НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
сентябрь–октябрь 2023 Том 23 № 5 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
September–October 2023 Vol. 23 No 5 http://ntv.ifmo.ru/en/
ISSN 2226-1494 (print) ISSN 2500-0373 (online)

doi: 10.17586/2226-1494-2023-23-5-1001-1008
УДК 65.012.122

Планирование заданий в распределенной вычислительной системе на кристалле с минимизацией потребляемой мощности

Николай Викторович Колесов[1], Елизавета Геннадьевна Литуненко[2]✉, Юрий Михайлович Скородумов[3], Марина Владимировна Толмачева[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] skorum@mail.ru, https://orcid.org/0000-0003-0825-1720
[4] marina-tolm@yandex.ru, https://orcid.org/0000-0003-0795-7617

Аннотация

Введение. Планирование вычислений занимает важное место в процессе проектирования распределенных систем обработки информации и управления, особенно в условиях ограничения вычислительных и энергетических ресурсов системы. Эти ограничения особенно остро проявляются для вычислителей, размещенных на автономных носителях, таких как беспилотные летательные аппараты, необитаемые подводные и надводные аппараты. В данной работе представлен метод планирования заданий в распределенной вычислительной системе на кристалле, который позволяет сократить потребляемую системой мощность. **Метод.** Предложенный метод включает два этапа. На первом этапе осуществляется назначение заданий с определением энергоэффективной архитектуры системы, характеризующейся минимальной потребляемой мощностью. На втором этапе выполняется планирование заданий с учетом критерия, позволяющего минимизировать среднее время пребывания задания в системе. Особенностью решаемой задачи является появление в общем случае у разрабатываемой системы после первого этапа более одного информационного выхода, что не позволяет применять к системе ни один из известных методов планирования. **Основные результаты.** Первый этап метода реализован введением дополнительных процессоров с одновременным снижением тактовой частоты и напряжения питания. Для второго этапа предложен алгоритм планирования заданий, который выполняет предварительное построение для каждого выхода системы частного плана с дальнейшим их интегрированием в общий план путем применения эвристической процедуры. Приведен пример решения для пояснения работы алгоритма планирования. **Обсуждение.** Достоинством эвристического алгоритма является возможность планирования вычислений с учетом критериев минимумов потребляемой мощности и среднего времени пребывания задания в системе. Это позволяет повысить энергоэффективность решения задач в распределенных вычислительных системах на кристалле и автономность систем, в которых они применяются. Предложенный алгоритм обладает полиномиальной сложностью, поэтому благодаря относительной простоте алгоритма возможно его применение при планировании и перепланировании заданий в реальном времени в сложных системах.

Ключевые слова

планирование вычислений, распределенная вычислительная система реального времени, снижение потребляемой мощности, flow shop планирование, энергоэффективные вычисления

Благодарности

Работа выполнена при финансовой поддержке Российского научного фонда (проект № 22-29-00339).

Ссылка для цитирования: 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. doi: 10.17586/2226-1494-2023-23-5-1001-1008

Job scheduling in a distributed computing system on a chip with power consumption minimization

Nikolai V. Kolesov[1], Elizaveta G. Litunenko[2]✉, Iurii M. Skorodumov[3], Marina V. Tolmacheva[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] skorum@mail.ru, https://orcid.org/0000-0003-0825-1720
[4] marina-tolm@yandex.ru, https://orcid.org/0000-0003-0795-7617

Abstract

Scheduling of computing operations takes an important place in the process of distributed information processing and control systems design, especially in conditions of limited energy resources of the system. This becomes especially important for computers located on autonomous carriers, such as unmanned aerial vehicles, autonomous underwater vehicles, etc. The energy resources in such systems are limited that leads to high requirements for the energy efficiency of the carrier systems including computing ones. The paper presents the job scheduling method for a distributed computing system on a chip which allows reducing the power consumed by the system. The proposed task scheduling method includes two stages. At the first stage, jobs are assigned with the determination of an energy-efficient architecture of the system characterized by the minimum power consumption. At the second stage, jobs are scheduled taking into account the criterion that minimalizes the average job implementation time. A feature of the problem being solved in this case is the necessity of job scheduling in the system with more than one information output which does not allow applying any of the known scheduling methods to the system. The first stage of the proposed method is implemented by implementation additional processors with a simultaneous decrease in the clock frequency and supply voltage. For the second stage of the method, the job scheduling algorithm is proposed which involves the preliminary construction of a private schedule for each output of the system with further integration of these schedules into the general schedule using a heuristic procedure. The scheduling algorithm functioning is illustrated by an example of a solution for a simple system. The advantage of the proposed heuristic method is the possibility of scheduling calculations, taking into account criteria of the minimum power consumption and the minimum average residence time of a task in the system simultaniously. This makes it possible to increase the energy efficiency of solving problems in distributed computing systems on a chip, which contributes to increasing the autonomy of systems in which they are used in. The proposed algorithm has polynomial complexity, therefore, due to the relative simplicity of the algorithm, it can be used for scheduling and rescheduling jobs in real time for complex systems.

