А. А. Широкий, А. О. Калашников, Применение методов естественных вычислений для управления рисками сложных систем, *Пробл. управл.*, 2021, выпуск 4, 3–20

DOI: https://doi.org/10.25728/pu.2021.4.1

УДК 519.711+004.8 DOI: http://doi.org/10.25728/pu.2021.4.1

ПРИМЕНЕНИЕ МЕТОДОВ ЕСТЕСТВЕННЫХ ВЫЧИСЛЕНИЙ ДЛЯ УПРАВЛЕНИЯ РИСКАМИ СЛОЖНЫХ СИСТЕМ

А.А. Широкий, А.О. Калашников

Аннотация. Представлен обзор моделей и методов естественных вычислений в контексте их применимости к решению задач управления рисками сложных систем. Обоснована эквивалентность задачи минимизации рисков задаче эффективного управления. Приводится общая постановка задачи управления рисками сложных систем в условиях неопределённости, описана структура фундаментальных и прикладных задач управления рисками. Наиболее известные модели и методы естественных вычислений кратко описаны и проанализированы с точки зрения их применимости для решения задач управления рисками по критериям формализма, универсальности и способности к обучению. Проведён анализ предпочтений научного сообщества в части применения тех или иных моделей и методов естественных вычислений для решения различных классов задач управления рисками. Сделаны выводы о перспективных с этой точки зрения подходах, которым в настоящий момент, на взгляд авторов, уделяется недостаточно внимания.

Ключевые слова: управление рисками, задача эффективного управления, естественные вычисления.

ВВЕДЕНИЕ

Многие задачи управления сложными системами могут быть сведены к оценке и минимизации рисков для управляемой системы или интерпретированы в этих терминах. Задача управления рисками является комплексной и предполагает решение подзадач различных классов, включая задачи идентификации, мониторинга, прогноза, оптимизации и др. В условиях непрерывной циркуляции объёмных потоков данных эффективные решения невозможны без построения достаточно сложных моделей управляемых систем и применения адекватных вычислительных методов. Реально существующие динамические системы, для которых ставятся такие задачи, являются открытыми – соответственно, применяемые методы также должны обеспечивать способность моделей адаптироваться к непрерывно изменяющимся (в том числе непрогнозируемо) условиям внешней среды.

В связи с этим напрашивается аналогия с процессами, протекающими в живой природе, – живые организмы постоянно решают задачу выживания в условиях переменного окружения, являющегося источником как предсказуемых, так и непрогнозируемых угроз. Анализ моделей поведения живых организмов, структур группового управления, механизмов защиты от внешних угроз представляется весьма перспективным ввиду заведомой работоспособности таких моделей (чему в природе есть множество примеров), а также их высокой эффективности, достигнутой в ходе эволюционных процессов. В настоящий момент активно развивается такая научная область, как природные (или естественные) вычисления (natural computing), соединяющая в себе аспекты связности элементов, группового поведения и эмерджентности. В книге [1] термин «*естественные вычисления*» определяется как область исследований, изучающая модели и вычислительные методы, вдохновлённые природой, а также рассматривающая происходящие в природе явления с точки зрения обработки информации.

Многие подразделы естественных вычислений (например, искусственные нейронные сети) сейчас находятся на пике популярности, другие (к примеру, бактериальные и ДНК-вычисления) изучены сравнительно мало. Настоящая работа представляет собой обзор основных известных классов алгоритмов природных вычислений с анализом их применимости в задачах управления рисками. Отметим, что описание самих подходов не является целью авторов.

Структура изложения материала такова. В § 1 сформулирована общая постановка задачи управления рисками сложных систем в условиях неопределённости и показана её эквивалентность задаче эффективного управления. В § 2 рассматриваются фундаментальные и прикладные задачи управления рисками сложных систем, делается вывод о том, какие из них могут быть решены методами естественных вычислений. В § 3 приводятся возможные основания классификации естественных вычислений и выделяются те из них, которые важны для оценки применимости описываемых алгоритмов и методов к задачам управления рисками сложных систем. В § 4 приведён обзор известных методов и алгоритмов естественных вычислений и их классификация по основаниям, отобранным в § 3. В § 5 делаются выводы о перспективных с точки зрения решения задач управления рисками подходах, которым в настоящий момент, на взгляд авторов, уделяется недостаточно внимания.

1. ОБЩАЯ ПОСТАНОВКА ЗАДАЧИ УПРАВЛЕНИЯ РИСКАМИ СЛОЖНЫХ СИСТЕМ В УСЛОВИЯХ НЕОПРЕДЕЛЁННОСТИ

В настоящей работе под сложными системами будем понимать системы с бесконечным разнообразием реакций на внешние воздействия. В самой общей постановке задача управления такими системами заключается в нахождении управляющих воздействий, переводящих их в целевое состояние. Такие управляющие воздействия называют *эффективными*. Формально эту задачу можно записать следующим образом.

Пусть U – множество наборов управляющих воздействий, Q и Θ – множества состояний системы и внешней среды соответственно, причём состояние системы $q \in Q$ однозначно определяется состоянием внешней среды и применяемыми управляющими воздействиями, т. е. $q = q(u, \theta)$, $u \in U$, $\theta \in \Theta$. Введём функционал $K = K(u, q, \theta) = K(q(u, \theta))$ – *целевой функции* управления u при состоянии системы q и состоянии внешней среды θ. Задача заключается в поиске *оптимальных в широком смысле* управляющих воздействий $u^* \in U$, т. е. таких, что

$$K(u^*, q, \theta) = \max_{u \in U} K(u, q, \theta). \qquad (1)$$

В вырожденном случае, когда управляющей системе известны актуальные состояния управляемой системы $q_0 \in Q$ и внешней среды $\theta_0 \in \Theta$, целевая функция является функцией одной переменной u и задача (1) записывается так:

$$K(u^*, q_0, \theta_0) = \max_{u \in U} K(u, q_0, \theta_0). \qquad (2)$$

Допустим, что существует набор (возможно, не единственный) аргументов (u^*, q^*, θ^*), при котором целевая функция принимает максимально возможное значение (*идеальная ситуация*):

$$K(u^*, q^*, \theta^*) = \max_{u \in U} \max_{q \in Q} \max_{\theta \in \Theta} K(u, q, \theta).$$

Величину

$$\varphi(u, q, \theta) = K(u^*, q^*, \theta^*) - K(u, q, \theta),$$

равную разности максимального и текущего (при некоторых u, q и θ) значения целевой функции назовём *функцией потерь*. Легко видеть, что

$$\min_{u \in U} \varphi(u, q, \theta) = K(u^*, q^*, \theta^*) - \max_{u \in U} K(u, q, \theta),$$

т. е. задача максимизации целевой функции эквивалентна задаче минимизации функции потерь. В детерминированном случае эквивалентность также сохраняется:

$$\min_{u \in U} \varphi(u, q_0, \theta_0) = K(u^*, q^*, \theta^*) - \max_{u \in U} K(u, q_0, \theta_0). \qquad (3)$$

В реальных постановках полная информированность встречается редко, главным образом, при управлении техническими системами. Чаще всего управляющая система не имеет полной и/или достоверной информации о состоянии управляемой системы и внешней среды. В этом случае для решения задачи необходимо тем или иным образом устранить неопределённость. Например, можно выбирать наиболее вероятные состояния, наименее благоприятные состояния или действовать иным образом – см., например, статью [2].

Зафиксируем некоторое управляющее воздействие $u \in U$. Зададим устраняющий неопределённость оператор $\mathfrak{I}$, т. е. такой, что результат его применения к целевой функции зависит только от выбранного управления u:

$$\mathfrak{K}(u) = \mathfrak{I}K(u, \cdot, \cdot).$$

Тогда, по аналогии с выражением (2), можем записать задачу поиска оптимального в широком смысле управления u^*, максимизирующего значение целевой функции в условиях неопределённости:

$$\mathfrak{K}(u^*) = \max_{u \in U} \mathfrak{K}(u). \qquad (4)$$

Зададим действительнозначную *функцию риска*

$$\rho(u) = \mathfrak{K}(u^*) - \mathfrak{K}(u).$$

В общем случае *риск* часто определяют как системный параметр, свойство системы управления, в частности ЛПР, принимать решения в условиях неопределённости, которые могут повлечь за собой как нежелательные (опасные), так и существенно выигрышные последствия (см., например,

работу [3]). В рассматриваемой постановке задачи введем абсолютный максимум значения целевой функции, поэтому выигрышных последствий быть не может и значения $\rho(u)$ интерпретируются как нежелательный риск, связанный с применением управляющего воздействия u. По аналогии с выражением (3) поставим задачу минимизации риска:

$$\rho\left(u^*\right) = \min_{u \in U} \rho(u). \qquad (5)$$

Очевидно, что

$$u^* = \operatorname*{Argmax}_{u \in U} \mathcal{R}(u) = \operatorname*{Argmin}_{u \in U} \rho(u). \qquad (6)$$

Таким образом, в случае управления сложной системой в условиях неопределённости задача ***максимизации целевой функции*** (4), при условии, что ***функция риска*** определяется выражением (5), становится эквивалентной задаче ***минимизации риска*** (6).

2. СТРУКТУРА ЗАДАЧ УПРАВЛЕНИЯ РИСКАМИ СЛОЖНЫХ СИСТЕМ

Постановка задачи управления рисками сложных систем, приведённая в предыдущем параграфе, может быть без потери общности декомпозирована на несколько более специфических, последовательно возникающих задач.

Вначале для определения компонентов рассматриваемой системы и её операционного окружения необходимо решить *задачу идентификации*. Затем для идентифицированных объектов нужно определить эталонную модель их поведения, т. е. решить *задачу моделирования*. Далее необходимо сравнивать поведение объектов с эталонной моделью с целью *выявления аномалий* в нём. Наконец, требуется *спрогнозировать* развитие ситуации с учётом выявленных на предыдущем шаге аномалий.

На основе результатов решения этих четырёх задач можно искать общее решение задачи минимизации риска. Отметим, что эти задачи не зависят от сущности и структуры рассматриваемой системы. В то же время, для построения технологии их решения необходимо учесть специфику области применения и перейти к прикладным и технологическим задачам управления рисками. Классов таких задач тоже можно выделить четыре (рис. 1).

Соотнесём фундаментальные и прикладные задачи управления рисками сложных систем (см. рис. 1). Решение фундаментальной задачи идентификации объектов предполагает определение наиболее существенных их параметров. Совокупность (например, декартово произведение) областей значений всех принимаемых во внимание параметров всех допустимых в модели объектов образует пространство состояний системы и её окружения. Построение такого пространства представляет собой решение прикладной задачи параметризации.

