НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
январь–февраль 2025 Том 25 № 1 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
January–February 2025 Vol. 25 No 1 http://ntv.ifmo.ru/en/
ISSN 2226-1494 (print) ISSN 2500-0373 (online)

НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

doi: 10.17586/2226-1494-2025-25-1-106-113
УДК 65.012.122

Планирование распределенных вычислений в недетерминированных системах

Николай Викторович Колесов[1], Елизавета Геннадьевна Литуненко[2]✉, Марина Владимировна Толмачева[3], Виктор Сергеевич Тюльников[4]

[1,2,3,4] АО «Концерн «ЦНИИ «Электроприбор», Санкт-Петербург, 197046, Российская Федерация

[1] kolesovnv@mail.ru, https://orcid.org/0000-0003-3287-7504
[2] lisa.litunenko@gmail.com✉, https://orcid.org/0000-0001-5280-0593
[3] marina-tolm@yandex.ru, https://orcid.org/0000-0003-0795-7617
[4] viktor.tyulnikov@gmail.com, https://orcid.org/0009-0005-5414-1567

Аннотация

Введение. Планирование вычислений занимает важное место в процессе проектирования распределенных систем обработки информации и управления. Эффективные алгоритмы планирования позволяют находить технические решения, адекватные существующим ограничениям. Это становится особенно актуально для вычислителей, размещенных на автономных носителях, таких как беспилотные летательные аппараты, необитаемые подводные аппараты и другие подвижные объекты. В работе предложены и исследованы алгоритмы планирования заданий для распределенной вычислительной недетерминированной системы в случае, когда время выполнения заданий известно неточно и задано в виде временных интервалов. **Метод.** Решение поставленной задачи достигается путем сведения ее к известной задаче flow shop планирования с последующим применением формализма разрешимых классов распределенных вычислительных систем. **Основные результаты.** Предложены два алгоритма планирования заданий для недетерминированной распределенной вычислительной системы. Алгоритмы допускают отсутствие изоморфизмов между графами заданий и графом межпроцессорных связей. В этом случае невозможно применение известных алгоритмов flow shop планирования. Алгоритмы предполагают предварительное приведение рассматриваемой системы к требуемому виду и базируются на положениях интервального анализа и понятии разрешимого класса распределенных вычислительных систем. Критерием оптимальности предложенных алгоритмов служит минимум среднего времени пребывания задания в системе. Дополнительно для второго алгоритма используется критерий минимума максимального отклонения от заданных директивных сроков. Для введенных разрешимых классов систем для обоих критериев сформулированы оптимальные алгоритмы планирования полиномиальной сложности. **Обсуждение.** Предложенные решения могут быть применены при планировании вычислений в распределенных вычислительных системах при неточно известных длительностях решаемых задач, в частности, при планировании экономических процессов.

Ключевые слова

планирование вычислений, распределенная вычислительная система реального времени, flow shop планирование, разрешимый класс систем, время пребывания задания в системе, директивный срок

Ссылка для цитирования: Колесов Н.В., Литуненко Е.Г., Толмачева М.В., Тюльников В.С. Планирование распределенных вычислений в недетерминированных системах // Научно-технический вестник информационных технологий, механики и оптики. 2025. Т. 25, № 1. С. 106–113. doi: 10.17586/2226-1494-2025-25-1-106-113

Scheduling distributed computations in non-deterministic systems

Nikolai V. Kolesov[1], Elisaveta G. Litunenko[2]✉, Marina V. Tolmacheva[3], Viktor S. Tiulnikov[4]

[1,2,3,4] Concern CSRI Elektropribor, JSC, Saint Petersburg, 197046, Russian Federation

[1] kolesovnv@mail.ru, https://orcid.org/0000-0003-3287-7504
[2] lisa.litunenko@gmail.com✉, https://orcid.org/0000-0001-5280-0593
[3] marina-tolm@yandex.ru, https://orcid.org/0000-0003-0795-7617
[4] viktor.tyulnikov@gmail.com, https://orcid.org/0009-0005-5414-1567

Abstract
Computation scheduling is very important in the design of distributed information processing and control systems. Effective scheduling algorithms allow developer to find technical solutions that are adequate to the existing constraints. This is especially important for computers located on autonomous carriers, such as unmanned aerial vehicles, autonomous underwater vehicles, and other vehicles. Scheduling algorithms for tasks in the distributed non-deterministic computing system, when the task execution time is known inaccurately and described as time interval, are proposed and researched. The solution of the problem is achieved by reducing it to a known problem of flow shop scheduling with subsequent application of the formalism of solvable classes of distributed computing systems. Authors propose two algorithms for scheduling tasks for a non-deterministic distributed computing system. Algorithms allow the absence of isomorphisms between task graphs and the graph of interprocessor communications for the system, and the presence of many information outputs and branches between tasks. In these conditions, it is impossible to use known algorithms of flow shop scheduling. Proposed algorithms assume preliminary reduction of the considered system to the required form and base on the provisions of interval analysis and the concept of a solvable class of distributed computing systems. Optimality criterion for proposed algorithms is the criterion of minimum average time a task remains in the system. Additionally, the criterion of minimum of maximum deviation from directive terms is used. For the introduced solvable classes of systems, optimal scheduling algorithms of polynomial complexity are proposed. These algorithms allow us to schedule computations in real distributed computing systems when the system deviates from the canonical form and when the durations of the tasks are not precisely known. Proposed algorithms can be applied when scheduling computations in real distributed computing systems and solving tasks with not precisely known durations and also, for example, in scheduling economic processes.

