УДК 004.032.3
DOI: 10.26102/2310-6018/2025.50.3.038

Вычислительная сложность в реальном времени

А.А. Зеленский, А.А. Грибков✉

Научно-производственный комплекс «Технологический центр», Москва, Зеленоград, Российская Федерация

Резюме. В статье излагаются результаты исследования, направленного на расширение теоретической базы в области вычислений в реальном времени. К числу рассматриваемых вопросов относятся: определение показателей вычислительной сложности в реальном времени, методология их количественной оценки, выявление способов достижения вычислимости алгоритмов в реальном времени, формализация подходов к оптимальной технической реализации вычислительных систем реального времени. Проведенные исследования опираются на существующие представления теории алгоритмов и теории вычислений, в том числе, вычислений в реальном времени. К числу значимых новых научных результатов проведенного исследования относятся: введение, наряду с известными показателями временной и пространственной вычислительной сложности, дополнительного показателя конфигурационной вычислительной сложности, необходимого для оценки вычислительной сложности в реальном времени; констатация возможности управления временной, пространственной и конфигурационной сложностью в рамках заданного функционала алгоритма исключительно за счет изменения числа потоков исполнения вычислений; теоретическое обоснование возможности снижения времени выполнения алгоритма конфигурирования с экспоненциального до полиномиального или даже линейного за счет конденсации исходного графа алгоритма с образованием из сильно связанных компонент совокупности функций-акторов и получения в результате ациклического ориентированного графа, топологическая сортировка которого выполнима за линейное время; определение подходов к оптимальной технической реализации алгоритма с заданной конфигурацией, в том числе, в виде интегральной схемы с разводкой, оптимизируемой на основе решения прямоугольной задачи Штейнера.

Ключевые слова: вычислительная сложность, реальное время, вычислимость, конфигурация, поисковый алгоритм, функции-акторы, портовость.

Благодарности: Исследование выполнено при поддержке Российского научного фонда по гранту № 24-19-00692, http://rscf.ru/project/24-19-00692/

Для цитирования: Зеленский А.А., Грибков А.А. Вычислительная сложность в реальном времени. *Моделирование, оптимизация и информационные технологии.* 2025;13(3). URL: https://moitvivt.ru/ru/journal/pdf?id=2017 DOI: 10.26102/2310-6018/2025.50.3.038

Real-time computational complexity

A.A. Zelenskii, A.A. Gribkov✉

Scientific and Production Complex "Technological Center", Moscow, Zelenograd, the Russian Federation

Abstract. The article presents the results of a study aimed at expanding the theoretical basis in the field of real-time computing. The issues considered include: defining indicators of computational complexity in real time, a methodology for their quantitative assessment, identifying ways to achieve the computability of algorithms in real time, and formalizing approaches to the optimal technical implementation of real-time computing systems. The research is based on existing concepts in algorithm theory and computation theory, including real-time computation. Significant new scientific results of the research include: the introduction, along with the known indicators of temporal and spatial

computational complexity, of an additional indicator of configuration computational complexity, necessary for assessing computational complexity in real time; the confirmation of the possibility of controlling temporal, spatial, and configuration complexity within the framework of a given algorithm functional solely by changing the number of computation execution threads; theoretical justification of the possibility of reducing the execution time of the configuration algorithm from exponential to polynomial or even linear by condensing the initial graph of the algorithm with the formation of strongly connected components of a set of actor functions and obtaining as a result an acyclic directed graph, whose topological sorting can be performed in linear time; determination of approaches to the optimal technical implementation of the algorithm with a given configuration, including in the form of an integrated circuit with wiring optimized based on the solution of Steiner's rectangular problem.

Keywords: computational complexity, real time, computability, configuration, search algorithm, actor functions, portability.