Различные объекты системы и её окружения могут демонстрировать отличающееся поведение. Поэтому, чтобы начать решать фундаментальную задачу моделирования последнего, необходимо вначале решить прикладную задачу классификации объектов управляемой системы.

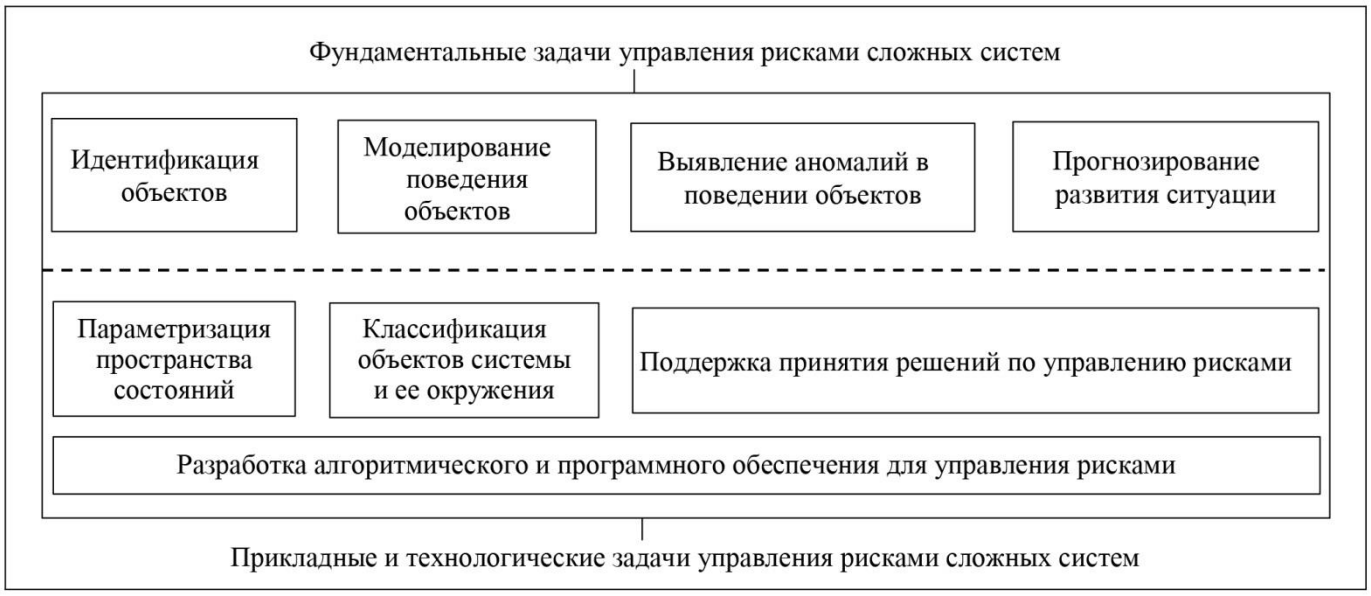

Рис. 1. Иерархическая структура задач управления рисками сложных систем

Фундаментальные задачи выявления аномалий в поведении и прогнозировании развития ситуации связаны на прикладном уровне с алгоритмами классификации, моделирования поведения, выявления аномалий и прогнозирования, которые при управлении рисками применяются совместно. В то же время разработка алгоритмов принятия решений является отдельной прикладной задачей, зависящей от предметной области, но не зависящей, как правило, от реализации других алгоритмов.

Рассматриваемые в настоящей работе алгоритмы и методы естественных вычислений призваны решать весь спектр вышеперечисленных прикладных задач классификации, моделирования поведения, выявления аномалий и прогнозирования. С точки зрения авторов, хороший метод должен быть примен*и*м для решения любой из этих задач, в том числе позволять разрабатывать алгоритмы для реализации в прикладном программном обеспечении. Все дальнейшие рассуждения мы будем строить исходя из этой посылки.

3. КЛАССИФИКАЦИЯ ЕСТЕСТВЕННЫХ ВЫЧИСЛЕНИЙ

При классификации естественных вычислений исследователи часто прибегают к основанию *биологического процесса* (биологическая эволюция, работа мозга, работа органов чувств и т. д.), на котором базируется алгоритм или модель вычислений. Это основание удобно тем, что хорошо разделяет классы алгоритмов. Его минус заключается в невозможности понять, какие задачи разрешимы с их помощью.

Другим подходом к классификации моделей и методов естественных вычислений является использование в качестве основания *принципа построения алгоритма*. Например, в обзоре [4] все алгоритмы естественных вычислений разделены на эволюционные, роевые и экологические. Отметим, что эти классы пересекаются: например, искусственные иммунные сети имеют характерные черты, присущие как роевым алгоритмам, так и эволюционным.

При рассмотрении моделей и алгоритмов естественных вычислений с точки зрения применимости к задаче управления рисками сложных систем (6) представляется целесообразным использовать для их классификации основания, отвечающие специфике этой задачи. С точки зрения авторов, последняя заключается в том, что разнообразие реакций таких систем на внешние воздействия не описывается конечным множеством. В условиях неопределённости также можно говорить о том, что множество внешних воздействий Θ идентифицировано нами не полностью. Стандартные методы оптимизации будут работать только в идентифицированной части этого множества. В то же время, весьма важная задача заключается в минимизации риска при возникновении новых, ранее не прогнозировавшихся внешних воздействий.

Технологическая задача разработки алгоритмического и программного обеспечения, будучи решаемой в рамках единой парадигмы, требует от применяемого подхода *универсальности*. Иными словами, выбранный подход должен позволять решать все прикладные задачи управления рисками сложных систем. Поэтому первым (и самым важным) основанием классификации природных вычислений будем считать универсальность их возможного применения. Поскольку речь идёт о вычислительных моделях, то под универсальностью понимается прежде всего способность реализовать произвольный алгоритм (*универсальный вычислитель*) или, по крайней мере, вычислить с заданной точностью произвольную функцию (*универсальный аппроксиматор*).

Формулировка задачи управления рисками сложных систем предполагает, что применяемый метод или алгоритм должен иметь адекватный задаче *формализм*. Кроме того, поскольку рассматриваемые системы чаще всего функционируют в нестационарном окружении, вид функций может (и будет) со временем меняться. Отсюда вытекает ещё одно требование к применяемому алгоритму или методу – *обучаемость*.

Таким образом, для решения поставленной задачи целесообразно использовать следующие основания классификации:

– универсальность применения (универсальный вычислитель, универсальный аппроксиматор),

– тип вычислений (формальная грамматика, математическая модель, семейство алгоритмов, элементная база и др.),

– формализм (лежит в основе формальная модель или нет),

– обучаемость (возможность реализации алгоритмов машинного обучения, возможность реализации адаптивных алгоритмов, нет обучаемости),

– техническая реализация (аппаратная платформа, программная платформа).

4. ОБЗОР ЕСТЕСТВЕННЫХ ВЫЧИСЛЕНИЙ

Модели и методы естественных вычислений включают в себя эвристические алгоритмы (или их семейства, объединённые общей идеей), некоторые формальные грамматики, элементные базы вычислителей, а также ряд математических моде-

лей. К первым относятся искусственные иммунные системы, роевой интеллект, аморфные и эволюционные вычисления. Формальными грамматиками описываются мембранные вычисления [5] и системы Линденмайера [6] (P- и L-системы соответственно). Относящиеся к естественным элементные базы вычислителей основаны на молекулах ДНК [7], амёбах Physarum [8] и некоторых видах бактерий [9, 10]. К естественным вычислениям на основе математических моделей относят клеточные автоматы [11], искусственные нейронные сети [12] и вычисления, основанные на динамических системах и хаосе [13]. Кратко опишем некоторые из них.

4.1. Формальные грамматики

Формальная **P-система** определяется как кортеж

$\Pi = (O, C, \mu, w_1, \ldots, w_m, R_1, \ldots, R_m, i_o)$,

где

O – непустой конечный алфавит *объектов*;

$C \subset O$ – множество *катализаторов*;

μ – мембранная структура, состоящая из пронумерованных от 1 до m мембран, задающих *регионы* P-системы;

$w_1, \ldots, w_m$ – строки над алфавитом O, описывающие *мультимножества объектов*, находящихся в регионах мембранной структуры с номерами 1, …, m соответственно;

$R_1, \ldots, R_m$ – конечный *набор правил эволюции* регионов мембранной структуры с номерами 1, …, m соответственно;

$i_o \in \{0, \cdots, m\}$ – номер региона, в который будет помещён результат вычислений (если он равен нулю, результат будет отправлен во внешнюю среду).

Правила эволюции имеют вид $u \to v$ или $u \to v\delta$, где $u \in O^+$, $v \in (O \times Tar)^*$, $Tar = \{here, in, out\}$. Через O^+ здесь обозначено множество всех возможных строк над алфавитом O, за исключением пустой строки λ, а $O^* = O^+ + \{\lambda\}$. Правила с произвольным воздействием u называют *кооперативными*, правила с $u \in O - C$ – *некооперативными*. Некооперативные правила с катализатором, т. е. вида $cu \to cv$ или $cu \to cv\delta$, где $u \in O - C$, $c \in C$ $v \in \left((O - C) \times Tar\right)^*$, называют *каталитическими*. Катализаторы в таких правилах «помогают» эволюционировать другим объектам P-системы, но сами не изменяются и никогда не перемещаются между её регионами.

Мембранная структура и мультимножества объектов, расположенных в ограниченных мембранами регионах, задают конфигурацию P-системы. Начальная конфигурация задана мембранной структурой и содержащимися в регионах объектами $(\mu, w_1, \ldots, w_m)$. В ходе эволюции системы в результате применения правил могут изменяться как мультимножества объектов, так и мембранная структура. Пример начальной конфигурации P-системы приведён на рис. 2.

Отметим, что правила эволюции на каждом её шаге применяются во всех регионах системы одновременно. Подробное описание мембранных вычислителей можно найти в работе [5].