Keywords
computation scheduling, distributed real-time computing system, (flow shop)-scheduling, solvable class of systems, task residence time in the system

For citation: Kolesov N.V., Litunenko E.G., Tolmacheva M.V., Tiulnikov V.S. Scheduling distributed computations in non-deterministic systems. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2025, vol. 25, no. 1, pp. 106–113 (in Russian). doi: 10.17586/2226-1494-2025-25-1-106-113

Введение

Вопросы организации вычислений остаются в центре внимания разработчиков вычислительной техники. При этом для решения возникающих проблем привлекаются методы теории планирования. Здесь можно выделить планирование как в центрах коллективного пользования [1, 2], так и в системах реального времени [3–7]. В настоящей работе рассматривается направление планирования вычислений в распределенных системах реального времени. В качестве критериев могут быть использованы, например, минимум общего времени выполнения или минимум максимального отклонения от заданных директивных сроков. Отметим, что данными критериями не исчерпываются важные аспекты проектирования вычислительных систем.

Обсуждаемый подход принадлежит к области так называемого flow shop планирования [8–11]. В качестве практической интерпретации этой задачи часто называют обработку информации в многоканальных системах, когда в распределенной системе на общих процессорах выполняется совокупность алгоритмов, каждый из которых использует информацию со своего набора датчиков. Оптимальное решение проблемы планирования в общем случае может быть получено переборными алгоритмами, которые характеризуются экспоненциальной вычислительной сложностью, и в силу этого их применение в целом ряде приложений оказывается невозможным. По этой причине широкое распространение на практике получили приближенные алгоритмы, которые применены в работе. При этом если в классической постановке flow shop планирования рассматриваемая система представляет собой конвейер с линейным графом информационных связей, то в настоящей работе этот граф — ациклический и используемый подход основан на так называемых разрешимых классах систем — РКС-подход. Этот выбор не является случайным, так как РКС-подход в отличие от известных проявил достаточную эффективность при использовании различных критериев оптимальности. Так в работе [10] показано, что базовая идея РКС-подхода, предложенного для flow shop планирования с минимизацией общей длительности плана или максимального отклонения от заданных директивных сроков, оказывается эффективной и при минимизации среднего по заданиям времени пребывания задания в системе. Заметим, что это время складывается из двух составляющих — времени ожидания в очереди и времени исполнения. В результате разработаны два алгоритма flow shop планирования в недетерминированных системах. В первом алгоритме использован критерий минимума среднего по заданиям времени пребывания задания в системе, во втором алгоритме дополнительно добавлен критерий минимума максимального отклонения от заданных директивных сроков.

В настоящей работе приведены предварительные сведения о flow shop планировании и РКС-подходе. Сформулирована постановка проблемы и описаны предлагаемые алгоритмы планирования вычислений.

Предварительные сведения и постановка проблемы

Рассматриваемая распределенная вычислительная система включает процессоры, имеющие индивидуальную память для хранения кода программ и обменивающиеся информацией через каналы передачи данных. При этом рассматриваемое множество задач задано информационным графом $G(V, E)$, содержащим n компонент связности (задания). Каждое задание τ_j, $j = \overline{1, n}$

состоит из множества задач. Графы заданий изоморфны графу межпроцессорных связей системы. Считаем, что порядок выполнения заданий на процессорах одинаков. Планированию подлежат n независимых равноприоритетных заданий, каждое из которых состоит из множества задач $\tau_j = \{\tau_{j,k} | k = \overline{1, m}, j = \overline{1, n}\}$, где $\tau_{j,k}$ — k-я задача j-го задания; m — число задач в каждом задании (равное числу процессоров). Предположим, что для каждой задачи $\tau_{j,k}$ известна длительность ее выполнения по верхней границе $e_{j,k}$. Произведенное назначение заданий соответствует случаю flow shop системы. Это означает, что имеется m изоморфизмов φ_j: $G_j(S_j, T_j) \rightarrow$ $\rightarrow H(Q, P)$ $j = \overline{1, m}$, где $G_j(S_j, T_j)$ — граф межзадачных связей j-го задания; S_j — множество ребер графа G_j; T_j — множество вершин (задач); $H(Q, P)$ — граф межпроцессорных связей; Q — множество ребер (информационных связей) графа H; P — множество процессоров. Все введенные графы являются направленными, ациклическими, содержащими в общем случае не один путь между любыми вершинами. Графы характеризуются выделенным подмножеством входных вершин и одной выходной вершиной. Далее для рассматриваемой системы используется обозначение $C(P, \tau)$. Предполагается, что время решения задачи известно неточно и задается временным интервалом $\tilde{e}_j = [\underline{e}_j, \bar{e}_j]$, где $\underline{e}_j$ и $\bar{e}_j$ — нижняя и верхняя границы временного интервала соответственно. Пусть поставлена задача — определить такой порядок выполнения заданий, чтобы оптимизировать заданный критерий. Для решения задачи используем критерий — минимум среднего времени пребывания заданий в системе.