Acknowledgements: The research was supported by the Russian Science Foundation under grant No. 24-19-00692, http://rscf.ru/project/24-19-00692/

For citation: Zelenskii A.A., Gribkov A.A. Real-time computational complexity. *Modeling, Optimization and Information Technology*. 2025;13(3). (In Russ.). URL: https://moitvivt.ru/ru/journal/pdf?id=2017 DOI: 10.26102/2310-6018/2025.50.3.038

Введение

Ключевыми составляющими третьей (с начала 1960-х до конца 1990-х годов), четвертой (с середины 2010-х годов до настоящего времени) и грядущей пятой (предположительно в конце 2030-х – начале 2040-х) промышленных революций является автоматизация и роботизация – начиная с 3-й промышленной революции, информатизация и цифровизация – начиная с 4-й промышленной революции, интеллектуализация (на базе искусственных когнитивных систем) и кибернетизация (объединение технологических и биологических систем) – начиная с 4-й или 5-й промышленных революций. Все обозначенные тенденции соответствуют постепенному замещению человека в производстве: вначале освобождению от монотонного ручного труда по производству материальных благ, затем от неинтеллектуальных задач управления машинами, производящими материальные блага, и, наконец, от интеллектуального управления такими машинами [1].

Определяющим технологическим фактором в замещении человека машинами является повсеместное использование различных систем управления: технологическим оборудованием, технологическим процессами и производствами, транспортными средствами и др. Характер используемых систем управления определяется объектами управления, разнообразие которых неуклонно расширяется. При этом совершающиеся промышленные революции сопровождаются сменой приоритетных групп объектов управления. Эффективность современного производства в наибольшей степени зависит от управления технологическим оборудованием, причем для высокотехнологичного оборудования приоритетным является управление движением (рабочего органа) в реальном времени.

Вычислительные системы реального времени – системы, обеспечивающие выполнение необходимого (заданного) комплекса вычислений за ограниченный малый интервал времени. Такие вычислительные системы бывают двух основных видов: системы мягкого реального времени, требующие соблюдения временных ограничений «в-среднем», и системы жесткого реального времени, для которых нарушения временных ограничений недопустимы (фатальны с точки зрения достижения цели вычислений). Для решения задач управления движением в реальном времени требуются вычислительные системы жесткого реального времени. В этой связи необходимо

отметить, что практически все системы жесткого реального времени – это системы управления (наиболее часто – управления движением).

Технологическое развитие производства сопровождается постоянным повышением сложности объектов управления. Исследования, проведенные авторами, показали, что повышение сложности объекта управления требует уменьшения длительности цикла управления (примерно пропорционально сложности объекта управления) [2].

Практическое обеспечения реализуемости вычислений за цикл управления – задача, решение которой связано с существенными сложностями при управлении современным высокотехнологичным оборудованием. В среднесрочной перспективе сложность задачи ожидаемо продолжит увеличиваться. Для объективной оценки перспектив обеспечения реализуемости в условиях роста сложности объектов управления необходимо формирование теоретической базы в рамках теории сложности, включающей в себя соответствующий понятийный аппарат для характеристики вычислительной сложности, определение подходов к снижению вычислительной сложности до полиномиальной, а также методология оптимальной технической реализации вычислительных алгоритмов в реальном времени.

Показатели вычислительной сложности

Согласно принятому определению, функция f с натуральными аргументами и значениями называется вычислимой, если существует алгоритм, её вычисляющий[1]. Для вычислимости функций в реальном времени необходимо введение дополнительных ограничений. В частности, согласно Хисао Ямаде [3], для того, чтобы функция была вычислима в реальном времени, она должна задаваться исключительно операторами сложения, умножения, возведения в степень и суперпозиции (подстановки функций или отождествления переменных).

Вычислительная сложность алгоритма в рамках теории сложности характеризуется двумя показателями: временной и пространственной сложностью алгоритма.

Временная сложность алгоритма определяется как некоторая функция от количества шагов, за которые выполняемый последовательный алгоритм приводит к искомому результату. В пределах какого-то количества шагов вычислительные операции, соответствующие этому алгоритму, могут выполняться параллельно, что позволяет сократить общее числа шагов, приводящих к искомому результату, а, значит, уменьшить временную сложность алгоритма.

Пространственная сложность алгоритма обычно интерпретируется как функция от объема памяти, используемой алгоритмом. Эта память включает в себя память, необходимую для хранения входных данных и дополнительную память, используемую для выполнения алгоритма.

Определение сложности вычислений в реальном времени – специфическая задач, для решения которой указанных двух показателей недостаточно. Кроме того, эти показатели, а также используемые для их определения понятия, требуют уточнения и детализации.

Алгоритм вычислений в реальном времени целесообразно представлять в виде композиции функций, в совокупности формирующих функционал вычислительной системы, реализуемый за цикл управления. Конкретная структурная реализация

[1] Верещагин Н.К., Шень А. *Лекции по математической логике и теории алгоритмов. Часть 3. Вычислимые функции.* Москва: МЦНМО; 2012. 160 с.