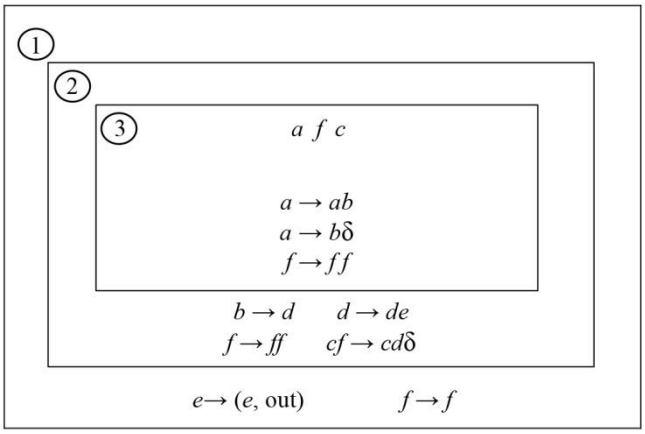

Рис. 2. Начальная конфигурация P-системы с тремя мембранами, алфавитом $O = \{a, b, c, d, e, f, \delta\}$ и множеством катализаторов $C = \{c\}$

Базовая **L-система** состоит из алфавита V, начальной строки – *аксиомы* $\omega \in V^+$, где V – множество всех строк ненулевой длины над V, и набора порождающих правил P вида p: $a \to x$, $a \in V$, $x \in V^+$. Для каждого символа $a \in V$, не входящего в левую часть правил P, предполагается правило $a \to a$. Такие символы называются *константами* или *терминальными символами*.

Правила из множества P применяются итеративно, начиная с аксиомы (начального состояния). На каждой итерации все символы, входящие в левую часть правил P, заменяются на соответствующие правые части. Например, система $G = (V = \{a, b\}, \omega = a, P = \{a \to b, b \to ba\})$ порождает строки $a \to b \to ba \to ba\,b \to ba\,b\,ba \to ba\,b\,ba\,ba\,b \to \ldots$

Базовые L-системы (так называемые 0L-системы) не используют специальных символов. Более сложные символьные L-системы (SL-системы) распознают специальные подстроки – зарезервированные символы-операторы. Подробное описание этих символов, а также L-систем в целом с рядом содержательных примеров можно найти в работе [6].

4.2. Элементные базы

ДНК-вычислители кодируют информацию в нуклеотидных последовательностях. Молекулы ДНК имеют в своём составе четыре азотистых основания – цитозин (C), гуанин (G), аденин (A) и тимин (T). Соответственно, в модели ДНК-вычислителя нуклеотидные последовательности представлены в виде строк над конечным алфавитом $\{A, T, G, C\}$. Производимые вычислителем с помощью различных ферментов операции над молекулами ДНК полностью описываются правилами расширенных H-систем (или сплайсинг-систем, см. книгу [14], *гл. 7*).

Пусть

$$S = (V, \Sigma, A, R),$$

где V – конечный алфавит, $\Sigma \in V$ – терминальный алфавит, $A \subset V^*$ – множество аксиом, а $A \subset V^* \times V^* \times$
$\times V^* \times V^*$ – правила сплайсинга.

Каждое правило $r = (u_1, u_2, u_3, u_4) \in R$ записывается в виде

$$r = u_1 \# u_2 \$ u_3 \# u_4;\ u_1, u_2, u_3, u_4 \in V^*;\ \#, \$ \notin V.$$

Интерпретируется эта запись так: разрезать строку между подряд идущими вхождениями подстрок u_1 и u_2, u_3 и u_4, а затем склеить фрагменты слева от первого разреза и справа от второго таким образом, чтобы u_1 и u_4 оказались рядом. Например, для некоторых $x_1, x_2, y_1, y_2 \in V^*$ и строк $x = x_1 u_1\ u_2 x_2$, $y = y_1 u_3 u_4 y_2$ правило r, применённое к паре (x, y), даст результирующую строку $z = x_1 u_1 u_4 y_2$.

Более подробно формальная модель ДНК-компьютера описана в работе [15]. Примером реализации такого вычислителя в живом организме являются бактериальные вычисления [9, 10].

4.3. Математические модели

Клеточный автомат задаётся кортежем $(\mathcal{L}, \mathcal{S}, \mathcal{N}, f)$, где

$\mathcal{L} \subseteq \mathbb{Z}^D$ – D-мерное клеточное пространство – множество (возможно, бесконечное) ячеек в $\mathcal{L}$, образующее регулярную решётку;

$\mathcal{S}$ – конечное множество состояний клеток;

$\mathcal{N} = (\vec{v_1}, \ldots, \vec{v_N})$ – вектор окружения из N элементов множества $\mathcal{L}$, связывающий клетку с соседними, причём соседями клетки, расположенной в $\vec{v} \in \mathcal{L}$, будут клетки в расположениях $(\vec{v} + \vec{v}_i) \in \mathcal{L} \forall \in \ \in \{1, \ldots, N\}$;

$f: S^N \to S$ – локальное правило перехода, задающее состояние клетки в следующий момент времени, $f(a_1, \ldots, a_N)$, где $a_1, \ldots, a_N$ – состояния соседних с ней N клеток.

Конфигурацией клеточного автомата называют глобальное состояние $C: \mathcal{L} \to \mathcal{S}$. Зададим множество всех конфигураций $\mathcal{S}^{\mathcal{L}}$. Тогда отображение $G: \mathcal{S}^{\mathcal{L}} \to \mathcal{S}^{\mathcal{L}}$ называется глобальной функцией перехода и клеточный автомат можно задать набором $\left(\mathcal{L},\ \mathcal{S},\ \mathcal{S}^{\mathcal{L}}, G\right)$.

Вычисления с помощью клеточного автомата происходят путём последовательного вычисления функции перехода. Как правило, клеточный автомат полагают синхронным (т. е. все клетки меняют своё состояние одновременно) и гомогенным (все клетки использую единое правило перехода). Тогда, если конфигурация $y = \left(y_{\vec{v}}\right)_{\vec{v} \in \mathcal{L}}$ следует за конфигурацией $x = (x_{\vec{v}})_{\vec{v} \in \mathcal{L}}$ (т. е. $y = G(x)$), то y – результат вычисления следующего выражения для каждого $\vec{v} \in \mathcal{L}$:

$$y_{\vec{v}} = G(x)_{\vec{v}} = G(x_{\vec{v}}) = f\left(x_{\vec{v}+\vec{v_1}}, \ldots, x_{\vec{v}+\vec{v_N}}\right).$$

Подробно о модели, типах и свойствах клеточных автоматов можно прочитать в книге [16].

Формальная модель **искусственного нейрона** записывается так:

$$s = \mathbf{w} \cdot \mathbf{x} + b,\ z = g(s),$$

где $\mathbf{x} \in \mathbb{Z}_{\{0,1\}}^N$ – вектор *входных данных*; $\mathbf{w}$ – вектор *весов*; b – смещение; $\mathrm{g}(\cdot)$ – *функция активации*; z – *выход*. При формировании искусственной нейронной сети отдельные нейроны s объединяются в слой

$$\mathbf{s} = \mathbf{W}\mathbf{x} + \mathbf{b},\ \mathbf{z} = g(\mathbf{s}),$$

где $\mathbf{W}$ – матрица весов. В нейронной сети может быть несколько связанных между собой слоёв.

Обучение искусственной нейронной сети осуществляется путём корректировки весов, как правило, на основе минимизации некоторого функционала потерь $L(\mathbf{w}) \to \min_{\mathbf{w}}$.

Эволюция архитектур искусственных нейронных сетей, режимов обучения и практического применения подробно обсуждается в обзоре [17].

Модель вычислений, основанных на динамических системах и хаосе, базируется на понятии хаотического элемента – *хаотического чипа* или *хаотического процессора* [18]. Пусть состояние некоторого такого элемента определяется значением переменной x. Логический вентиль, реализуемый этим элементом, построен так:

- входные сигналы:

$x \to x_0 + X_1 + X_2$ – для бинарных логических операций (NAND, NOR, XOR, AND, OR, XNOR);

$x \to x_0 + X$ – для унарных операций (NOT),

где x_0 – начальное состояние системы, а

$$X = \begin{cases} 0 \text{ при } I = 0, \\ V_{in} > 0 \text{ при } I = 1, \end{cases}$$

где V_{in} – положительная константа.

- динамическое обновление, т. е. $x \to f(x)$, где $f(x)$ – нелинейная функция.
- пороговый механизм для получения выходного сигнала Z:

$$Z = \begin{cases} 0, & \text{если } f(x) \le E, \\ f(x) - E, & \text{если } f(x) > E, \end{cases}$$

где E – пороговое значение.

Выходное значение интерпретируется как логический нуль, если $Z = 0$, и как логическая единица, если $Z > 0$.

Особенностью хаотических элементов является возможность менять тип реализуемого ими логического вентиля с помощью управления значениями (x_0, E). Классическим примером служит хаотический элемент с $f(x) = 4x(1 - x)$ и $V_{in}=1/4$. Такой элемент способен реализовывать одну из логических операций AND, OR, XOR, NAND или NOT в зависимости от значений (x_0, E) (табл. 1).

Таблица 1

Значения x_0 и E, соответствующие различным логическим операциям хаотического элемента с $f(x) = 4x(1 - x)$ и $V_{in}=1/4$

Логическая операция	AND	OR	XOR	NAND	NOT
x_0	0	1/8	1/4	3/8	1/2
E	3/4	11/16	3/4	11/16	3/4

Подробно парадигма хаотических вычислений обсуждается в работе [18].

Остальные подходы либо являются уникальными в классе природных вычислений, либо объединяют в себе столь разнородные элементы, что не могут быть чётко отнесены ни к одной из описанных групп.

4.4. Универсальность применения естественных вычислений

Рассмотрим вопрос об универсальности применения различных видов естественных вычислений.

Клеточные автоматы и искусственные нейронные сети представляют собой универсальные вычислители. Описанию Тьюринг-полного клеточного автомата посвящена классическая работа Элви Смита [19]. Теорема о Тьюринг-полноте полностью связанных рекуррентных искусственных нейронных сетей с сигмоидальными функциями активации была доказана в статье [20].

Парадигма вычислений, основанных на нелинейных динамических системах (хаосе), появилась как возможный ответ на ограничения транзисторов как традиционной элементной базы. Логические элементы, построенные на основе транзисторов, не могут быть изменены после аппаратной реализации. Программирование же осуществляется путём переключения между несколькими различными одноцелевыми элементами. А вот хаотический элемент способен преобразовываться в различные логические элементы с помощью механизма морфинга на основе порогов. В работе [21] описаны реализации логических вентилей AND, OR, NOT, XOR, а также обоснована универсальность хаотических чипов.

Вычисления на основе столкновений [22], также называемые в литературе «баллистическими вычислениями», «вычислениями свободного пространства» и «бильярдными вычислениями», используют для реализации логических схем однородную неструктурированную среду с перемещающимися мобильными локализациями. Таковыми могут быть планеры в клеточных автоматах, солитоны в оптической системе, волновой фрагмент в возбудимой химической системе. Логическая истина соответствует наличию локализации, логическая ложь – отсутствию локализации. При столкновении двух или более перемещающихся локализаций они изменяют свои векторы скорости и/или состояния. Постколлизионные траектории и/или состояния локализаций представляют собой результаты логических операций, реализуемых столкновением. Для модели бильярдных вычислений на основе двумерного блочного клеточного автомата доказана эквивалентность обратимой машине Тьюринга [23].