Каждый входной процессор P_i связан с выходным процессором P_0 некоторым вычислительным путем (последовательностью процессоров) $p_k = P_i, P_j, \ldots, P_0$. Обозначим интервал $\tilde{E}_j(p_k)$ как время выполнения пути p_k на j-м задании. Это время определяется как сумма длительностей интервалов $\tilde{e}_{j,i}$ выполнения задач j-го задания процессорами, которые находятся на этом пути. Используя нумерацию процессоров вдоль пути p_k, выражение для $\tilde{E}_j(p_k)$ можно записать следующим образом:

$$\tilde{E}_j(p_k) = \sum_{i=1}^{m_k} \tilde{e}_{j,i}, \qquad (1)$$

где m_k — число процессоров, принадлежащих пути p_k, а при суммировании используется интервальная операция [12]:

$$\tilde{e}_i + \tilde{e}_j = [\underline{e}_i, \bar{e}_i] + [\underline{e}_j, \bar{e}_j] = [\underline{e}_i + \underline{e}_j, \bar{e}_i + \bar{e}_j].$$

Определение 1. Интервал $\tilde{e}_i = [\underline{e}_i, \bar{e}_i]$ не меньше интервала $\tilde{e}_j = [\underline{e}_j, \bar{e}_j]$ ($\tilde{e}_i \geq \tilde{e}_j$), если $\underline{e}_i \geq \bar{e}_j$.

Таким образом, зоны неопределенности сравниваемых интервалов не должны пересекаться.

Назовем вычислительный путь p_j^* критическим для j-го задания, если время его выполнения на j-м задании является наибольшим среди всех остальных путей в системе. Очевидно, что для разных заданий, выполняемых в одной и той же системе, критические пути могут быть различными.

Для определения разрешимых классов потребуется ввести понятие доминирования на множестве процессоров.

Определение 2. Процессор P_q доминирует над процессором P_r ($P_q > P_r$), если

$$\min_j \underline{e}_{q,j} \geq \max_j \bar{e}_{r,j}.$$

Рассмотрим разрешимые классы систем, общее свойство которых заключается в следующем: для любого задания, выполняемого в системе, критический путь проходит через одни и те же процессоры. При этом определяющие свойства классов следующие.

Определение 3 (класс 1). Множество процессоров критического пути представляет собой последовательность $P_1 > P_2 > \ldots > P_{m^*}$, убывающую по отношению доминирования.

Определение 4 (класс 2). Множество процессоров критического пути представляет собой последовательность $P_1 < P_2 < \ldots < P_{m^*}$, возрастающую по отношению доминирования.

Определение 5 (класс 3). Множество процессоров критического пути представляет собой пару соединенных последовательностей:

$$P_1 < P_2 < \ldots < P_{h^*} > \ldots > P_{m^*-1} > P_{m^*}, \; 1 < h^* < m^*,$$

первая из которых возрастает, а вторая убывает по отношению доминирования (h^* — номер процессора стыковки двух последовательностей).

Приведение модели системы к каноническому виду

Как следует из раздела «Предварительные сведения и постановка проблемы», возможность применения известных алгоритмов flow shop планирования ограничена некоторыми требованиями, предъявляемыми к рассматриваемой системе: существование изоморфизмов между графами заданий (требование 1) и графом межпроцессорных связей (требование 2).

На практике эти требования могут не выполняться. В этом случае говорят о job shop системе. Однако это не всегда означает неприменимость к подобным системам известных алгоритмов flow shop планирования. Зачастую они могут быть применены после некоторых преобразований модели системы к требуемому каноническому виду. Приведем пример преобразования. На рис. 1 представлены четыре ациклических графа заданий. Вершины каждого графа соответствуют задачам, входящим в данное задание. Номер вершины является номером процессора, исполняющего данную задачу. Сплошной линией выделены задачи, решаемые в данном задании. Только первое задание (рис. 1, *a*) изоморфно графу межпроцессорных связей. Остальные задания неизоморфны, а значит, система не является flow shop системой. Отметим, что рассматриваемый случай сводится к flow shop системе. Для этого достаточно задания на рис. 1, *b–d* достроить до задания на рис. 1, *a*, введя в каждое из них соответствующие задачи нулевой длительности (изображены штриховой линией). Так, например, в задание (рис. 1, *b*) необходимо ввести задачи 2, 4, 5 нулевой длительности.

Условия оптимальности планов в системах из разрешимых классов

Приведем условия оптимальности flow shop планов при неопределенных длительностях задач и при использовании в качестве критерия минимума среднего по заданиям времени пребывания заданий в системе. Эти условия с точностью до обозначений совпадают с соответствующими условиями для случая, когда длительности задач известны точно [10]. Таким образом, для любого набора представителей из этих интервалов, а значит, и для интервалов в целом, алгоритм, основанный на известных условиях, будет формировать оптимальный план.

Для системы из класса 1 минимальное значение среднего по заданиям времени $\overline{F_1}(\pi)$ пребывания заданий в системе достигается в плане π, в котором задания упорядочены по неубыванию длительностей временных интервалов первых задач критического пути, т. е.

$$\tilde{e}^*_{1,1} \le \tilde{e}^*_{2,1} \le \ldots \le \tilde{e}^*_{n,1}. \quad (2)$$

Для системы из класса 2 минимальное значение среднего по заданиям времени $\overline{F_2}(\pi)$ пребывания заданий в системе достигается в плане π, который удовлетворяет двум условиям:

— задания упорядочены по неубыванию длительностей временных интервалов последних задач критического пути, т. е.

$$\tilde{e}^*_{1,m^*} \le \tilde{e}^*_{2,m^*} \le \ldots \le \tilde{e}^*_{n,m^*}; \quad (3)$$