композиции функций – это конфигурация композиции, определяющая распределение выполнения отдельных функций по времени и потокам исполнения.

Вычислительная сложность в реальном времени адекватно характеризуется тремя показателями: временной, пространственной и (вводимой авторами) конфигурационной сложностями.

Временная вычислительная сложность в реальном времени $C_t = C_t(\tau)$ является линейной функцией от длительности τ цикла управления (вычислений) – по мере роста длительности цикла прямо пропорционально повышается временная вычислительная сложность.

Пространственная вычислительная сложность в реальном времени $C_m = C_m(W)$ является линейной функцией от максимального объема памяти W, одновременно используемого функциями, формирующими композицию, – по мере роста максимального потребного объема памяти прямо пропорционально повышается пространственная вычислительная сложность.

Конфигурационная вычислительная сложность в реальном времени $C_R = C_R(S)$ является функцией от минимальной портовости, обеспечивающей полную функциональность конфигурации композиции функций. Портовость S – это минимальное число портов у акторов, соответствующих функциям, формирующим композицию, определяемое в рамках акторной модели представления конфигурации, соответствующее числу действующих связей между функциями в композиции.

Конфигурационная вычислительная сложность в реальном времени отражает сложность алгоритма, рассматриваемого как система функций-акторов. Как известно [4], методологические подходы к определению сложности систем могут быть обобщены в виде четырех типов концепций: логической (определяются меры некоторых свойств отношений, которые считаются упрощающими); теоретически-информационной (в рамках которой сложность отожествляется с энтропией); алгоритмической (сложность определяется длиной алгоритма, необходимого для определения исследуемого объекта); теоретико-множественной (сложность связывается с мощностью множества элементов, из которых состоит изучаемый объект). Проведенный анализ [5] позволил выделить в качестве базового, обеспечивающего наибольшую достоверность, методологический подход в рамках теоретико-множественной концепции, заключающийся в определение сложности системы как числа связей между параметрами формирующих ее элементов.

Применительно к рассматриваемой нами системе функций-акторов, формирующих алгоритм (в виде композиции), сложность системы определяется количеством связей между функциями-акторами, причем не всеми связями, а только значимыми (действующими, активными). Количество именно таких связей отражает показатель портовости. В результате функция $C_R(S)$, как и две другие функции вычислительной сложности в реальном времени, является линейной – по мере роста портовости прямо пропорционально увеличивается «системная сложность» алгоритма, показателем которой является конфигурационная вычислительная сложность.

Оценка показателей вычислительной сложности

Значения временной, пространственной или конфигурационной сложности в реальном времени могут задаваться в рамках формулирования оптимизационной задачи (или задачи поиска годного решения) в виде ограничения по времени, по использованию памяти или по числу портов. В этом случае заранее известны временная, пространственная или конфигурационная вычислительные сложности и все они (или часть из них) не требуют определения. Имеет место решение обратной задачи синтеза: определение вычислительного алгоритма (в виде конфигурации для композиции

функций), при котором все или отдельные показатели сложности вычислений в реальном времени не превышают допустимых значений.

В контексте вычислений в реальном времени временная сложность может быть отожествлена с длительностью цикла вычислений. В этом случае, в соответствии с теоремой о промежутках [6], по мере снижения временной сложности набор реализуемых функций сокращается. Для сохранения реализуемости вычислительного алгоритма (алгоритма управления) необходимо использовать параллельные вычисления. При этом происходит увеличение пространственной и конфигурационной сложности, поскольку параллельное выполнение функций требует большего объема памяти и большей портовости (большего количества действующих связей).

Для качественной оценки показателей вычислительной сложности в реальном времени рассмотрим условную, предельно упрощенную конфигурацию, в которой все функции, формирующие композицию, выполняются за постоянное время t, а для всех прочих параметров алгоритма будем использовать их средние значения.