Keywords

computing scheduling, real-time distributed computing system, power consumption reduction, flow shop scheduling, energy-efficient computing

Acknowledgements

This work was supported by the project no. 22-29-00339 of the Russian Science Foundation, Russian Federation.

For citation: 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). doi: 10.17586/2226-1494-2023-23-5-1001-1008

Введение

Проблема планирования каких-либо действий возникает в разных приложениях и постановках. Среди возможных приложений: планирование бизнес-процессов, технологических процессов [1], движения транспорта [2], вычислительных процессов [3], и др. При этом разнообразие представленных задач весьма велико. Нередко рассматриваемые в различных приложениях математические постановки задачи оказываются близки или даже идентичны, что способствует перетеканию предлагаемых решений из одной прикладной области в другую. В области планирования вычислительных процессов можно выделить планирование как в центрах коллективного пользования [3], так и в системах реального времени [4–8]. В настоящей работе рассмотрено второе направление — планирование вычислений в распределенных системах реального времени, а именно планирование распределенных вычислений в многоядерной системе на кристалле. Под планированием вычислений обычно понимают определение для каждой решаемой задачи из заданного множества временного интервала исполнения на выделенном для нее процессоре.

В качестве критериев могут быть использованы, например, минимум общего времени выполнения или минимум максимального отклонения от заданных директивных сроков. Данными критериями, конечно, не исчерпываются важные аспекты проектирования вычислительных систем. В частности, одной из ключевых проблем при проектировании вычислительных систем всегда было снижение энергопотребления. В последние десятилетия данной проблеме уделяется особое внимание [9–12], в том числе и в связи с ее обсуждением в отношении систем на кристалле, когда снижение энергопотребления (рассеиваемой мощности) достигается за счет снижения тактовой частоты и напряжения питания. В предлагаемых алгоритмах планирования этот критерий будет использоваться наряду с критерием минимума среднего времени пребывания задания в системе.

Обсуждаемый подход принадлежит к области так называемого flow shop планирования, когда рассматриваемая система реализуется на совокупности вы-

числительных конвейеров. В качестве практической интерпретации этой задачи часто называют обработку информации в многоканальных системах, когда в распределенной системе выполняется совокупность алгоритмов, каждый из которых использует информацию со своего набора датчиков. Оптимальное решение проблемы планирования 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 субоптимальные алгоритмы, которые и рассматриваются далее. При этом, если в классической постановке рассматриваемая система представляет собой конвейер с линейным графом информационных связей, то в настоящей работе этот граф — ациклический, а рассматриваемый в статье подход основан на использовании так называемых разрешимых классов систем (РКС-алгоритм). В работе [13] показано, что базовая идея РКС-алгоритма, предложенного для flow shop планирования с минимизацией общей длительности плана или максимального отклонения от заданных директивных сроков, оказывается эффективной и при минимизации среднего времени пребывания задания в системе. Напомним, что это время складывается из двух составляющих — времени ожидания в очереди и времени исполнения.

В настоящей работе предложен новый метод энергоэффективного flow shop планирования для системы со многими информационными выходами. Приведены сведения о flow shop планировании и РКС-алгоритме, а также представлено описание процедуры синтеза энергоэффективной архитектуры распределенной системы. Рассмотрен алгоритм планирования вычислений и приведен пример его применения.