— первое задание плана π удовлетворяет условию

$$j^* = \arg\min_j \sum_{i=1}^{m^*-1} \tilde{e}^*_{j,i}. \quad (4)$$

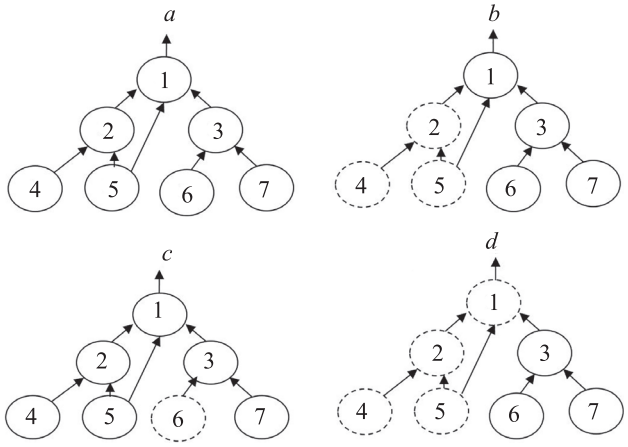

Рис. 1. Примеры заданий job shop системы (*a*), сводящейся к flow shop системе: отсутствуют задачи, решаемые процессорами 2, 4 и 5 (*b*); отсутствует задача, решаемая процессором 6 (*c*); отсутствуют задачи, решаемые процессорами 1, 2, 4 и 5 (*d*)

Fig. 1. Examples of jobs for job shop system adductable to flow shop system (*a*): no tasks for processors 2, 4, 5 to solve (*b*); no task for processor 6 to solve (*c*); no tasks for processors 1, 2, 4, 5 to solve (*d*)

Для системы из класса 3 минимальное значение среднего по заданиям времени $\overline{F_3}(\pi)$ пребывания заданий в системе достигается в плане π, который удовлетворяет двум условиям:

— задания упорядочены по неубыванию длительностей временных интервалов задач стыковки критического пути, т. е.

$$\tilde{e}^*_{1,h^*} \le \tilde{e}^*_{2,h^*} \le \ldots \le \tilde{e}^*_{n,h^*}; \quad (5)$$

— первое задание плана π удовлетворяет условию

$$j^* = \arg\min_j \sum_{i=1}^{h^*-1} \tilde{e}^*_{j,i}. \quad (6)$$

Алгоритм планирования в системах общего вида

Предварительно отметим, что для недетерминированной системы может не существовать оптимального плана, поскольку описание ее экземпляров может значительно различаться. Детерминированные экземпляры системы получаются из недетерминированных систем путем замены интервала времени выполнения каждой задачи на конкретное значение из этого интервала. В результате в неблагополучном случае разным экземплярам системы могут соответствовать разные оптимальные планы.

Очевидно, что на практике для flow shop системы общего вида, описанной в постановке задачи, условия ее принадлежности к тому или иному разрешимому классу чаще всего не выполняются. В частности, могут не совпадать критические пути разных работ, может нарушаться отношение доминирования между машинами, а если оно и выполняется, то может не быть упорядоченных по этому отношению цепочек машин, характерных для разрешимых классов. В результате исчезают гарантии оптимальности алгоритмов, описанных в разделе «Условия оптимальности планов в системах из разрешимых классов». В связи с этим для системы общего вида, не принадлежащей ни к одному из разрешимых классов, планирование осуществляется в соответствии с положениями РКС-подхода [3]. При этом алгоритмы планирования являются приближенными, но справедливыми для любой из рассматриваемых систем. Они имеют рекурсивный характер и выполняются за число шагов, не большее, чем число заданий.

Алгоритм 1.

Шаг 1. Найти в системе $C' = (P, \tau')$ (при условии $\tau' = \tau$, $C' = C$) на каждом шаге рекурсии вычислительный путь p_k, который характеризуется наибольшим значением суммы $\tilde{E}(p_k) = \sum_{j=1}^{n} \sum_{i=1}^{m_k} \tilde{e}_{j,i}$.

Шаг 2. Определить, к какому разрешимому классу наиболее близка рассматриваемая на шаге 1 система $C' = (P, \tau')$ на основе геометрического классификационного правила [3], предполагающего аппроксимацию зависимости средней длительности решаемых задач от номера машины и формирование оценки δ точности аппроксимации (достоверности классификации). В случае, если достигнутая на k-ом шаге достоверность классификации окажется меньше значения достоверности на предыдущем шаге, то планирование завершается

с сохранением упорядоченности (k – 1)-го шага, иначе перейти к шагу 3.

Шаг 3. Определить с использованием соответствующих алгоритмов, основанных на условиях оптимальности (раздел «Условия оптимальности планов в системах из разрешимых классов»), упорядоченность оставшихся заданий на интервале свободных позиций формируемого плана. Исключить первое задание в этой упорядоченности из множества τ' планируемых заданий. Если результирующее множество планируемых заданий непустое, то перейти к шагу 1, иначе конец.

Двухкритериальный алгоритм планирования

С учетом алгоритма 1 рассмотрим более сложную задачу, когда при планировании вместо одного используются два критерия оптимальности. В данном случае к критерию минимума среднего времени пребывания задания в системе добавляется критерий минимума максимального отклонения моментов завершения заданий от их директивных сроков. Далее для этой ситуации предлагается приближенный алгоритм flow shop планирования. Рассмотрим суть алгоритма. В соответствии с правилами РКС-подхода, продемонстрированными выше (раздел «Алгоритм планирования в системах общего вида»), построим план для критерия минимума максимального отклонения от заданных директивных сроков. Выполним корректировку плана, стремясь удовлетворить условия (2)–(6), не нарушая определенных для заданий директивных сроков. Корректировку плана осуществим с использованием допустимых перестановок заданий. При этом может оказаться, что в полной мере желаемое упорядочение получить невозможно. В этом случае необходимо допустить пониженное качество плана. Рассмотрим сначала проблему flow shop планирования для критерия минимума максимального отклонения от заданных директивных сроков.