Под термином «вычисления «реакция – диффузия» может пониматься как вычислительная модель на основе соответствующего семейства дифференциальных уравнений, так и химический компьютер, где данные представляются в виде концентраций различных химических элементов, а обработка выполняется с помощью химических реакций – например, реакции Белоусова – Жаботинского [24, 25]. Уравнения же реакции-диффузии часто используются для моделирования других вычислительных сред, в частности хаоса и вычислений на основе столкновений. В работе [26] показана возможность универсальных вычислений в химических компьютерах в условиях ограниченных ресурсов. В этой же работе аналогичный результат был получен для Physarum – вычислений, которые обсуждаются далее.

Перейдём к рассмотрению формальных грамматик. Мембранные вычисления – они же P-

системы [5] – используют понятие мембраны, основанное на простейшей аналогии с биологическими клетками. Особенность таких систем заключается в том, что в процессе их функционирования происходит эволюция не только объектов вычислений (наборов символьных мультимножеств), но и самих мембранных структур. Поскольку каждая мембранная структура – клетка рассматривается как отдельный вычислительный элемент, а также благодаря применению принципа максимального параллелизма, такая грамматика позволяет записывать P-системы, решающие NP-полные задачи за полиномиальное время (за счёт экспоненциального роста числа мембран и, соответственно, параллелизма). Для некоторых P-систем доказана вычислительная полнота. В частности, в работе [27] это было сделано для коммуникативных (действия над объектом производятся только при его переходе через мембрану) мембранных систем. В статье [28] аналогичное утверждение доказано для нескольких классов P-систем с катализаторами (символы в мультимножестве, вызывающие применение того или иного правила, но не «потребляемые» им). В 2009 г. для расширения P-систем под названием «The mutual mobile membrane systems» (возможный перевод – «взаимно мобильные мембранные системы») была показана Тьюринг-полнота для случая с тремя мембранами [29].

Системы Линденмайера (L-системы) первоначально были предложены ботаником А. Линденмайером [30] как формальный язык для описания роста водорослей. В дальнейшем подход был расширен до полноценной формальной грамматики. В 1991 г. было показано, как можно использовать L-системы для разложения в ряд Тейлора некоторых элементарных функций [6], а вскоре создан компилятор и подсистема визуализации для платформы IBM PC [31]. Однако основной областью применения систем Линденмайера остаётся генерация фрактальных структур, вопросы вычислительной полноты таких систем в литературе не обсуждаются.

Отдельным направлением развития природных вычислений является поиск новых, альтернативных вычислительных элементных баз. Например, в 1994 г. было предложено использовать в качестве таковой молекулу ДНК [7]. Одно из преимуществ этого подхода заключается в том, что метод ДНК позволяет сгенерировать сразу все возможные варианты решений комбинаторных задач (например, задачу о гамильтоновом пути в ориентированном графе) с помощью известных биохимических реакций. Затем возможно быстро отфильтровать именно ту молекулу-нить, в которой закодирован нужный ответ. Однако при масштабировании предложенной Л. Адлеманом методики с ростом размерности задачи число необходимых для поиска решения молекул ДНК экспоненциально растёт, что накладывает физические ограничения на вычислительные возможности такого компьютера. Тем не менее, уже в 1999 г. были получены первые результаты, свидетельствующие о том, что ДНК-компьютеры могут быть универсальными вычислителями [32], а в 2017 г. предложен дизайн ДНК-компьютера, реализующего недетерминированную универсальную машину Тьюринга [33].

В 2010 г., используя идеи Адлемана, группа исследователей создала бактериальный компьютер на основе генетически модифицированной кишечной палочки [10]. В отличие от более ранних экспериментов [9], в этот раз исследователи поместили кодирующие задачу ДНК-последовательности в разрывы ДНК-цепочек генов, кодирующих флуоресцентные белки. При этом внедрённые фрагменты обрамлялись так называемыми hix-сайтами, что позволило использовать эффект сайт-специфической инверсии ДНК бактерий в присутствии специального белка – ДНК-инвертазы Hin. При инверсии цепочки нуклеотидов кодирующие флуоресцентные белки восстанавливаются – и «нашедшие» правильное решение бактерии светятся под микроскопом.

Другой подход к созданию нетрадиционных вычислителей заключается в использовании некоторых специальных свойств живых организмов. Наиболее известен эксперимент с амёбой *Physarum polycephalum* (L.), которая всегда стремится принять форму, минимизирующую воздействие на неё солнечного света. В работе [34] описано применение такого «амёбного компьютера» для приближённого решения NP-полной задачи коммивояжёра. Было показано, что амёба решает эту задачу за линейное время. Однако для расчёта схемы освещения амёбы используется рекуррентная нейронная сеть, динамика которой определяется весовой матрицей с n^4 элементами (n – число городов). Так что выигрыш во времени на больших размерностях представляется неочевидным. Вопрос об универсальности такого вычислителя также пока не исследовался.

Наконец, рассмотрим семейства эвристических и метаэвристических алгоритмов, обычно относимые к естественным вычислениям. Искусственные иммунные системы включают в себя несколько классов таких алгоритмов, созданных по аналогии с их природной реализацией в иммунных системах позвоночных животных. Несмотря на отсутствие общепринятой формализации (вариант, предложенный в статье [35], не получил широкого рас-

пространения), результаты работ [36, 37] позволяют предположить, что на основе искусственной иммунной системы можно построить универсальный аппроксиматор.

К алгоритмам роевого интеллекта обычно относят модели систем, состоящих из множества агентов, локально взаимодействующих с окружающей средой и между собой. Несмотря на то, что агенты подчиняются некоторым достаточно простым правилам поведения, система в целом демонстрирует сложное, «интеллектуальное» поведение. На практике такие алгоритмы применяются для решения различных задач оптимизации – см., например, обзор [38]. Вопрос об универсальности их применения (например, в роли универсального аппроксиматора) в литературе, по всей видимости, не поднимался.

Термин «аморфные вычисления» был введён группой исследователей из Массачусетского технологического института в 1996 г. для обозначения класса вычислительных устройств, состоящих из очень большого числа дешёвых, практически идентичных блоков обработки информации. Дешевизна является в данном случае существенным свойством, поскольку практическое применение таких устройств предполагало в том числе добавление их в качестве присадки при оптовом производстве «умных» конструкционных материалов. В работе [39] была впервые показана возможность проведения универсальных вычислений на аморфном компьютере, вычислительные блоки которого являются асинхронно работающими конечными вероятностными автоматами. В работе [40] рассмотрен ряд аморфных вычислительных систем, относящихся к числу универсальных вычислительных устройств.

Эволюционные вычисления, наряду с уже рассмотренными ранее моделями нечёткой логики и роевого интеллекта, относят к большому классу так называемых «мягких вычислений», основанных на приближённых моделях. Технически эволюционные вычисления – это семейство алгоритмов глобальной оптимизации, основанных на идее биологической эволюции. Семейство возможных решений-кандидатов образует «популяцию», которая постепенно улучшается с помощью селекции или случайных «мутаций». Процесс прекращается, когда решения достигают требуемого уровня точности.

Виды природных вычислений, их типы и сведения об универсальности их применения со ссылками описаны в табл. 2.

Таблица 2

Известные методы и алгоритмы естественных вычислений и универсальность их применения

Вид вычислений	Тип	Универсальность применения
Клеточные автоматы	Математическая модель, элементная база	Универсальный вычислитель [19]
Искусственные нейронные сети	– ” – ” –	– ” – ” – [20]
Вычисления, основанные на динамических системах и хаосе	– ” – ” –	– ” – ” – [21]
Вычисления, основанные на столкновениях	Вычислительная модель	– ” – ” – [23]
Вычисления «реакция – диффузия»	Математическая модель, элементная база	– ” – ” – [26]
Мембранные вычисления (P-системы)	Формальная грамматика	– ” – ” – [27–29]
Системы Линденмайера (L-системы)	– ” – ” –	Можно применять для символьных вычислений [6]
ДНК-вычисления	Элементная база	Универсальный вычислитель [32, 33]
Бактериальные вычисления	– ” – ” –	Возможно, являются универсальным вычислителем
Physarum-вычисления	– ” – ” –	Универсальный вычислитель [26]
Искусственные иммунные системы	Семейство алгоритмов	Возможно, являются универсальным аппроксиматором [36, 37]
Роевой интеллект	– ” – ” –	Возможно, является универсальным аппроксиматором
Аморфные вычисления	– ” – ” –	Универсальный вычислитель [39, 40]
Эволюционные вычисления	Семейство метаэвристических алгоритмов	Неприменимо

5. АНАЛИЗ ПРИМЕНИМОСТИ МОДЕЛЕЙ И МЕТОДОВ ЕСТЕСТВЕННЫХ ВЫЧИСЛЕНИЙ ДЛЯ РЕШЕНИЯ ЗАДАЧ УПРАВЛЕНИЯ РИСКАМИ СЛОЖНЫХ СИСТЕМ

Как уже отмечалось ранее, инструменты из числа природных вычислений, подходящие для решения задач управления рисками сложных систем, должны прежде всего быть универсальными, иметь в своей основе формальную математическую модель и, наконец, демонстрировать высокий уровень адаптивности. В смысле практического применения также важно, чтобы модель или алгоритм были реализованы программно (и результаты их работы можно было использовать при создании специализированного ПО) или аппаратно (для ускорения работы). Последнее требование особенно актуально при решении задач управления рисками в сфере информационной безопасности – например, при создании анализаторов трафика. В предыдущем параграфе мы отобрали те виды природных вычислений, для которых доказана их универсальность. Обсудим теперь соответствие отобранных моделей и методов остальным критериям отбора.

Для клеточных автоматов в 1969 г. был доказана теорема Кёртиса – Хедлунда – Линдона, утверждающая, что переходы между любыми двумя пространствами сдвига могут быть определены равномерно локальным правилом [41]. Класс адаптивных стохастических клеточных автоматов позволяет реализовывать адаптивные алгоритмы [42]. Технически формальная модель автомата реализуется на обычном компьютере – доступны свободно распространяемые программные модули на языках Python и Wolfram (пакет Mathematica). Таким образом, на основе этого типа вычислений вполне могут создаваться адаптивные системы управления.