Предварительные сведения

Flow shop планирование. Приведем традиционную постановку задачи flow shop планирования в распределенной вычислительной системе. Система включает множество $C = \{C_i | i = \overline{1, n}\}$ процессоров (ядер), имеющих индивидуальную память для хранения кода программ и обменивающихся информацией через каналы передачи данных. Предположим, что рассматриваемое множество задач разбито на независимые группы задач (задания), связанных отношением предшествования. Планированию подлежат m независимых равноприоритетных заданий $\tau = \{\tau_j | j = \overline{1, m}\}$, обрабатывающих входные данные, которые поступают с периодом T. Каждое j-ое задание состоит из n задач $\tau_{j,i}$ длительностью $e_{j,i}$, $i = \overline{1, n}$. Допустим, что значения длительностей известно точно. С практической точки зрения это означает, что использованы, например, верхние границы длительностей. Все процессоры системы имеют одинаковую производительность. При flow shop планировании примем, что проблема назначения решена [5, 11–14]. В настоящей работе это означает, что имеется m изоморфизмов: φ_j: $\mathbf{G_j}(E_j, T_j) \rightarrow \mathbf{F}(Q, C), j = \overline{1, m}$, где $\mathbf{G_j}(E_j, T_j)$ — граф межзадачных связей j-го задания, E_j — множество ребер, T_j — множество вершин (задач); $\mathbf{F}(Q, C)$ — граф межпроцессорных связей, Q — множество ребер, C — множество процессоров.

Процессор C_i будем называть i-ой стадией системы. Заметим, что в простейшем случае граф $\mathbf{G_j}(E_j, T_j)$ является линейным, т. е. представляет собой цепочку. В этом случае система будет конвейером, а процессор C_i — его стадией.

Все введенные графы являются направленными, ациклическими, содержащими в общем случае не один путь между любыми выделенными вершинами. Графы характеризуются выделенным подмножеством входных вершин и одной выходной вершиной. Передача и прием данных между задачами одного задания, выполняемыми на смежных процессорах, осуществлялись без дополнительных задержек по их готовности. Далее для рассматриваемой системы используем обозначение $S(C, \tau)$.

РКС-алгоритм. Рассмотрим последовательность действий, которые включают в себя РКС-алгоритм, реализующий flow shop планирование. Алгоритм основан на использовании разрешимых классов распределенных flow shop систем. Для их определения введем отношение доминирования на множестве $C = \{C_i | i = \overline{1, n}\}$ процессоров.

Определение 1. Процессор C_q доминирует над процессором $C_r (C_q > C_r)$, если $\min\limits_j e_{j,q} \geq \max\limits_j e_{j,r}, j = \overline{1, m}$.

Разрешимые классы систем имеют следующие отличительные особенности. Первый класс имеет множество процессоров системы убывающей последовательности по отношению доминирования; второй — возрастающую последовательность; третий и четвертый — две соединенные последовательности (возрастающей и убывающей для третьего класса; убывающей и возрастающей для четвертого класса) [7, 8].

Для всех разрешимых классов известны алгоритмы flow shop планирования, которые отличает существенная простота, поскольку в них отсутствует перебор вариантов плана. Их сложность является линейной — $O(m)$.

Очевидно, что на практике для распределенной системы общего вида, описанной в постановке проблемы, жесткие условия ее принадлежности к тому или иному разрешимому классу чаще всего не выполняются. В частности, могут не совпадать критические пути разных заданий, нарушаться отношение доминирования между процессорами, а если оно и выполняется, то может не быть характерных для разрешимых классов упорядоченных по этому отношению цепочек процессоров. В результате исчезают гарантии оптимальности алгоритмов, перечисленные в разделе «Введение», и теряет силу справедливый для разрешимых классов факт независимости качества плана от варианта упорядоченности некрайних заданий. В связи с этим предлагается пусть приближенный, но справедливый для любой из рассматриваемых систем рекурсивный алгоритм планирования (РКС-алгоритм), выполняемый за число шагов, не большее, чем число заданий. На каждом шаге рекурсии определяется некоторый аналог критического пути, называемый псевдокритическим. Далее исполь-

зуется алгоритм планирования, соответствующий тому разрешимому классу, к которому наиболее близка рассматриваемая на данном шаге система $S' = \{C, \tau'\}$, где τ' — множество неразмещенных заданий ($\tau' \subseteq \tau$). При этом выбранные задания занимают обе крайние позиции из интервала свободных позиций формируемого плана или одну из них. После размещения эти задания исключаются из исходного множества и осуществляется переход к следующему шагу рекурсии, реализуемому уже для оставшегося множества заданий на множестве свободных позиций плана. В результате алгоритм последовательно размещает в плане все рассматриваемые задания в направлении от крайних заданий плана к центру.