Предположим, что для каждого задания из исходного списка определен директивный срок $\{d_j | j = \overline{1, n}\}$ их завершения. Рассмотрим некоторый план $\pi = (\tau_1, \tau_2, \ldots, \tau_n)$ выполнения этих заданий. Каждое задание при этом будет характеризоваться временным интервалом завершения $\{\tilde{t}_j | j = \overline{1, n}\}$, для которого может быть определено отклонение $\{\tilde{\delta}_j | j = \overline{1, n}\}$ относительно заданного директивного срока: $\tilde{\delta}_j = \tilde{t}_j - d_j$. Будем считать, что директивный срок для j-го задания выполняется, если $\tilde{\delta}_j \le 0$. Проблема состоит в поиске алгоритма составления плана, минимизирующего максимальное отклонение $\tilde{\delta}_{\max} = \max\limits_j \tilde{\delta}_j$.

Воспользовавшись известными результатами для детерминированных систем [3], приведем условия оптимальности планов для рассматриваемых разрешимых классов недетерминированных систем. Приближенный алгоритм, рассчитанный на системы общего вида, будет совпадать с алгоритмом 1. Условия оптимальности требуют упорядочивания заданий по возрастанию значений либо директивных сроков (класс 2), либо некоторых функций от директивных сроков и длительностей задач (виртуальных директивных сроков) в виде:

— класс 1 — $\tilde{d}_j' = d_j - \sum\limits_{i=2}^{m^*} \tilde{e}_{i,j}^*$;

— класс 2 — заданные директивные сроки;

— класс 3 — $\tilde{d}_j'' = d_j - \sum\limits_{i=h^*+1}^{m^*} \tilde{e}_{i,j}^*$.

В данном случае, при упорядочивании заданий классов 1 и 3 (подобно ситуации заданий (2), (3) и (5)) можно столкнуться с ситуацией несравнимости интервальных величин, что также может привести к отсутствию оптимального плана.

Рассматриваемая проблема является более сложной, поскольку оптимизация по критерию минимума среднего времени пребывания задания в системе должна осуществляться при ограничениях, определяемых директивными сроками для заданий. Напомним, что в общем случае поиск оптимального решения не будет выполнен. При этом в качестве основы алгоритма используем сортировку задач по соответствующим значениям (2)–(6).

Предварительно определим основные понятия и докажем утверждения, которые послужат теоретическим основанием для предлагаемого алгоритма.

Определение 6. Перестановка — преобразование плана, при котором перемещается в новую позицию одно задание исходного плана.

Определение 7. Соседняя парная перестановка — преобразование плана, при котором два соседних задания исходного плана меняются местами.

Определение 8. Допустимая (недопустимая) парная перестановка — перестановка, при которой не нарушаются (нарушаются) директивные сроки всех планируемых заданий.

Заметим, что используемая далее в предлагаемом алгоритме процедура обменной (пузырьковой) сортировки представляет собой последовательность соседних парных перестановок.

Утверждение 1. Каждая соседняя парная перестановка, реализуемая в рамках обменной сортировки заданий по значению длительности соответствующих задач (2)–(6), уменьшает среднее время пребывания задания в системе для получаемого плана.

Доказательство. Запишем известное выражение для среднего по заданиям времени пребывания задания в системе $\overline{F}_1(\pi)$ в случае системы из класса 1 [13]:

$$\overline{F}_1(\pi) = \frac{1}{n}\sum_{k=1}^{n}\left[\sum_{j=1}^{k}\tilde{e}_{j,1} + \sum_{i=2}^{m}\tilde{e}_{k,i}\right] = \frac{1}{n}\left[\sum_{k=1}^{n}\sum_{j=1}^{k}\tilde{e}_{j,1} + \sum_{k=1}^{n}\sum_{i=2}^{m}\tilde{e}_{k,i}\right]. \quad (7)$$

Проанализируем второе слагаемое в скобках выражения (7). Каждое k-е слагаемое ее внешней суммы представляет собой сумму длительностей всех стадий, кроме первой, выполняемых в k-м процессе. Поскольку такая сумма длительностей присутствует в двойной сумме для каждого процесса, ее значение не зависит от упорядоченности процессов, а значит, при последующем анализе второе слагаемое можно исключить из (7) и перейти к минимизации выражения

$$\overline{F}'_1(\pi) = \frac{1}{n}\sum_{k=1}^{n}\sum_{j=1}^{k}\tilde{e}_{j,1}. \quad (8)$$

Если далее в выражении (8) развернуть внутреннюю сумму и привести подобные члены, то получим

$$\overline{F}'_1(\pi) = \frac{1}{n}\sum_{k=1}^{n}[(n-k+1)\tilde{e}_{k,1}]. \quad (9)$$

Предположим, что в результате перестановки меняются местами задания с q-ой и $q+1$-ой позиций. Для результирующего плана $\hat{\pi}$ выражение (9) имеет вид

$$\overline{F}'_1(\hat{\pi}) = \frac{1}{n}\sum_{k=1}^{n}[(n-k+1)\hat{\tilde{e}}_{k,1}]. \quad (10)$$

Вычислим и проанализируем разность выражений (9) и (10). Поскольку в суммах, входящих в эти выражения все слагаемые, кроме двух участвующих в перестановке, совпадают, получим

$$\overline{F}'_1(\pi) - \overline{F}'_1(\hat{\pi}) = \frac{1}{n}[(n-q+1)\tilde{e}_{q,1} - (n-q-1+1)\tilde{e}_{q+1,1} -$$

$$-(n-q+1)\tilde{e}_{q+1,1} + (n-q-1+1)\tilde{e}_{q,1}] =$$

$$= \frac{1}{n}[(n-q+1)(\tilde{e}_{q,1} - \tilde{e}_{q+1,1}) - (n-q+1+1)(\tilde{e}_{q,1} - \tilde{e}_{q+1,1})] =$$

$$= \frac{1}{n}[\tilde{e}_{q,1} - \tilde{e}_{q+1,1}] > 0.$$

Таким образом, значение критерия в результате перестановки уменьшается.