Искусственные нейронные сети получили формальную модель на основе теории конечных автоматов ещё в 1956 г. [43]. Уже в следующем году была опубликована классическая работа с описанием алгоритма обучения искусственной нейронной сети на основе персептрона с двумя слоями, один из которых является скрытым и необучаемым [44]. В настоящий момент доступно множество запрограммированных архитектур нейронных сетей для решения различных классов задач (в основном распознавания и классификации), а также постоянно совершенствуются их аппаратные реализации – см, например, обзор [45].

В основе хаотических вычислений лежат модели различных нелинейных динамических систем [13], что даёт необходимый формализм. Инструменты для численного моделирования поведения таких систем есть, например, в пакете MATLAB. Также ведётся работа по созданию хаотического процессора – в частности, в статье [18] описана работа хаотических транзисторов. Этой же командой запатентована архитектура арифметико-логического устройства на основе таких транзисторов [46]. Способности к обучению такие системы сами по себе не демонстрируют.

Вычислительные системы, основанные на столкновениях, часто называют бильярдными компьютерами, а лежащую в их основе модель – «моделью формального бильярда». В работе [47] описаны все используемые в настоящее время варианты подвижных и стационарных локализаций – «шаров» и «столов» соответственно. Способностью к обучению такие системы не обладают. Построение моделей таких вычислителей возможно в системе MATLAB. Аппаратной реализации пока нет, в наиболее свежей известной работе по этому направлению [48] описана идея химического транзистора, использующего в качестве стационарных локализаций химические волновые фрагменты.

Первой формальной моделью системы «реакция – диффузия» для одномерного случая считают уравнение Колмогорова – Петровского – Пискунова [49]. Для описания сред, пригодных для использования в качестве вычислителя, требуются более сложные модели. В качестве примера можно привести химический компьютер, основанный на реакции Белоусова – Жаботинского [26, 50]. Обучаемостью такие системы не обладают. Наиболее современная программная реализация модели вычислений «реакция – диффузия» – библиотека ReaDDy 2 [51]. Также проводились эксперименты с программированием химического компьютера [24].

Основной компонент мембранного вычислителя – мембранная структура – был формально описан автором P-систем Г. Пауном [5]. Такая грамматика позволяет записывать адаптивные алгоритмы, но сам по себе мембранный вычислитель способностью обучаться не обладает. Программная реализация представлена в виде языка программирования P-Lingua [52], доступного в виде плагина к IDE Eclipse. Аппаратных реализаций нет.

Природные вычисления, основанные на альтернативных вычислительных базах (ДНК-вычисления, бактериальные компьютеры и

Physarum-вычисления), не имеют в основе уникальной вычислительной модели, а используют классические – машину Тьюринга, недетерминированные конечные автоматы или другие, в зависимости от предпочтений исследователя. Соответственно, «врождённых» способностей к обучению у таких вычислителей нет, хотя они позволяют реализовывать адаптивные и даже самообучающиеся алгоритмы. Все такие вычислители имеют аппаратную реализацию (см. ссылки в табл. 3).

Искусственные иммунные системы пока ещё рассматриваются большинством исследователей как набор эвристических алгоритмов. В 1998 г. была предпринята попытка положить в основу таких систем модель формального пептида [35], но этот формализм пока не является общепринятым. Возможно, именно это сдерживает развитие программных модулей для реализации решений на основе искусственных иммунных систем – несколько найденных авторами программных продуктов (см. табл. 3) в настоящий момент не развиваются. Иммунные сети обладают «врождённой» способностью к обучению и по потенциальным возможностям сравнимы с искусственными нейронными сетями – см., например, работу [53].

Алгоритмы роевого интеллекта не имеют в своей основе единой формальной модели элемента (частицы, агента и т. д.) роя, их объединяет принцип локального взаимодействия элементов между собой и с окружающей средой. Соответственно, такие алгоритмы не могут обучаться, хотя некоторые задачи прогнозирования и решаются с помощью роевых алгоритмов. Наиболее популярный из этого семейства метод роя частиц имеет несколько программных реализаций в виде подключаемых модулей для пакетов MATLAB [54] и SCILAB [55]. Остальные роевые алгоритмы распространены меньше и обычно реализуются исследователями для решения конкретных узких задач.

Аморфные вычисления также не подразумевают наличия формальной модели в своей основе – это скорее название класса вычислительных систем с очень высокой степенью параллелизма. Для облегчения программирования подобных систем – например, сенсорных сетей, – созданы специализированные языки программирования (см. табл. 3).

Таблица 3

Методы и алгоритмы природных вычислений и их характеристики: наличие в их основе формальной модели, способность этой модели к обучению, примеры технической реализации

Вид вычислений	Формализм	Обучаемость	Техническая реализация
Клеточные автоматы	Да [41]	Адаптивность [42]	Программная платформа (модули Python, Wolfram Mathematica)
Искусственные нейронные сети	– ” – [43]	Да [44]	Программные (PyTorch, TensorFlow и др.) и аппаратные (см. обзор [45]) реализации
Вычисления, основанные на динамических системах и хаосе	– ” – [13]	Нет	Программная (MATLAB) и аппаратная [18]
Вычисления, основанные на столкновениях	– ” – [47]	– ” –	Программная реализация в MATLAB
Вычисления «реакция – диффузия»	– ” – [49, 50]	– ” –	Программная – библиотека ReaDDy для Python и Java [51]
Мембранные вычисления (P-системы)	– ” – [5]	– ” –	Программная – язык P-lingua [52]
ДНК-вычисления	– ” – [15]	– ” –	Самособирающиеся ДНК-плитки [56]
Бактериальные вычисления	Нет	– ” –	Модифицированная E. Coli [9]
Physarum-вычисления	– ” –	– ” –	Amoeba-based computing system [34]
Искусственные иммунные системы	Да [35]	Да [53]	Программная – система Jisys [57], библиотеки iNet Framework [58] и libtissue [59]
Роевой интеллект	Нет	Нет	Программная, для PSO [54, 55]
Аморфные вычисления	– ” –	– ” –	Языки программирования GPL (Growing Point Language) [60] и Proto [61]

6. РЕШЕНИЕ ЗАДАЧ УПРАВЛЕНИЯ РИСКАМИ С ПОМОЩЬЮ ПРИРОДНЫХ ВЫЧИСЛЕНИЙ

Приведённый анализ позволил формально выявить несколько методов природных вычислений, определённо подходящих для решения задач управления рисками сложных систем. Это клеточные автоматы, искусственные нейронные сети и искусственные иммунные системы.

Для того чтобы сравнить распространённость этих подходов для решения фундаментальных и прикладных задач управления рисками сложных систем, проанализируем количество ссылок, выдаваемых Google Scholar в ответ на запрос, содержащий формулировку задачи (с синонимическими конструктами) и название метода (табл. 4 и 6). Отметим, что эффективность рассматриваемых методов в решении конкретных задач не оценивается и не сравнивается. Предложенный показатель скорее позволяет оценить «популярность» подхода в сообществе и косвенно характеризует глубину его разработки.

Таблица 4

Перечень запросов к Google Scholar для поиска публикаций, посвящённых решению фундаментальных задач управления рисками сложных систем

Класс задач	Текст запроса, примеры результатов поиска
Клеточные автоматы	
Идентификация	intext: «identification» allintitle: «by cellular automata» | «cellular automata for» – «identification of cellular automata» – «identification of optimal cellular automata» –«identification number» [62, 63]
Моделирование поведения	allintext: «behavior» «activity» «modeling» | «simulation» allintitle: «by cellular automata» | «cellular automata for» [64, 65]
Выявление аномалий	allintext: «anomaly detection» allintitle: «by cellular automata» | «cellular automata for" [66, 67]
Прогнозирование	allintext: «forecast» | «prediction» allintitle: «by cellular automata» | «cellular automata for» [68, 69]
Искусственные нейронные сети	
Идентификация	identification «artificial neural network» OR «neural network» OR «deep learning» – «identification of ANN» – «identification of neural network» – «identification number» – «neural network identification» [70, 71]
Моделирование поведения	behavior activity (simulation OR modelling) AND («artificial neural network» OR «neural network» OR «deep learning») – «neural network dynamics» – «neural network training» [72, 73]
Выявление аномалий	«anomaly detection» AND («artificial neural network» OR «neural network» OR «deep learning»)» [74, 75]
Прогнозирование	(forecast OR prediction) AND («artificial neural network» OR «neural network» OR «deep learning») [76, 77]
Искусственные иммунные сети	
Идентификация	allintext: «identification» «computing» «artificial immune system» allintitle: «AIS» | «immune system» | «artificial immune» – «identification of AIS» – «identification of artificial immune system» – «identification number» – «immune system identification» [78, 79]
Моделирование поведения	allintext: «behavior» «activity» «modeling» | «simulation» «artificial immune system» allintitle: «AIS» | «immune system» [80, 81]
Выявление аномалий	allintext: «anomaly detection» «artificial immune system» allintitle: «AIS» | «immune system» [82, 83]
Прогнозирование	allintext: «forecast» | «prediction» «artificial immune system» allintitle: «AIS» | «immune system» [84, 85]

Запросы структурированы следующим образом. В тексте подходящей под запрос работы должна содержаться формулировка задачи (например, «identification» – для задачи идентификации; «forecast OR prediction» – для задачи прогнозирования). Также в тексте работы должно быть указание на соответствующий метод или модель природных вычислений (например, «artificial immune system» – для искусственных иммунных систем), причём это может быть не полное наименование метода, а общепринятое сокращение или аббревиатура. Синонимы в запросе перечислены через оператор OR (логическое «ИЛИ»).

Отметим, что высокие значения результатов, полученных для искусственных нейронных сетей, на самом деле могли быть ещё выше – Google Scholar ограничивает время выполнения поискового запроса и по его истечении прекращает просматривать индекс публикаций. Таким образом, этот подход для решения задач управления рисками сложных систем применяется намного чаще других моделей искусственного интеллекта. Причём это замечание касается как фундаментальных, так и прикладных, и технологических задач (см. табл. 7).