Длительность цикла управления (вычислений) определится следующим образом:

$$\tau = T \cdot \mu / p = n \cdot t \cdot \mu / p, \quad (1)$$

где $T = n \cdot t$ – длительность отработки всех функций композиции в случае, если бы они выполнялись последовательно; n – количество функций-акторов, выполняемых внутри цикла; p – количество потоков исполнения (принимается постоянным для всех групп); u – количество групп функций-акторов; $\mu \le \frac{n}{up} / \left(\left\lfloor \frac{n}{up} \right\rfloor + \frac{1}{p} \sum_{j=p}^{1} \left(\frac{n}{up} \operatorname{Mod}_{i=p}^{j} i \right) \right)$ – показатель равномерности заполнения потоков исполнения функциями-акторами (в случае отсутствия задержек задействования функций-акторов для соблюдения установленного порядка их выполнения: $n / (n + u) \le \mu \le 1$).

Определение функции $\mu(n,u,p)$ основывается на описании заполнения потоков следующим образом: вначале функции-акторы, относящиеся к группе, поровну делятся по потокам; далее, если при делении образовался остаток, он делится между меньшим числом потоков.

Величины n и u в формуле (1) заданы исходя из функциональных свойств композиции и не могут быть изменены. Единственным варьируемым параметром является количество потоков исполнения p. Как можно видеть, по мере роста числа потоков длительность цикла управления (вычислений) сокращается, что неудивительно, поскольку именно с этой целью используется многопоточность.

Количество потоков исполнения, в свою очередь, определяется исходя из установленных ограничений на объем используемой памяти W. Если для оценочного расчета принять, что для каждой из функций-акторов требуется одинаковый объем памяти, равный w, то:

$$W = w \cdot p. \quad (2)$$

Портовость S конфигурации алгоритма (композиции функций) зависит от количества функций-акторов, групп, на которые они делятся, и количества потоков (принимаемого постоянным для всех групп) [7]:

$$S = s \cdot n = \lambda \cdot u \cdot p \cdot n, \quad (3)$$

где $s = \lambda \cdot u \cdot p$ – средняя портовость одной функции-актора; λ – среднее значение доли (из общего числа) функций-акторов, для которых относительно какой-либо одной функции-актора установлена последовательность выполнения.

Единственным варьируемым параметром при определении используемой памяти и портовости конфигурации алгоритма, как и в случае длительности цикла управления,

является количество потоков исполнения p. Рост числа потоков требует увеличения объема доступной памяти, что является ограничением для реализации многопоточности. Рост числа потоков также сопровождается увеличением портовости, обусловленным повышением доступности функций-акторов друг для друга (для любого актора в пределах каждого потока каждой группы возможны только один порт входа и один порт выхода).

Достижимость вычислимости алгоритмов в реальном времени

Задача определения оптимальной или годной конфигурации для заданного алгоритма вычислений, которая должна решаться за ограниченный интервал времени (в реальном времени) идентична проблеме рюкзака в реальном времени (real-time knapsack problem или temporal knapsack problem) [8]. Как известно, алгоритм решения этой проблемы не может быть представлен в полиноминальной форме, а значит, относится к задачам класса сложности NP (non-deterministic polynomial). Выполнение такого алгоритма может потребовать экспоненциального, в частности факториального времени, а вычислительная сложность может быть супер-полиномиальной (super-polynomial computational complexity), характеризующейся супер-полиномиальным временем с нотацией для наибольшего времени выполнения $\omega(n^c)$, где c – любая постоянная величина.

Конкретный анализ связей между функциями-акторами в конфигурации показывает, что сложность задачи определения оптимальной или годной конфигурации может быть снижена до полиномиальной, соответствующей вычислимости алгоритма. Если представить связи между функциями-акторами в виде графа, то, в процессе конфигурирования, вершины (функции-акторы) должны быть упорядочены исходя из заданных между ними отношений (задание формулируется в виде матрицы отношений, регламентирующей последовательность выполнения функций-акторов). Поскольку пути между каждой парой вершин (функций-акторов) существуют только в одном направлении (от функции-актора, выполняемой раньше по времени, к функции-актору, выполняемой позже) и обход путей (в процессе выполнения алгоритма) не завершается в начальной точке, следовательно, это ациклический ориентированный граф (DAG, от англ. directed acyclic graph).

Важным свойством любого такого графа является то, что он имеет хотя бы одно топологическое упорядочение, и существуют алгоритмы, например, алгоритм Кана (Kahn's algorithm [9]), его построения за линейное время. Топологическое упорядочение (сортировка) – это линейное упорядочение вершин, для которого выполняется следующее условие: для каждого направленного ребра uv вершина u предшествует вершине v в упорядочении. Нотация сложности алгоритма топологического упорядочения $O(|V|+|E|)$, где V – количество вершин, E – количество дуг (путей между вершинами).