По изложенной схеме можно получить аналогичный результат для разрешимых классов 2 и 3. Этот результат позволяет рассчитывать, что алгоритм планирования, основанный на процедуре обменной сортировки, будет достаточно эффективен.

Содержание предлагаемого алгоритма планирования в упрощенном виде выглядит следующим образом.

Алгоритм 2.

Шаг 1. Построить план $\pi_{дс}$ с использованием критерия минимума максимального отклонения от заданных директивных сроков.

Шаг 2. Построить план $\pi_{вп}$ с использованием критерия минимума среднего времени пребывания задания в системе.

Шаг 3. Осуществить переупорядочивание (сортировку) заданий плана $\pi_{дс}$ методом обменной сортировки, следуя упорядоченности заданий в плане $\pi_{вп}$ до первого нарушения директивных сроков.

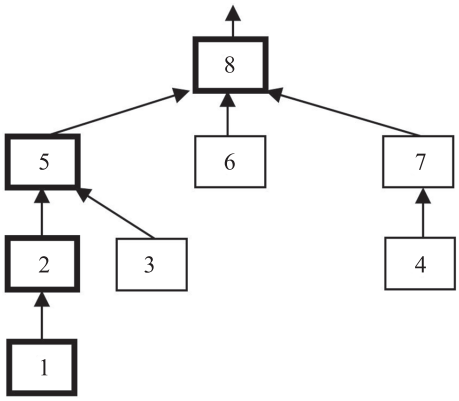

Рис. 2. Пример системы (жирными линиями выделены процессоры критического пути)

Fig. 2. Example of the system (processors of critical path marked with bold frames)

Пример реализации

Рассмотрим пример, иллюстрирующий работу алгоритма 1. Построим план для четырех заданий, выполняющихся в системе, показанной на рис. 2, и включающей восемь процессоров.

В табл. 1 приведены интервальные значения длительностей задач, входящих в выполняемые задания и выраженные в условных единицах.

Как уже отмечалось в разделе «Предварительные сведения и постановка проблемы», в любой flow shop системе каждый процессор выполняет одну и только одну задачу из каждого конкретного задания. В рассматриваемой системе это не так. Действительно, в некоторых клетках табл. 1 присутствуют прочерки. Они свидетельствуют о том, что процессор, соответствующий данному столбцу табл. 1, не участвует в выполнении задания, соответствующего данной строке табл. 1. Отсюда следует, что система не удовлетворяет указанному требованию. В таком виде к системе невозможно применить алгоритмы планирования. Решить эту проблему можно путем приведения системы к каноническому виду, введя задачи нулевой длительности как показано в табл. 2.

Это преобразование позволяет рассчитать время выполнения каждого вычислительного пути в соответствии с выражением (1):

$$\tilde{E}_1(1, 2, 4, 8) = [301; 360], \tilde{E}_2(3, 5, 8) = [199; 214],$$

$$\tilde{E}_3(6, 8) = [60; 80], \tilde{E}_4(4, 7, 8) = [130; 160].$$

Таблица 1. Интервалы значений длительности [минимальная; максимальная] выполнения задач на процессорах исходной системы, усл. ед.

Table 1. Duration value intervals [minimum; maximum], for processors of initial system, in conventional units

Задания	Процессоры							
	1	2	3	4	5	6	7	8
1	[115; 120]	[137; 140]	[110; 114]	—	[70; 71]	[50; 55]	—	[10; 12]
2	[110; 112]	[135; 136]	—	[96; 100]	[76; 77]	—	[32; 36]	[16; 18]
3	[90; 94]	[131; 132]	—	[92; 98]	[72; 74]	[56; 58]	[35; 40]	[12; 17]
4	[95; 100]	[133; 134]	—	[94; 100]	[77; 80]	—	[34; 38]	[19; 20]

Таблица 2. Интервалы значений длительности [минимальная; максимальная] выполнения задач на процессорах преобразованной системы, усл. ед.

Table 2. Duration value intervals [minimum; maximum], for processors of transformed system, in conventional units

Задания	Процессоры							
	1	2	3	4	5	6	7	8
1	[115; 120]	[137; 140]	[110; 114]	[0; 0]	[70; 71]	[50; 55]	[0; 0]	[10; 12]
2	[110; 112]	[135; 136]	[0; 0]	[96; 100]	[76; 77]	[0; 0]	[32; 36]	[16; 18]
3	[90; 94]	[131; 132]	[0; 0]	[92; 98]	[72; 74]	[56; 58]	[35; 40]	[12; 17]
4	[95; 100]	[133; 134]	[0; 0]	[94; 100]	[77; 80]	[0; 0]	[34; 38]	[19; 20]

Примечание. Серым цветом отмечены процессоры критического пути.

В соответствии с полученными значениями $\tilde{E}_j$, наиболее загруженным — критическим вычислительным путем — является путь, включающий в себя процессоры 1, 2, 5, 8: $p_k = P_1, P_2, P_5, P_8$. На рис. 2 процессоры критического пути выделены жирной рамкой.