Постановка проблемы планирования и путь решения

Рассмотрим расширенную проблему flow shop планирования, которую назовем проблемой энергоэффективного flow shop планирования. В отличие от традиционной однокритериальной проблемы flow shop планирования, обсуждаемая проблема становится двухкритериальной. Подход к решению многокритериальных задач, хотя и хорошо известен, но непрост. Однако, исходя из практических соображений о существенной предпочтительности, по мнению авторов, критерия энергоэффективности, в настоящей работе принят подход, при котором данная двухкритериальная задача формулируется как две последовательно решаемые на разных этапах проектирования однокритериальные задачи. В результате на этапе назначения задач изменим архитектуру A системы с одновременным изменением тактовой частоты и напряжения питания процессоров с целью минимизации потребляемой мощности P:

$$A^* = \arg\min_A P(A). \quad (1)$$

Произведенное назначение заданий соответствует случаю flow shop системы. На втором этапе планирования найдем наилучший план π вычислений по критерию минимума среднего по заданиям времени пребывания в системе $\bar{F}(\pi)$:

$$J = \min_\pi \bar{F}(\pi).$$

Отличие рассматриваемой на втором этапе постановки проблемы планирования от традиционной постановки flow shop планирования состоит в том, что, в общем случае система S после преобразования ее архитектуры на первом этапе будет иметь более одного информационного выхода. В этом случае известные алгоритмы flow shop планирования оказываются неработоспособными.

Предлагаемый алгоритм планирования состоит из двух частей. В первой части для каждого информационного выхода системы независимым образом формируется частный план вычислений с использованием модифицированного PKC-алгоритма [6], предназначенного для flow shop планирования. Далее во второй части частные планы интегрируются в общий план вычислений, для чего использован эвристический алгоритм.

Энергоэффективный алгоритм назначения задач на процессоры

Обсудим соотношения, описывающие потребляемую системой мощность P, поскольку именно она будет составлять основу критерия оптимизации (1) на этапе назначения задач. Отметим, что случай с минимизацией энергии вместо мощности аналогичен рассматриваемому ниже, поскольку энергия, потребляемая системой, равна произведению мощности на время работы системы. Известно [8], что мощность имеет две составляющие — динамическую P_d и статическую P_s, которые описываются следующими выражениями:

$$P_d = aNV^2f,\ P_s = bN, \quad (2)$$

где a, b — коэффициенты пропорциональности, зависящие от свойств кристалла; N — число процессоров в системе; V — напряжение питания; f — тактовая частота. Так как вклад статической мощности в суммарную потребляемую мощность невелик, учтем только динамическую составляющую. Для анализа P_d используем приближенную формулу, определяющую задержку, вносимую схемой при напряжении питания V [8]:

$$D = \frac{c}{V}, \quad (3)$$

где c — коэффициент пропорциональности, зависящий от свойств кристалла.

В случае снижения частоты тактовых импульсов в обратной пропорции возрастает их период, ограничивающий допустимое время для переходных процессов, возникающих в системе при каждом срабатывании. При исходном значении напряжения питания фактическое время переходных процессов будет мало по отношению к новому увеличенному значению периода, а значит, возникает возможность пропорционально снизить напряжение питания с увеличением задержки в рамках периода тактовых импульсов в соответствии с выражением (3). В результате выполнения этих двух шагов может быть достигнуто существенное снижение потребляемой мощности (2). Пусть частота и напряжение питания снижены в k раз. Тогда, в соответствии с (2), динамическая мощность снизится в k^3 раз, а время работы ядра увеличится в k раз. Если для сохранения времени работы на прежнем уровне увеличить в k раз число ядер, а вычислительную нагрузку разделить между всеми ядрами поровну, то в результате потребляемая мощность по отношению к исходному варианту уменьшится в k^2 раз. Описанный факт положен в основу предлагаемого подхода к определению архитектуры системы.

Суть подхода состоит в последовательном определении для каждой стадии числа реализующих ее процессоров. Причем предполагается, что все процессоры стадии работают на одной частоте и при одном напряжении питания, а также что для системы известен допустимый исходный вариант значений параметров

f_0, V_0. Безусловно, проектируя систему на кристалле, разработчик всегда ограничен не только значениями потребляемой мощности и площадью s_0 кристалла, отведенной для реализации вычислительной системы, но и величинами напряжения питания и частоты.