Таблица 5

Применение моделей и методов искусственного интеллекта (число ссылок в Google Scholar) для решения фундаментальных задач управления рисками сложных систем

Задача	Клеточные автоматы	Искусственные нейронные сети	Искусственные иммунные сети
Идентификация	2 390	17 800	5 720
Моделирование поведния	1 630	20 700	2 550
Выявление аномалий	56	17 200	4 170
Прогнозирование	4 220	18 000	7 520

Таблица 6

Перечень запросов к Google Scholar для поиска публикаций, посвящённых решению прикладных и технологических задач управления рисками сложных систем

Класс задач	Текст запроса
Клеточные автоматы	
Классификация	allintext: «classification of» allintitle: «by cellular automata» | «cellular automata for» [86, 87]
Поддержка принятия решений	allintext: «decision support» allintitle: «by cellular automata» | «cellular automata for» [88, 89]
Разработка информационно-управляющих систем	allintext: «information system development» | «software development» «control» allintitle: [96, 97]
Искусственные нейронные сети	
Классификация	«classification of» AND («artificial neural network» OR «neural network» OR «deep learning») – «ANN classification» – «neural network classification» – «classification of neural» [92, 93]
Поддержка принятия решений	«decision support» AND («artificial neural network» OR «neural network» OR «deep learning») [94, 95]
Разработка информационно-управляющих систем	«control» AND («software development» OR «system development») AND («artificial neural network» OR «neural network» OR «deep learning») [96, 97]
Искусственные иммунные сети	
Классификация	allintext: «classification of» «artificial immune system» allintitle: «AIS» | «immune system» – «AIS classification» – «classification of artificial immune» – «classification of immune» – «immune system classification» [98, 99]
Поддержка принятия решений	allintext: «decision support» «artificial immune system» allintitle: «AIS» | «immune system» [100, 101]
Разработка информационно-управляющих систем	allintext: «information system development» | «software development» «control» «artificial immune system» allintitle: «AIS» | «immune system» [102, 103]

А вот клеточные автоматы уступают по частоте применения и нейронным, и иммунным сетям. Особенно редко удаётся встретить работы, где они применены для выявления аномалий и в качестве компонента информационной системы. При этом задачи выявления аномалий успешно решаются с их помощью, в том числе есть свежие публикации на эту тему (см., например, статью [67]).

Искусственные иммунные сети начали развиваться сравнительно недавно, при этом во многих областях показали себя довольно перспективным подходом. Этим можно объяснить их положение между клеточными автоматами и нейронными сетями по количеству работ, посвящённых решению одних и тех же классов задач. Однако применение их в информационных системах сдерживается отсутствием общепринятого формализма и, соответственно, программной реализации в виде софтверной библиотеки.

Таблица 7

Применение моделей и методов искусственного интеллекта для решения прикладных и технологических задач управления рисками сложных систем

Задача	Клеточные автоматы	Искусственные нейронные сети	Искусственные иммунные сети
Классификация	4 330	18 200	4 780
Поддержка принятия решений	1 050	17 800	2 210
Разработка информационно-управляющих систем	242	17 300	628

Ещё раз заметим, что число ссылок, выводимых поисковой системой по запросам из табл. 5 и 7 обосновывает не сравнительную «пригодность» модели для решения того или иного класса задач, а скорее описывает распределение предпочтений сообщества исследователей. Сравнительно малое число ссылок свидетельствует лишь о том, что метод или модель пользуется меньшей популярностью, чем другие. Причинами этого может быть как недостаточная исследованность модели, так и недостаток удобных программных инструментов для её использования.

ЗАКЛЮЧЕНИЕ

Применение аналитических методов для решения задач управления рисками сложных систем показывает предсказуемые результаты. Однако качество управления напрямую зависит от структуры управляемой системы. Для многоагентных гетерогенных открытых систем поиск аналитического решения такой задачи может занимать продолжительное время, при этом найденное решение чаще всего работает лишь в небольшой области пространства состояний.

Решения, основанные на естественных вычислениях, выглядят более перспективными из-за их высокой адаптивности, являющейся следствием сложности природных систем. Но не все такие подходы обладают нужным уровнем абстрактности, адаптивности и обучаемости.

В настоящей работе проанализированы известные модели и методы природных вычислений на предмет их применимости для решения как фундаментальных, так и прикладных/технологических задач управления рисками сложных систем. Выделены три модели, полностью соответствующие всем критериям – искусственные нейронные сети, клеточные автоматы, искусственные иммунные сети.

Искусственные нейронные сети применяются для решения задач управления рисками сложных систем уже много лет и доказали свою эффективность. Другие два подхода менее распространены, при этом нельзя сказать, что их потенциал ниже. Наиболее вероятными причинами, сдерживающими их развитие, является отсутствие общепринятого формализма для искусственных иммунных сетей и необучаемость клеточных автоматов.

Представляется целесообразным развить методы управления рисками сложных систем на основе клеточных автоматов и искусственных иммунных систем в будущих исследованиях.

ЛИТЕРАТУРА

1. *Handbook* of Natural Computing / Rozenberg, G., Bäck, T., Kok, J.N. (eds.). – Berlin: Springer Berlin Heidelberg, 2012. – 2105 p.
2. *Калашников А.О.* Управление информационными рисками организационных систем: общая постановка задачи // Информация и безопасность. – 2016. – Т. 19, № 1. – С. 36–45. [*Kalashnikov, A.O.* Upravlenie informatsionnymi riskami organizatsionnykh sistem: obshchaya postanovka zadachi // Informatsiya i bezopasnost'. – 2016. – Vol. 19, no. 1. – P. 36–45. (In Russian)]

3. *Кононов Д.А.* Исследование безопасности систем управления на основе анализа их системных параметров / Материалы 28-й Международной конференции «Проблемы управления безопасностью сложных систем» (ПУБСС'2020, Москва). М.: ИПУ РАН, 2020. С. 102–108. [*Kononov, D.A.* Issledovanie bezopasnosti sistem upravleniya na osnove analiza ikh sistemnykh parametrov / Materialy 28-i Mezhdunarodnoi konferentsii «Problemy upravleniya bezopasnost'yu slozhnykh sistem» (ICCSS'2020, Moscow). – M.: ICS RAS, 2020. – P. 102–108. (In Russian)]
4. *Nemade, M.N., Rane, M.D.* A Review on Bio-Inspired Computing Algorithms and Application // Proceedings of National Conference on Recent Trends in Computer Science and Information Technology (NCRTCSIT-2016). – Nagpur, India, 2016. – P. 12–19.
5. *Paun, G.* Introduction to Membrane Computing / in book Ciobanu, G., Paun, G., Perez-Jimenez, M.J. (eds.) Applications of Membrane Computing. – Berlin: Springer Berlin Heidelberg, 2006. – P. 1–42.
6. *Goel, N.S., Goodwin, M.D.* Symbolic computation using L-systems // Applied Mathematics and Computation. – 1991. – Vol. 42, no. 3. – P. 223–253.
7. *Adleman, L.M.* Molecular computation of solutions to combinatorial problems // Science : journal. – 1994. – Vol. 266, no. 5187. – P. 1021–1024.
8. *Adamatzky, A.* Physarum machines: computers from slime mould. – Vol. 74. – Singapore: World Scientific, 2010. – 280 p.
9. *Haynes, K., Broderick, M., Brown, A.* et al. Engineering bacteria to solve the Burnt Pancake Problem // Journal of biological engineering. – 2008. – Vol. 2. – P. 8–8-12.
10. *Poet, J.L., Campbell, A.M., Eckdahl T.T., Heyer L.J.* Bacterial computing // XRDS: Crossroads, The ACM Magazine for Students. – 2010. – Vol. 17, no. 1. – P. 10–15.
11. *Codd, E.F.* Cellular automata. – New-York: Academic Press, 2014. – 122 p.
12. *Гудфеллоу Я., Бенджио И., Курвилль А.* Глубокое обучение. – М.: ДМК-Пресс, 2018. – 652 с. [*Goodfellow, I., Bengio, Y., Courville, A.* Deep Learning. – Cambridge: The MIT Press, 2016.]
13. *Ditto, W.L., Miliotis, A., Murali, K., Sinha, S.* The Chaos Computing Paradigm / in book Reviews of nonlinear dynamics and complexity by Heinz Georg Schuster. – Weinheim, Germany: Wiley-VCH, 2010. – P. 1–35.
14. *Paun, G., Rozenberg, G., Salomaa, A.* DNA Computing. New Computing Paradigma. – Berlin: Springer, Heidelberg, 1998. – 420 p.
15. *Krasinski, T., Sakowski, S.* A theoretical model of the Shapiro finite state automaton built on DNA // Theoretical and Applied Informatics. – 2006. – Vol. 18. – P. 161–174.
16. *Hadeler, K.-P., Müller, J.* Cellular Automata: Analysis and Applications. – Cham, Switzerland: Springer, Cham, 2017. – 467 p.
17. *Макаренко А.В.* Глубокие нейронные сети: зарождение, становление, современное состояние // Проблемы управления. – 2020. – № 2. – С. 3–19. [*Makarenko, A.V.* Deep neural networks: origins, development, current status // Control Scienses. – 2020. – No 2. – P. 3–19. (In Russian)]
18. *Ditto, W.L., Miliotis, A., Murali, K., et al.* Chaogates: Morphing logic gates that exploit dynamical patterns // Chaos: An Interdisciplinary Journal of Nonlinear Science. – 2010. – Vol. 20, no. 3. – P. 037107–037107-8.
19. *Smith, A.* Simple Computation-Universal Cellular Spaces // Journal of the ACM. – 1971. – Vol. 18, no. 3. — P. 339–353.
20. *Siegelmann, H., Sontag, E.* Turing computability with neural nets // Appl. Math. Lett. – 1991. – Vol. 4, no. 6. – P. 77–80.
21. *Munakata, T., Sinha, S., Ditto, W.L.* Chaos computing: implementation of fundamental logical gates by chaotic elements // IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications. – 2002. – Vol. 49, no. 11. – P. 1629–1633.
22. *Adamatzky, A.* Collision-Based Computing. – London: Springer-Verlag, 2002. – 549 p.
23. *Durand-Lose, J.* About the Universality of the Billiard ball model // Multiple-Valued Logic. — 1998. – Vol. 6, no. 5. – P. 118–132.
24. *Adamatzky, A.* Programming Reaction-Diffusion Processors / in book Unconventional Programming Paradigms. – Berlin: Springer Berlin Heidelberg, 2005. – P. 33–46.
25. *Gorecki, J., Gizynski, K., Guzowski, J.,* et al. Chemical computing with reaction-diffusion processes // Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences. – 2015. – Vol. 373, no. 2046. – art. ID 20140219.
26. *Adamatzky, A., de Lacy Costello, B., Shirakawa, T.* Universal Computation with Limited Resources: Belousov – Zhabotinsky and Physarum Computers // International Journal of Bifurcation and Chaos. – 2008. – Vol. 18, no. 8. – P. 2373–2389.
27. *Alhazov, A., Margenstern, M., Rogozhin, V.,* et al. Communicative P systems with minimal cooperation // Membrane Computing. International Workshop WMC5. Revised Papers, LNCS 3365. – Milan, Italy, 2004. – Berlin: Springer, 2005. – P. 162–178.
28. *Freund, R., Kari, L., Oswald, M., Sosik, P.* Computationally universal P systems without priorities: two catalysts are sufficient // Theoretical Computer Science. – 2005. – Vol. 330, no. 2. — P. 251–266.
29. *Aman, B. Ciobanu, G.* Turing Completeness Using Three Mobile Membranes / in book Calude, C.S., Costa, J.F., Dershowitz, N., Freire, E., Rozenberg, G. (eds) Unconventional Computation. UC 2009. Lecture Notes in Computer Science, vol. 5715. – Berlin: Springer-Verlag Berlin Heidelberg, 2009. – P. 42–55.
30. *Lindenmayer, A.* Mathematical models for cellular interaction in development I-II // J. Theoret. Biology. – 1968. – No. 18. – P. 280–315.
31. *Goel, N., Rozehnal, I.* A High-Level Language for L-systems and Its Applications / in book Rozenberg, G., Salomaa, A. (eds.) Lindenmayer Systems: Impacts on Theoretical Computer Science, Computer Graphics, and Developmental Biology. – Berlin: Springer-Verlag, 1992. – P. 231–251.
32. *Winfree, E., Yang, X., Seeman, N.C.* Universal Computation via Self-assembly of DNA: Some Theory and Experiments // DNA Based Computers II: DIMACS Workshop June 10–12, 1996. DIMACS series in discrete mathematics and theoretical computer science. – No. 44, 1999. Proceeding of the Second DIMACS Workshop on DNA based computers. – Princeton, NJ, 1996. – P. 191–213.
33. *Currin, A., Korovin, K., Ababi, M.,* et al. Computing exponentially faster: implementing a non-deterministic universal Turing machine using DNA // Journal of the Royal Society, Interface. – 2017. – Vol. 14, no. 128. – art. ID 20160990.
34. *Zhu, L., Kim, S.-J., Hara, M., Aono, M.* Remarkable problem-solving ability of unicellular amoeboid organism and its mechanism // Royal Society Open Science. – 2018. – Vol. 5, no. 12. – P. 180396–180396-13.
35. *Tarakanov, A., Dasgupta, D.* A formal model of an artificial immune system // Biosystems. – 2000. – Vol. 55, no. 1-3. – P. 151–158.
36. *Gong, M., Jiao, L., Zhang, X.* A population-based artificial immune system for numerical optimization // Neurocomputing. – 2008. – Vol. 72, no. 1–3. – P. 149–161.
37. *Ülker E., Arslan, A.* Automatic knot adjustment using an artificial immune system for B-spline curve approximation // Information Sciences. – 2009. – Vol. 179, no. 10. – P. 1483–1494.