Поиск топологической сортировки может реализовываться не только посредством топологического упорядочивания. Более универсальным методом (применимым не только для DAG) является обход в глубину (DFS, от англ. depth-first search) [10]. Наиболее известными алгоритмами DFS, характеризующимися, как и топологическая сортировка, временной сложностью алгоритма упорядочивания $O(|V|+|E|)$, являются: двухпроходный алгоритм Косарайю (Kosaraju's algorithm [11]), однопроходный алгоритм Тарьяна (Tarjan's algorithm [12]) и алгоритм Габова (Gabow's algorithm [13]).

Интерес также представляет алгоритм обхода графа в ширину (BFS, от англ. breadth-first search [14]), обеспечивающий нахождение «первого подходящего пути», а в

результате сортировки – первого годного варианта конфигурации. Алгоритм BFS выполняет передвижение по ребрам графа от одной вершине к другой от уровня к уровню: сначала проходит по всем ближайшим от начальной точки вершинам, потом спускается глубже. Нотация временной вычислительной сложности данного алгоритма, как и рассмотренных выше, составляет $O(|V|+|E|)$.

Важным свойством рассматриваемой конфигурации, формируемой как распределение выполнения функций-акторов по потокам, группам и времени, является представление каждой их функций-акторов как «черного ящика», внутренние связи в котором не рассматриваются. Между тем, функции-акторы, если раскрыть их в виде локальных графов в составе исходного графа вычислительного алгоритма, могут быть сильно связанными, что делает невозможной вычислимость всего алгоритма за полиноминальное время. Для того, чтобы избавиться от сильной связанности вершин локальных графов необходимо трансформировать исходный граф в граф, составленный из локальных компонент сильной связности, а не индивидуальных вершин. Задачу о сжатии каждой компоненты сильной связности в одну вершину называют конденсацией графа [15]. Результатом такой конденсации является формирование множества функций-акторов, инкапсулированных от цикла, структурно объединенных в конфигурацию, описываемую ациклическим ориентированным графом, не препятствующих полиномиальной (или даже линейной) вычислимости алгоритма.

Поиск областей сильной связности в ориентированном графе осуществляется посредством топологической сортировки с помощью алгоритма Косарайю. Если локальные компоненты исходного графа не формируют сильной связности (для образования функции-актора), возможно их дополнение до сильно связанных. Для решения невзвешенной версии задачи о дополнении графа до сильно связного используется приближенный алгоритм, полученный К. Эсмараном и Р. Терьяном [16], позволяющий находить решение за линейное время. Взвешенная версия задачи о дополнении графа до сильно связанного относится NP-сложным и требует для решения экспоненциального времени.

Оптимальная техническая реализация конфигурации алгоритма

Эффективность вычислительного алгоритма, структурно представленного в виде конфигурации, может быть количественно оценена на основе констатированных ранее трех показателей вычислительной сложности в реальном времени. В рамках определения конфигурационной вычислительной сложности можно определить количество портов ввода и вывода, необходимых для выполнения вычислительного алгоритма. Это количество портов является показателем сложности технической реализации алгоритма: она тем ниже, чем меньше портов необходимо для его реализации.

Практическая техническая реализация алгоритма в виде вычислительной системы (например, на базе интегральной микросхемы) требует определения физического распределения образующих ее элементов (соответствующих акторам). Оптимизация распределения элементов может осуществляться по различным критериям, наиболее применяемым из которых является суммарная длина каналов связи, подлежащая минимизации с целью экономии металла, повышения быстродействия, уменьшения диссипации энергии и снижения помех.

Оптимизация физического распределения элементов по критерию минимизации суммарной длины каналов связи может быть сведена к решению задачи Штейнера о минимальном дереве, формулируемой для случая эвклидовой метрики следующим образом [17]: «под задачей Штейнера на евклидовой плоскости понимается проблема нахождения на плоскости с евклидовой метрикой кратчайшего дерева, связывающего n

заданных точек (терминалов)... в отличие от известной задачи о кратчайшей связывающей сети на графе в задаче Штейнера допускается введение при необходимости новых вершин дерева, отличных от терминальных. Эти вершины называются точками Штейнера. Полученное в результате решение является деревом и называется минимальным деревом Штейнера».