В общем случае, когда стадий несколько, возникает вопрос о том, как наилучшим образом с точки зрения минимизации потребляемой мощности распорядиться дополнительными ядрами (запасом по площади) в рамках ограничений. Для ответа на данный вопрос рассмотрим алгоритм определения энергоэффективной архитектуры. Назовем ядра исходной реализации системы «исходными», процесс добавления в i-ую стадию $n_{d,i}$ дополнительных ядер — «расщеплением i-го исходного ядра», а образованное в результате расщепления множество ядер — «расщепленным множеством i-го ядра». Будем считать расщепленное множество i-го ядра предельным, если исключение из него одного ядра делает его среднюю по множеству потребляемую мощность максимальной среди стадий системы. Очевидно, что экономия мощности будет максимальной, если разгружаться будут наиболее загруженные ядра, а распределение нагрузки между ядрами в расширенном множестве будет осуществляться сбалансированным образом, т. е. равномерно. По этому принципу работает предложенный алгоритм. Заметим, что идеальная сбалансированность при распределении нагрузки, разделенной на равные части между ядрами стадии, на практике в общем случае невозможна. В связи с этим речь идет только о приближенной сбалансированности. Архитектуру системы представим вектором состава $\mathbf{A} = (a_1\ a_2\ \dots\ a_n)$, где a_i — число ядер в расщепленном множестве i-го ядра, и вектором средней потребляемой мощности $\bar{\mathbf{P}} = (\bar{P}_1\ \bar{P}_2\ \dots\ \bar{P}_n)$, где $\bar{P}_i$ — средняя по расщепленному множеству i-го ядра потребляемая мощность. Итак, предлагается следующий оптимальный алгоритм назначения задач на процессоры.

Алгоритм 1 (определение энергоэффективной архитектуры).

Шаг 1. Сделать начальные присвоения: $M = n_d$ (допустимое число дополнительных ядер), $\mathbf{A} = (1\ 1\ \dots\ 1)$, $\bar{\mathbf{P}} = (\bar{P}_1\ \bar{P}_2\ \dots\ \bar{P}_n)$.

Шаг 2. Выбрать в $\bar{\mathbf{P}} = (\bar{P}_1\ \bar{P}_2\ \dots\ \bar{P}_n)$ компоненту с максимальным значением $\bar{P}_{\max}$. Пусть ее номер равен l. Ввести дополнительное ядро в l-ую стадию. Произвести между ядрами l-ой стадии приближенно сбалансированное перераспределение нагрузки. Пересчитать параметры алгоритма: $\bar{\mathbf{P}}$, $a_l := a_l + 1$, $M := M - 1$. Если $M \neq 0$, то повторить шаг 2, в противном случае — конец алгоритма.

Алгоритм планирования в системах со многими выходами

Для осуществления второго этапа планирования предложен алгоритм, который состоит из двух частей. Назначение первой части состоит в формировании для каждого выхода системы частного плана. Для этого с каждым i-м выходом системы сопоставляется подсистема, куда включаются процессоры, участвующие в формировании информации на этом выходе. Далее для каждой i-ой системы с использованием РКС-алгоритма составляется частный план. При исполнении данного шага в общем случае возникает трудность, не позволяющая использовать классические алгоритмы flow shop планирования. Трудность состоит в нарушении изоморфизмов графов заданий подсистемы и ее графа процессоров. В [14] предложено решение для преодоления этой трудности. Оно заключается во введении в графы заданий задач нулевой длительности.

Остается эти частные планы интегрировать в общий план. Рассмотрим эвристический алгоритм интегрирования, базовая идея которого заимствована у известного NEH-алгоритма [1].

Алгоритм 2.

Шаг 1. Упорядочить сформированные подсистемы по сложности в порядке убывания числа исполняемых ими заданий.

Шаг 2. Определить для системы подсистему, исполняющую наибольшее число заданий. Считать ее план текущим.

Шаг 3. Построить для системы интегрированный план, последовательно дополняя текущий план заданиями из других частных планов, выбирая эти планы в соответствии с упорядоченностью по сложности. При включении в интегрированный план заданий из очередного частного плана использовать NEH-алгоритм, сохраняя принятую в частном плане упорядоченность.

Результаты

Предложенный метод энергоэффективного планирования применен при проектировании автономных необитаемых подводных аппаратов. Часть результатов этих разработок описаны в работах [13, 15]. Приведем пример простой системы, включающей четыре стадии и для которой уже реализован первый этап планирования.