Рассматриваемая система относится к классу 3. В данном случае оптимальный план существует и в соответствии с (5) имеет вид

$$\pi_1 = \tau_3\tau_4\tau_2\tau_1,$$

поскольку условие (6) тоже выполняется. Таким образом, в данном случае оптимальный по критерию минимума среднего по заданиям времени пребывания заданий в системе план формируется за один шаг. При этом важно отметить, что оптимальный результат получен с использованием алгоритма 1, обладающего полиномиальной вычислительной сложностью, в отличие от алгоритма полного перебора факториальной сложности.

Обсуждение результатов

Важно отметить, что проблема планирования каких-либо действий возникает в разных приложениях и в разнообразных постановках. Среди возможных приложений, кроме вычислительных процессов, необходимо выделить планирование бизнес-процессов, технологических процессов [13, 14], движения транспорта [15], и т. п. При этом разнообразие представленных задач весьма велико. Часто рассматриваемые в различных приложениях математические постановки задачи оказываются близки или даже идентичны, что способствует перетеканию предлагаемых решений из одной прикладной области в другую. По этой причине выполненное исследование может быть использовано и в экономических приложениях, и в планировании движения каких-либо объектов.

Исследована вычислительная сложность предложенных алгоритмов, которая в обоих случаях характеризуется полиномиальной зависимостью. Это можно утверждать на том основании, что базовой процедурой для алгоритмов служит процедура сортировки полиномиальной сложности.

Заключение

В работе рассмотрены вопросы планирования вычислительных процессов в распределенных недетерминированных системах реального времени, т. е. системах, для которых время решения задач известно неточно и задано интервалами. Общей особенностью предложенных приближенных алгоритмов являются их простота и учет неопределенности длительности выполнения заданий. В процессе планирования на основе алгоритма 1 использован критерий минимума среднего времени пребывания заданий в системе. При планировании на основе алгоритма 2 использовано два критерия — минимум среднего времени пребывания заданий в системе и минимум максимального отклонения моментов завершения заданий от их директивных сроков.

Литература

1. Toporkov V., Yemelyanov D.,Toporkova A. Coordinated global and private job-flow scheduling in grid virtual organizations // Simulation Modelling Practice and Theory. 2021. V. 107. P. 102228. https://doi.org/10.1016/j.simpat.2020.102228
2. Малашенко Ю.Е., Назарова И.А. Управление ресурсоемкими разнородными вычислительными заданиями с директивными сроками окончания // Известия РАН. Теория и системы управления. 2012. № 5. С. 15–22.
3. Колесов Н.В., Толмачева М.В., Юхта П.В. Системы реального времени. Планирование, анализ, диагностирование // СПб.: ОАО «Концерн «ЦНИИ «Электроприбор», 2014. 180 с.
4. Liu J.W.S. Real-Time Systems. Prentice Hall, 2000. 610 p.
5. Cottet F., Delacroix J., Kaiser C., Mammeri Z. Scheduling in Real-Time Systems. J. Wiley, 2002. 288 p.

References

1. Toporkov V., Yemelyanov D.,Toporkova A. Coordinated global and private job-flow scheduling in grid virtual organizations. *Simulation Modelling Practice and Theory*, 2021, vol. 107, pp. 102228. https://doi.org/10.1016/j.simpat.2020.102228
2. Malashenko Y.E., Nazarova I.A. Controlling computationally intensive heterogeneous computational tasks with directive deadlines. *Journal of Computer and Systems Sciences International*, 2012, vol. 51, no. 5, pp. 628–635. https://doi.org/10.1134/S1064230712050024
3. Kolesov N.V., Tolmacheva M.V., Iukhta P.V. *Real-time Systems. Planning, Analysis, Diagnostics*. St. Petersburg, CSRI ELEKTROPRIBOR Publ., 2014, 180 p. (in Russian)
4. Liu J.W.S. *Real-Time Systems*. Prentice Hall, 2000, 610 p.
5. Cottet F., Delacroix J., Kaiser C., Mammeri Z. *Scheduling in Real-Time Systems*. J. Wiley, 2002, 288 p.

6. Cheng A.M.K. Real-Time Systems: Scheduling, Analysis, and Verification. Wiley-Interscience, 2002. 552 p.
7. Глонина А.Б., Балашов В.В. О корректности моделирования модульных вычислительных систем реального времени с помощью сетей временных автоматов // Моделирование и анализ информационных систем. 2018. Т. 25. № 2. С. 174–192. https://doi.org/10.18255/1818-1015-2018-2-174-192
8. Choudhari, S.D., Khanna R. Flow shop scheduling problem with loading and unloading time // International Journal of Mathematics Research. 2017. V. 9. N 1. P. 13–26.
9. Kurniawan D., Radja A.C., Suprayogi, Halim A.H. A Flow Shop Batch Scheduling and Operator Assignment Model to Minimise Actual Flow Time // Proc. of the Asia Pacific Industrial Engineering & Management Systems Conference. 2017. P. 1–6.
10. Грузликов А.М., Колесов Н.В., Литуненко Е.Г., Скородумов Ю.М. Маршрутизация в сетях автономных необитаемых подводных аппаратов // Научно-технический вестник информационных технологий, механики и оптики. 2021. Т. 21. № 6. С. 984–990. https://doi.org/10.17586/2226-1494-2021-21-6-984-990
11. Колесов Н.В., Литуненко Е.Г., Скородумов Ю.М., Толмачева М.В. Планирование заданий в распределенной вычислительной системе на кристалле с минимизацией потребляемой мощности // Научно-технический вестник информационных технологий, механики и оптики. 2023. Т. 23. № 5. С. 1001–1008. https://doi.org/10.17586/2226-1494-2023-23-5-1001-1008
12. Жолен Л., Кифер М., Дидри О., Вальтер Э. Прикладной интервальный анализ: с примерами по оцениванию параметров и состояний, робастному управлению и робототехнике. Москва; Ижевск: Институт компьютерных исследований, 2005. 467 с.
13. Brucker P. Scheduling Algorithms. Springer, 2007. 371 p.
14. Лазарев А.А. Метрики в задачах теории расписаний // Математическая теория управления и ее приложения. 15-я мультиконференция по проблемам управления. СПб, 2022. С.146–148.
15. Белоусов Ф.А., Неволин И.В., Хачатрян Н.К. Моделирование и оптимизация планов грузовых железнодорожных перевозок, выполняемых транспортным оператором // Бизнес-информатика. 2020. Т. 14. № 2. С. 21–35. https://doi.org/10.17323/2587-814X.2020.2.21.35