Задача Штейнера относится к классу NP-полных, поэтому, в случае неравенства классов P и NP, не существует алгоритма, дающего точного ее решения за полиномиальное время. Оптимальное решение задачи за экспоненциальное время теоретически может быть получено при использовании комбинаторных алгоритмов Мелзака [18] и Кокейна [19]. На практике, однако, использование этих алгоритмов затруднительно. Например, алгоритм Кокейна уже для множества из восьми терминальных вершин проверяет 105 топологий [20]. Поэтому в настоящее время для решения задачи Штейнера о минимальном дереве используют приближенный эвристический алгоритм Л. Коу, Г. Марковского и Л. Бергмана [21], позволяющий найти дерево, не являющееся оптимальным, но с длиной связывающей сети, удовлетворяющей минимальным технологическим требованиям.

Обобщением (классической) задачи Штейнера является сетевая задача Штейнера, в которой следует минимизировать не суммарную длину каналов связи, а их общую стоимость. В сетевой задаче Штейнера величины весов ребер зависят от потока по ним. В этом случае разработанные для задачи Штейнера алгоритмы декомпозиции неприменимы и вместо них используют подходы, основанные на динамической декомпозиции [22].

Частным случаем постановки задачи Штейнера является прямоугольная проблема Штейнера. Эта проблема формулируется в рамках манхэттенской плоскости [23, с. 17] – метрического пространства, в котором расстояние между точками порождается нормой $||(x,y)|| = |x| + |y|$. При определении оптимальных разводок микросхем [24], соответствующих технической реализации вычислительных алгоритмов, решается обычно именно прямоугольная проблема Штейнера.

Заключение

На основе проведенного в статье анализа можно сделать следующие выводы:

1. Одной из ключевых составляющих технологических революций, начиная с третьей (с начала 1960-х годов) и до грядущей (в начале 2030-х годов) пятой, является все более широкое использование различных систем управления, в том числе систем реального времени.

2. Вычислительная сложность в реальном времени, в отличие от обычной вычислимости, складывается не из двух, а из трех показателей: временной, пространственной и конфигурационной вычислимости. Последний показатель является функцией от минимальной портовости, обеспечивающей полную функциональность алгоритма (конфигурации композиции функций).

3. Все три показателя вычислительной сложности в реальном времени зависят от количества потоков исполнения алгоритма. Количество потоков – единственный вариативный параметр, изменяя который можно обеспечить удовлетворение требований по длительности цикла вычислений (управления) и по объему используемой памяти. Длительность цикла обратно пропорциональна количеству потоков исполнения алгоритма, а объем используемой памяти и необходимая портовость прямо пропорциональны количеству потоков.

4. Для достижения вычислимости алгоритмов в реальном времени необходимо снизить время их выполнения с экспоненциального (или даже факториального) до

полиномиального (или даже линейного). Для этого необходимо представить исходный вычислительный алгоритм в виде графа и конденсировать его сильно связанные локальные компоненты в функции-акторы, которые инкапсулируются и в дальнейшем рассматриваются в качестве «черного ящика». Полученный после конденсации граф, принимая во внимание специфику отражаемого им вычислительного алгоритма, будет ациклическим ориентированным графом. Топологическая сортировка вершин такого графа выполнима за линейное время посредством топологического упорядочивания (с помощью алгоритма Кана) и других методов поиска топологической сортировки (DFS-алгоритмы Косарайю, Тарьяна, BFS-алгоритмы и др.).

5. Оптимальная техническая реализация конфигурации алгоритма определяется в результате решения задачи Штейнера о минимальном дереве. Эта задача заключается в нахождении кратчайшего дерева, связывающего заданные точки (терминалы) и дополнительно вводимые точки Штейнера. При определении оптимальных разводок микросхем, посредством которых технически реализуется вычислительная система, решается частный случай постановки этой задачи – прямоугольная задача Штейнера.

