Network Health and Firm Vulnerability // Eur. J. Oper. Res. 2021. V. 291. No. 3. P. 1117–1131.
24. *Tversky A.* Features of similarity // Psycholog. Rev. 1977. V. 84. No. 4. P. 327–352.

Статья представлена к публикации членом редколлегии О.П. Кузнецовым.

Поступила в редакцию 07.12.2021

После доработки 04.01.2022

Принята к публикации 26.01.2022

С О Д Е Р Ж А Н И Е

Тематический выпуск

CONTENTS

Topical issue