38. *Chakraborty, A., Kar, A.K.* Swarm Intelligence: A Review of Algorithms / in book Patnaik, S., Yang, X.S., Nakamatsu, K. (eds) Nature-Inspired Computing and Optimization. Modeling and Optimization in Science and Technologies, vol. 10. – Cham, Switzerland: Springer, Cham, 2017. – P. 475–494.
39. *Wiedermann, J.* Computability and Non-computability Issues in Amorphous Computing / in book Baeten, J.C.M., Ball, T., de Boer. F.S. (eds.) Theoretical Computer Science. TCS 2012. Lecture Notes in Computer Science, vol 7604. – Berlin: Springer, Heidelberg, 2012. – P. 1–9.
40. *Wiedermann, J., Petrů, L.* On the Universal Computing Power of Amorphous Computing Systems // Theory of Computing Systems. – 2009. – Vol. 45, no. 4. – P. 995–1010.
41. *Hedlund, G.A.* Endomorphisms and automorphisms of the shift dynamical system // Mathematical Systems Theory. – 1969. – Vol. 3, no. 4. – P. 320–375.
42. *Lee, Y.C., Qian, S. Jones, R.D.,* et al. Adaptive stochastic cellular automata: Theory // Physica D: Nonlinear Phenomena. – 1990. – Vol. 45, no. 1-3. – P. 159–180.
43. *Kleene, S.C.* Representation of Events in Nerve Nets and Finite Automata / in book Automata Studies. (AM-34), Vol. 34. – Princeton: Princeton University Press, 1956. – P. 3–42.
44. *Rosenblatt, F.* The Perceptron – a perceiving and recognizing automaton. Report 85–460–1, Cornell Aeronautical Laboratory. – 1957. – 29 P.
45. *Schuman, C.D, Potok, T.E., Patton, R.M.,* et al. A survey of neuromorphic computing and neural networks in hardware // arXiv preprint. – 2017. – arXiv:1705.06963.
46. *Ditto, W.L., Sinha, S., Murali, K.* Method and apparatus for a chaotic computing module using threshold reference signal implementations // US Patent No WO/2005/036353. – 2006.
47. *Adamatzky, A. Durand-Lose, J.* Collision-Based Computing / in book G. Rozenberg et al. (eds.), Handbook of Natural Computing. – Berlin: Springer-Verlag Berlin Heidelberg, 2012. – P. 1950–1978.
48. *Toth, R., Stone, C., de Lacy Costello, B.,* et al. Simple Collision-Based Chemical Logic Gates with Adaptive Computing // International Journal of Nanotechnology and Molecular Computation. – 2009. – Vol. 1, no. 3. – P. 1–16.
49. *Колмогоров А.Н., Петровский И.Г., Пискунов Н.С.* Исследование уравнения диффузии, соединённой с возрастанием вещества, и его применение к одной биологической пробле-ме // Бюллетень МГУ, Сер. А. Математика и Механика. – 1937. – № 1. – С. 1–26. [*Kolmogorov, A.N., Petrovskii, I.G., Piskunov, N.S.* Issledovanie uravneniya diffuzii, soedinennoi s vozrastaniem veshchestva, i ego primenenie k odnoi biologicheskoi probleme // Byulleten' MGU, Ser. A. Matematika i Mekhanika. – 1937. – No. 1. – P. 1–26. (In Russian)]
50. *Vanag, V.K.; Epstein, I.R.*. Stationary and Oscillatory Localized Patterns, and Subcritical Bifurcations // Physical Review Letters. – 2004. – Vol. 92, no. 12. – P. 128301–128301-4.
51. *Hoffmann, M, Fröhner, C, Noé, F.* ReaDDy 2: Fast and flexible software framework for interacting-particle reaction dynamics // PLOS Computational Biology. – 2019. – Vol. 15, no. 2. – P. e1006830.
52. *Garcia-Quismondo, M., Gutiérrez-Escudero, Rosa, Martínez-del-Amor, M.A.,* et al. P-Lingua 2.0: A software framework for cell-like P systems // International Journal of Computers, Communications & Control. – 2009. – Vol. 4, no. 3. – P. 234–243.
53. *Dasgupta, D.* Artificial neural networks and artificial immune systems: similarities and differences // Proceedings of IEEE 1997 International Conference on Computational Cybernetics and Simulation: Systems, Man, and Cybernetics. – Vol. 1. – Orlando, FL, 1997. – P. 873–878.
54. *Birge, B.* PSOt – a particle swarm optimization toolbox for use with Matlab // Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No. 03EX706). – Indianapolis, IN, USA, 2003. – P. 182–186.
55. *Qi, R., Hu, B., Cournède, P.* PSOTS: A Particle Swarm Optimization toolbox in Scilab // Proceedings of IEEE 2009 International Workshop on Open-source Software for Scientific Computation (OSSC). – Guiyang, China, 2009. – P. 107–114.
56. *Woods, D., Doty, D., Myhrvold, C.,* et al. Diverse and robust molecular algorithms using repro-grammable DNA self-assembly // Nature. – 2019. – Vol. 567, no. 7748. – P. 366–372.
57. *Hunt, J., Timmis, J., Cooke, D.,* et al. Jisys: Development of an Artificial Immune System for real world applications / in book Dasgupta D. (ed.) Artificial Immune Systems and their Applications. – Berlin: Springer-Verlag, 1998. – P. 157–186.
58. *Shen, X, Gao, X.Z., Bie, R, Jin, X.* Artificial Immune Networks: Models and Applications // Proceedings of the International Conference on Computational Intelligence and Security, ICCIAS 2006. – Guangzhou, China, 2006. – P. 394–397.
59. *Twycross, J. Aickelin, U.* Libtissue – Implementing Innate Immunity // Proceedings of IEEE 2006 International Conference on Evolutionary Computation (CEC). – Vancouver, Canada, 2006. – P. 499–506.
60. *Coore, D.* Botanical Computing: A Developmental Approach to Generating Interconnect Topologies on an Amorphous Computing / PhD thesis, MIT, 1999. – 295 p.
61. *Beal, J., Bachrach, J.* Infrastructure for engineered emergence on sensor/actuator networks // IEEE Intelligent Systems. – 2006. –Vol. 21, no. 2. – P. 10–19.
62. *Secco, J., Farina, M., Demarchi, D.,* et al. Memristor cellular automata for image pattern recognition and clinical applications // Proceedings of IEEE 2016 International Symposium on Circuits and Systems (ISCAS). – Montreal, QC, 2016. – P. 1378–1381.
63. *Miranda, G., Machicao, J., Bruno, O.* Exploring Spatiotemporal Dynamics of Cellular Automata for Pattern Recognition in Networks // Scientific Reports. – 2016. – Vol. 6. – P. 37329–37329-15.
64. *Partanen J.* An Urban Cellular Automata Model for Simulating Dynamic States on a Local Scale // Entropy. – 2017. – Vol. 19, no. 1. P. 12–12-19.
65. *Das, A.K., Chattaraj, U.* Heterogeneous Traffic Simulation for Urban Streets Using Cellular Automata // Arabian Journal for Science and Engineering. – 2019. – Vol. 44, no. 10. – P. 8557–8571.
66. *Sree, P.K., Babu, I.R.* Towards a Cellular Automata Based Network Intrusion Detection System with Power Level Metric in Wireless Adhoc Networks (IDFADNWCA) // Proceedings of the 2008 International Conference on Advanced Computer Theory and Engineering (ICACTE). – Phuket, Thailand, 2008. – P. 1071–1075.
67. *Nisha, V.M., Jeganathan, L.* A symmetry based anomaly detection in brain using cellular automata for computer aided diagnosis // Indonesian Journal of Electrical Engineering and Computer Science. – 2019. – Vol. 14, no. 1. – P. 471–477.
68. *Kabli, F., Hamou, R.M., Amine, A.* DNA data clustering by combination of 3D cellular automata and n-grams for structure molecule prediction // International Journal of Bioinformatics Research and Applications. – 2016. – Vol. 12, no. 4. – P. 299–311.
69. *Berberoğlu, S., Akın, A., Clarke, K.C.* Cellular automata modeling approaches to forecast urban growth for adana, Turkey: A comparative approach // Landscape and Urban Planning. – 2016. – Vol. 153. – P. 11–27.
70. *Shen, L., Shen, Y., Song, C.,* et al. A Novel Power System Anomaly Data Identification Method Based on Neural Network