СПИСОК ИСТОЧНИКОВ / REFERENCES

1. Грибков А.А. Человек в цивилизации когнитивных технологий. *Философия и культура*. 2024;(1):22–33. https://doi.org/10.7256/2454-0757.2024.1.69678
   Gribkov A.A. Human Beings in a Civilization of Cognitive Technologies. *Philosophy and Culture.* 2024;(1):22–33. (In Russ.). https://doi.org/10.7256/2454-0757.2024.1.69678
2. Зеленский А.А., Кузнецов А.П., Илюхин Ю.В., Грибков А.А. Реализуемость управления движением промышленных роботов, станков с ЧПУ и мехатронных систем. Часть 2. *Вестник машиностроения.* 2023;102(3):213–220. https://doi.org/10.36652/0042-4633-2023-102-3-213-220
   Zelenskiy A.A., Kuznetsov A.P., Ilyukhin Yu.V., Gribkov A.A. Feasibility of Motion Control of Industrial Robots, CNC Machine Tools and Mechatronic Systems. Part 2. *Vestnik Mashinostroeniya.* 2023;102(3):213–220. (In Russ.). https://doi.org/10.36652/0042-4633-2023-102-3-213-220
3. Yamada H. Real-Time Computation and Recursive Functions Not Real-Time Computable. *IRE Transactions on Electronic Computers.* 1962;EC-11(6):753–760. https://doi.org/10.1109/TEC.1962.5219459
4. Уемов А.И. *Системный подход и общая теория систем.* Москва: «Мысль»; 1978. 272 с.
5. Грибков А.А. Эволюционный рост сложности систем. Часть 2. Сложность систем и энтропия. *Современная наука: актуальные проблемы теории и практики. Серия: Познание.* 2023;(6):68–72.
   Gribkov A.A. Evolutionary Growth of Complexity of Systems. Part 2. Complexity of Systems and Entropy. *Modern Science: Actual Problems of Theory and Practice. Series of "Cognition".* 2023;(6):68–72. (In Russ.).
6. Asperti A. A Formal Proof of Borodin-Trakhtenbrot's Gap Theorem. In: *Certified Programs and Proofs: Third International Conference, CPP 2013: Proceedings, 11–13 December 2013, Melbourne, VIC, Australia.* Cham: Springer; 2013. P. 163–177. https://doi.org/10.1007/978-3-319-03545-1_11
7. Зеленский А.А., Грибков А.А. Значение фактора портовости для конфигурирования цикла акторной системы управления реального времени. *Моделирование, оптимизация и информационные технологии*. 2025;13(1). https://doi.org/10.26102/2310-6018/2025.48.1.037

Zelenskii A.A., Gribkov A.A. An Importance of the Portability Factor for Configuring the Cycle of Real-Time Actuator Control System. *Modeling, Optimization and Information Technology.* 2025;13(1). (In Russ.). https://doi.org/10.26102/2310-6018/2025.48.1.037