6. Cheng A.M.K. *Real-Time Systems: Scheduling, Analysis, and Verification*. Wiley-Interscience, 2002, 552 p.
7. Glonina A.B., Balashov V.V. On the correctness of Real-Time Modular Computer Systems Modeling with stopwatch automata networks. *Automatic Control and Computer Sciences*, 2018, vol. 52, no. 7, pp. 817–827. (in Russian). https://doi.org/10.3103/S0146411618070271
8. Choudhari, S.D., Khanna R. Flow shop scheduling problem with loading and unloading time. *International Journal of Mathematics Research*, 2017, vol. 9, no. 1, pp. 13–26.
9. Kurniawan D., Radja A.C., Suprayogi, Halim A.H. A Flow Shop Batch Scheduling and Operator Assignment Model to Minimise Actual Flow Time. *Proc. of the Asia Pacific Industrial Engineering & Management Systems Conference*, 2017, pp. 1–6.
10. Gruzlikov A.M., Kolesov N.V., Litunenko E.G., Skorodumov Yu.M. Routing in networks of autonomous underwater vehicles. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2021, vol. 21, no. 6, pp. 984–990. (in Russian). https://doi.org/10.17586/2226-1494-2021-21-6-984-990
11. Kolesov N.V., Litunenko E.G., Skorodumov Iu.M., Tolmacheva M.V. Job scheduling in a distributed computing system on a chip with power consumption minimization. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2023, vol. 23, no. 5, pp. 1001–1008. (in Russian). https://doi.org/10.17586/2226-1494-2023-23-5-1001-1008
12. Jaulin L., Kieffer M., Didrit O, Walter É. *Applied Interval Analysis With Examples in Parameter and State Estimation, Robust Control and Robotics*. Springer, 2001, 379 p. https://doi.org/10.1007/978-1-4471-0249-6
13. Brucker P. *Scheduling Algorithms*. Springer, 2007, 371 p.
14. Lazarev A.A. Metrics for problem of scheduling theory. *Mathematical Control Theory and Its Applications. 15th Multiconference on Control Problems*. St. Petersburg, 2022, pp. 146–148. (in Russian)
15. Belousov F.A., Nevolin I.V., Khachatryan N.K. Modeling and optimization of plans for railway freight transport performed by a transport operator. *Business Informatics*, 2020, vol. 14, no. 2, pp. 21–35. (in Russian). https://doi.org/10.17323/2587-814X.2020.2.21.35

Авторы

Колесов Николай Викторович — доктор технических наук, профессор, главный научный сотрудник, АО «Концерн «ЦНИИ «Электроприбор», Санкт-Петербург, 197046, Российская Федерация, SC 6602000556, https://orcid.org/0000-0003-3287-7504, kolesovnv@mail.ru

Литуненко Елизавета Геннадьевна — младший научный сотрудник, АО «Концерн «ЦНИИ «Электроприбор», Санкт-Петербург, 197046, Российская Федерация, SC 57796319900, https://orcid.org/0000-0001-5280-0593, lisa.litunenko@gmail.com

Толмачева Марина Владимировна — кандидат технических наук, старший научный сотрудник, АО «Концерн «ЦНИИ «Электроприбор», Санкт-Петербург, 197046, Российская Федерация, SC 16305525100, https://orcid.org/0000-0003-0795-7617, marina-tolm@yandex.ru

Тюльников Виктор Сергеевич — инженер-программист, АО «Концерн «ЦНИИ «Электроприбор», Санкт-Петербург, 197046, Российская Федерация, https://orcid.org/0009-0005-5414-1567, viktor.tyulnikov@gmail.com

Authors

Nikolai V. Kolesov — D.Sc., Professor, Chief Researcher, Concern CSRI Elektropribor, JSC, Saint Petersburg, 197046, Russian Federation, 197046, Российская Федерация, SC 6602000556, https://orcid.org/0000-0003-3287-7504, kolesovnv@mail.ru

Elisaveta G. Litunenko — Junior Researcher, Concern CSRI Elektropribor, JSC, Saint Petersburg, 197046, Russian Federation, SC 57796319900, https://orcid.org/0000-0001-5280-0593, lisa.litunenko@gmail.com

Marina V. Tolmacheva — PhD, Senior Researcher, Concern CSRI Elektropribor, JSC, Saint Petersburg, 197046, Russian Federation, SC 16305525100, https://orcid.org/0000-0003-0795-7617, marina-tolm@yandex.ru

Viktor S. Tiulnikov — Software Engineer, Concern CSRI Elektropribor, JSC, Saint Petersburg, 197046, Russian Federation, https://orcid.org/0009-0005-5414-1567, viktor.tyulnikov@gmail.com

Статья поступила в редакцию 07.11.2025
Одобрена после рецензирования 25.12.2025
Принята к печати 26.01.2025

Received 07.11.2025
Approved after reviewing 25.12.2025
Accepted 26.01.2025