and Affine Propagation // Proceedings of the International Conference on Artificial Intelligence and Security (ICAIS). – New-York City, NY, 2019. – P. 499–508.
71. *Ishitaki, T., Oda, T., Barolli, L.* A neural network based user identification for Tor networks: Data analysis using Friedman test // Proceedinngs of IEEE 2016 30th International Conference on Advanced Information Networking and Applications Workshops (WAINA). – Crans-Montana, Switzerland, 2016. – P. 7–13.
72. *Tobiyama, S., Yamaguchi, Y., Shimada, H.,* et al. Malware detection with deep neural network using process behavior // Proceedings of IEEE 40th Annual Computer Software and Applications Conference (COMPSAC). – Vol. 2. – Atlanta, GA, 2016. – P. 577–582.
73. *Manoj, K., Charul, B.* Hybrid tracking model and GSLM based neural network for crowd behavior recognition // Journal of Central South University. – 2017. – Vol. 24, no. 9. – P. 2071–2081.
74. *Raman, M.G., Somu, N., Mathur, A.P.* Anomaly Detection in Critical Infrastructure Using Probabilistic Neural Network // Proceedings of the International Conference on Applications and Techniques in Information Security 2019. – Tamil Nadu, India, 2019 – P. 129–141.
75. *Yuan, F., Cao, Y., Shang, Y.* Insider threat detection with deep neural network // Proceedings of the International Conference on Computational Science 2018. – Wuxi, China, 2018. – P. 43–54.
76. *Jiang, Y., Li, C.H., Yu, L.S., Bao, B.* On network security situation prediction based on RBF neural network // Proceedings of IEEE 2017 36th Chinese Control Conference (CCC). – Liaoning, China, 2017. – P. 4060–4063.
77. *Pang, Y., Xue, X., Wang, H.* Predicting vulnerable software components through deep neural network // Proceedings of the 2017 International Conference on Deep Learning Technologies. – Chengdu, China, 2017. – P. 6–10.
78. *Poteralski, A.* Hybrid artificial immune strategy in identification and optimization of mechanical systems // Journal of computational science. – 2017. – Vol. 23. – P. 216–225.
79. *Lima, F.P., Chavarette, F.R., Souza, S.S., Lopes, M.L.* Monitoring and fault identification in aeronautical structures using an wavelet-artificial immune system algorithm / in book Ekwaro-Osire, S., Gonçalves, A., Alemayehu, F. (eds.) Probabilistic Prognostics and Health Management of Energy Systems. – Cham, Switzerland: Springer, Cham, 2017. – P. 203–219.
80. *Purbasari, A., Supriana, I., Santoso, O.S., Mandala, R.* Designing Artificial Immune System Based on Clonal Selection: Using Agent-Based Modeling Approach // Proceedings of IEEE 2013 7th Asia Modelling Symposium. – Hong Kong, 2013. – P. 11–15.
81. *Liu, Y., Ding, Y., Hao, K., Chen, L.* An immune system-inspired information diffusion model // Proceeding of IEEE 2017 36th Chinese Control Conference (CCC). – Liaoning, China, 2017. – P. 11238–11243.
82. *Vasilyev, V., Shamsutdinov, R.* Distributed Intelligent System of Network Traffic Anomaly Detection Based on Artificial Immune System // Proceedings of 7th Scientific Conference on Information Technologies for Intelligent Decision Making Support (ITIDS 2019). – Ufa, Russia, 2019. – Atlantis Press, 2019. – P. 40–45.
83. *Jiang, Q., Chang, F.* A novel antibody population optimization based artificial immune system for rotating equipment anomaly detection // Journal of Mechanical Science and Technology. – 2020. – Vol. 34, no. 9. – P. 3565–3574.
84. *Wang, M., Ge, J., Zhang, D., Zhang, F.* An Improved Artificial Immune System Model for Link Prediction // Proceedings of the Pacific Rim International Conference on Artificial Intelligence (PRICAI). – Nanjing, China, 2018. – P. 1–9.
85. *Günay, M., Orman, Z., Ensari, T.,* et al. Diagnosis of Lung Cancer Using Artificial Immune System // Proceedings of IEEE 2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science (EBBT). – Istanbul, Turkey, 2019. – P. 1–4.
86. *Maji, P., Shaw, C., Ganguly, N.,* et al. Theory and application of cellular automata for pattern classification // Fundamenta Informaticae. – 2003. – Vol. 58, no. 3-4. – P. 321–354.
87. *Zhou, L., Yang, M.* A classifier build around cellular automata for distributed data mining // Proceedings of IEEE 2008 International Conference on Computer Science and Software Engineering. – Vol. 4. – Wuhan, China, 2008. – P. 312–315.
88. *Chang, H., Baek, S., Kim, H.,* et al. Development of distributed real-time decision support system for traffic management centers using microscopic CA model // Iranian Journal of Science & Technology, Transaction B, Engineering. – 2007. – Vol. 31, no B2. – P. 155–166.
89. *Benhacine, F.Z., Atmani, B., Benamina, M.,* et al. A Visual Decision Making Support System for the Diabetes Prevention // Alfaries, A., Mengash, H., Yasar, A., Shakshuki, E. (eds) Advances in Data Science, Cyber Security and IT Applications: First International Conference on Computing, ICC 2019, Proceedings, Part II. – Riyadh, Saudi Arabia, 2019. – P. 81–92.
90. *Liu, Y., He, J.* Developing a web-based cellular automata model for urban growth simulation // Proceedings of SPIE 7492, International Symposium on Spatial Analysis, Spatial-Temporal Data Modeling, and Data Mining. – Wuhan, China, 2009. – P. 74925C–74925C-8.
91. *Wainer, G.* Developing a software toolkit for urban traffic modeling // Software: Practice and Experience. – 2007. – Vol. 37, no. 13. – P. 1377–1404.
92. *Li, Y, Zhang, H, Shen, Q.* Spectral–spatial classification of hyperspectral imagery with 3D convolutional neural network // Remote Sensing. – 2017. – Vol. 9, no. 1. – P. 67– 88.
93. *Jia, F., Lei, Y., Lu, N., Xing, S.* Deep normalized convolutional neural network for imbalanced fault classification of machinery and its understanding via visualization // Mechanical Systems and Signal Processing. – 2018. – Vol. 110. – P. 249–367.
94. *Beşikçi, E.B., Arslan, O., Turan, O., Ölçer, A.I.* An artificial neural network based decision support system for energy efficient ship operations // Computers & Operations Research. – 2016. – Vol. 66. – P. 393–401.
95. *Boonpeng, S., Jeatrakul, P.* Decision support system for investing in stock market by using OAA-neural network // Proceedings of IEEE 2016 8th International Conference on Advanced Computational Intelligence (ICACI). – Chiang Mai, Thailand, 2016. – P. 1–6.
96. *Tang, G., Zeng, H.* Chemical Production Information Management System Based on Artificial Intelligence Neural Network Algorithm // Chemical Engineering Transactions. – 2018. – Vol. 66. – P. 967–972.
97. *Elechi, P.* Improved Ghost Worker Fraud Detection System Using Artificial Neural Network // Journal of Electrical Engineering, Electronics, Control and Computer Science. – 2019. – Vol. 5, no. 1. – P. 17–24.
98. *Magna, G., Casti, P., Jayaraman, S.V.,* et al. Identification of mammography anomalies for breast cancer detection by an ensemble of classification models based on artificial immune system // Knowledge-Based Systems. – 2016. – Vol. 101. – P. 60–70.
99. *Aldhaheri, S., Alghazzawi, D., Cheng, L.,* et al. DeepDCA: Novel Network-Based Detection of IoT Attacks Using Artificial Immune System // Applied Sciences. – 2020. – Vol. 10, no. 6. – P. 1909–1909-23.
100. *Mnif, S, Elkosantini, S, Darmoul, S, Said, L.B.* An immune multi-agent based decision support system for the control of public transportation systems // Proceedings of International

Conference on Practical Applications of Agents and Multi-Agent Systems. – Seville, Spain, 2016. – P. 187–198.
101. *Berquedich, M., Kamach, O., Masmoudi, M., Deshayes, L.* Agile decision support system for the management of tensions in emergency services using AIS techniques // Proceedings pf the 2017 IEEE International Colloquium on Logistics and Supply Chain Management (LOGISTIQUA). – Rabat, France, 2017. – P. 118–123.
102. *do Nascimento Alves, H., Machado, R.C., Bergê, I.G.* Design and development of a software for fault diagnosis in radial distribution networks // Proceedings of IEEE/IAS 9th International Conference on Industry Applications INDUSCON 2010. – São Paulo, Brazil, 2010. – P. 1–6.
103. *Chitra, M.E., Rajaram, M.* A Software Reliability Estimation Tool Using Artificial Immune Recognition System // Proceedings of the International MultiConference of Engineers and Computer Scientists IMECS. – 2008 (Vol. 1). – Hong Kong, 2008. – P. 967–975.

Статья представлена к публикации членом редколлегии В.В. Кульбой.

Поступила в редакцию 28.01.2021,
после доработки 31.03.2021.
Принята к публикации 6.04.2021.

Широкий Александр Александрович – канд. физ.-мат. наук,
✉ shiroky@ipu.ru,

Калашников Андрей Олегович – д-р техн. наук,
✉ aokalash@ipu.ru.

Институт проблем управления им. В.А. Трапезникова РАН, г. Москва.

NATURAL COMPUTING WITH APPLICATION TO RISK MANAGEMENT IN COMPLEX SYSTEMS

A.A. Shiroky and A.O. Kalashnikov

Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia

✉ shiroky@ipu.ru, ✉ aokalash@ipu.ru

Abstract. This paper surveys natural computing models and methods with application to risk management in complex systems. The equivalence of risk minimization and effective control problems is shown. The general risk management problem is stated for complex systems under uncertainty. The structure of fundamental and applied risk management problems is described. The well-known natural computing methods are briefly considered with application to risk management by the criteria of formalism, universality, and learning capability. The scientific community's preferences in natural computing models and methods for solving different classes of risk management problems are analyzed. Some promising approaches are outlined, which are currently underinvestigated according to the authors' opinion.

Keywords: risk management, effective control problem, natural computing.