8. Bartlett M., Frisch A.M., Hamadi Yo., Miguel Ia., Tarim S.A., Unsworth Ch. The Temporal Knapsack Problem and Its Solution. In: *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems: Second International Conference, CPAIOR 2005, 31 May – 1 June 2005, Prague, Czech Republic.* Berlin, Heidelberg: Springer; 2005. P. 34–48. https://doi.org/10.1007/11493853_5
9. Kahn A.B. Topological Sorting of Large Networks. *Communications of the ACM*. 1962;5(11):558–562. https://doi.org/10.1145/368996.369025
10. Кормен Т., Лейзерсон Ч., Ривест Р. Штайн К. *Алгоритмы: построение и анализ.* Москва: Издательский дом «Вильямс»; 2005. 1293 с.
Cormen Th.H., Leiserson Ch.E., Rivest R.L., Stein C. *Introduction to Algorithms.* Moscow: Izdatel'skii dom "Vil'yams"; 2005. 1293 p. (In Russ.).
11. Hong S., Rodia N.C., Olukotun K. On Fast Parallel Detection of Strongly Connected Components (SCC) in Small-World Graphs. In: *SC '13: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, 17–21 November 2013, Denver, Colorado, USA.* New York: Association for Computing Machinery; 2013. https://doi.org/10.1145/2503210.2503246
12. Tarjan R. Depth-First Search and Linear Graph Algorithms. *SIAM Journal on Computing*. 1972;1(2):146–160. https://doi.org/10.1137/0201010
13. Gabow H.N. Path-Based Depth-First Search for Strong and Biconnected Components. *Information Processing Letters*. 2000;74(3–4):107–114. https://doi.org/10.1016/S0020-0190(00)00051-X
14. Bundy A., Wallen L. Breadth-First Search. In: *Catalogue of Artificial Intelligence Tools.* Berlin, Heidelberg: Springer; 1984. P. 13. https://doi.org/10.1007/978-3-642-96868-6_25
15. Hashemi M., Gong Sh., Ni J., Fan W., Prakash B.A., Jin W. A Comprehensive Survey on Graph Reduction: Sparsification, Coarsening, and Condensation. In: *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24), 03–09 August 2024, Jeju, South Korea.* 2024. P. 8058–8066. https://doi.org/10.24963/ijcai.2024/891
16. Eswaran K.P., Tarjan R.E. Augmentation Problems. *SIAM Journal on Computing*. 1976;5(4):653–665. https://doi.org/10.1137/0205044
17. Гордеев Э.Н., Тарасцов О.Г. Задача Штейнера. Обзор. *Дискретная математика*. 1993;5(2):3–28.
Gordeev E.N., Tarastsov O.G. The Steiner Problem: A Survey. *Diskretnaya Matematika.* 1993;5(2):3–28. (In Russ.).
18. Melzak Z.A. On the Problem of Steiner. *Canadian Mathematical Bulletin.* 1961;4(2):143–148. https://doi.org/10.4153/CMB-1961-016-2
19. Cockayne E.J. On the Efficiency of the Algorithm for Steiner Minimal Trees. *SIAM Journal on Applied Mathematics*. 1970;18(1):150–159. https://doi.org/10.1137/0118014
20. Лисин А.В., Файзуллин Р.Т. Эвристический алгоритм поиска приближённого решения задачи Штейнера, основанный на физических аналогиях. *Компьютерная оптика.* 2013;37(4):503–510. https://doi.org/10.18287/0134-2452-2013-37-4-503-510
Lisin A.V., Faizullin R.T. Heuristic Algorithm for Finding Approximate Solution of Steiner Problem Based on Physical Analogies. *Computer Optics.* 2013;37(4):503–510. (In Russ.). https://doi.org/10.18287/0134-2452-2013-37-4-503-510

21. Kou L., Markowsky G., Berman L. A Fast Algorithm for Steiner Trees. *Acta Informatica*. 1981;15(2):141–145. https://doi.org/10.1007/BF00288961
22. Багов М.А, Кудаев В.Ч. Преобразование терминальной сети в сеть Штейнера. *Известия Кабардино-Балкарского научного центра РАН*. 2015;(6–2):31–37.
Bagov M.A., Kudaev V.C. Conversion of Terminal Net into Steiner Network. *News of the Kabardino-Balkarian Scientific Center of RAS.* 2015;(6–2):31–37. (In Russ.).
23. Kahng A.B., Robins G. *On Optimal Interconnections for VLSI.* New York: Springer; 1995. 286 p. https://doi.org/10.1007/978-1-4757-2363-2
24. Терещук В.С., Ференец А.В., Федоров Е.Ю. Особенности разводки сложных электрических цепей летательных аппаратов. В сборнике: *Поиск эффективных решений в процессе создания и реализации научных разработок в российской авиационной и ракетно-космической промышленности: Международная научно-практическая конференция: Том IV, 05–08 августа 2014 года, Казань, Россия.* Казань: Издательство Казанского государственного технического университета; 2014. С. 109–111.

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Зеленский Александр Александрович, кандидат технических наук, доцент, ведущий научный сотрудник Научно-производственного комплекса «Технологический центр», Москва, Зеленоград, Российская Федерация.
e-mail: zelenskyaa@gmail.com
ORCID: 0000-0002-3464-538X

Aleksandr A. Zelenskii, Candidate of Engineering Sciences, Docent, Leading researcher, Scientific and Production Complex "Technological Center", Moscow, Zelenograd, the Russian Federation.

Грибков Андрей Армович, доктор технических наук, ведущий научный сотрудник Научно-производственного комплекса «Технологический центр», Москва, Зеленоград, Российская Федерация.
e-mail: andarmo@yandex.ru
ORCID: 0000-0002-9734-105X

Andrei A. Gribkov, Doctor of Engineering Science, Leading researcher, Scientific and Production Complex "Technological Center", Moscow, Zelenograd, the Russian Federation.

Статья поступила в редакцию 28.06.2025; одобрена после рецензирования 05.08.2025; принята к публикации 12.08.2025.

The article was submitted 28.06.2025; approved after reviewing 05.08.2025; accepted for publication 12.08.2025.