Пример. Построим план для простой системы, представленной на рисунке. Система содержит 9 процессоров, из которых процессоры 7–9 являются выходными. В структуре системы выделим следующие стадии: первая включает в себя процессоры 1 и 2; вторая — 3 и 4; третья — 5 и 6; четвертая — 7–9.

В системе исполняются четыре задания T_1, T_2, T_3 и T_4. На рис. 1 отмечены процессоры, на которых исполняются входящие в них задачи. Выполнение каждого из заданий разделим на четыре стадии, длительности стадий приведены в табл. 1.

Таблица 1. Длительности (τ_i) выполнения стадий заданий
Table 1. Durations of job implementation stages

Задание	Длительность (τ_i) выполнения стадии задания, отн. ед.			
	$\tau_{i,1}$	$\tau_{i,2}$	$\tau_{i,3}$	$\tau_{i,4}$
T_1	1	3	2	3
T_2	2	1	3	2
T_3	2	3	1,5	5
T_4	1	1	2	4

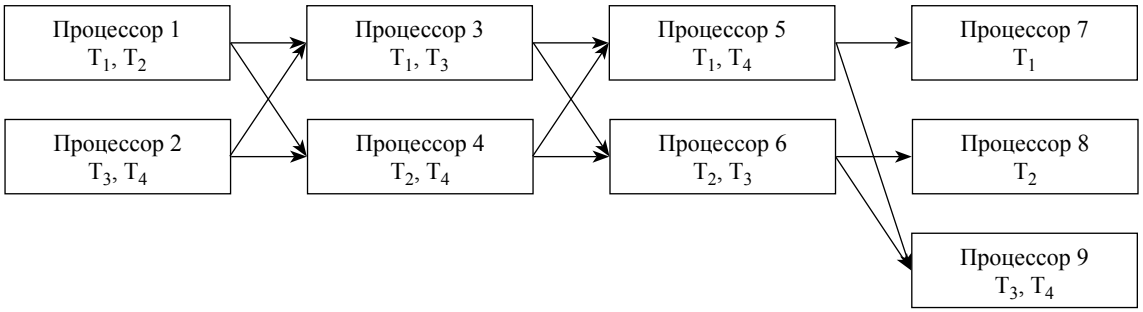

Рисунок. Система с тремя информационными выходами
Figure. System with three information outputs

Для каждого информационного выхода выделена в системе своя подсистема. Подсистема S1 включает процессоры 1, 3, 5, 7; S2 — 1, 4, 6, 8; S3 — 2–6, 9. В подсистеме S3 исполнены два задания — T_3 и T_4. Следовательно, необходимо осуществить их предварительное упорядочивание. Построим план для подсистемы S3, используя РКС-алгоритм.

Предварительно преобразуем описание системы к виду, допускающему применение алгоритмов flow shop планирования (табл. 2) путем введения задач нулевой длительности. Таким образом, можно считать, что задачи каждого задания выполнены на всех процессорах, т. е. структура исполняющей вычислительной системы для каждого задания одинакова, что и требуется при flow shop планировании. В этом случае число стадий выполнения каждого задания соответствует количеству процессоров в системе.

Выделим в подсистеме S3 псевдокритический путь на основе медианных значений длительностей решаемых на каждом процессоре задач, приведенных в табл. 2. В подсистеме существует два вычислительных пути с соответствующими длительностями:

$$\bar{e}(p_{2,3,6,9}) = 0{,}5 + 1{,}5 + 0{,}75 + 2 = 4{,}75;$$
$$\bar{e}(p_{2,4,5,9}) = 0{,}5 + 0{,}5 + 1 + 2 = 4.$$

Получим псевдокритический путь, состоящий из процессоров 2, 3, 6, 9, длительностью $\bar{e}(p_{2,3,6,9}) = 4{,}75$. Выполним анализ полученного пути с привлечением геометрического правила [8]. В результате получим, что подсистема S3 наиболее близка к системам из второго разрешимого класса. Построим план в соответствии с алгоритмом планирования для второго класса, согласно которому задание T_4 должно предшествовать заданию T_3. Таким образом, текущий план имеет вид (T_4, T_3).

Для завершения планирования определим в интегрированном плане позиции для заданий T_1 и T_2. В соответствии с предлагаемым алгоритмом это достигается применением базовой идеи NEH-алгоритма, когда задания встраиваются в формируемый план в результате использования ограниченного перебора вариантов. В частный план, содержащий предварительно упорядоченные задания τ_4 и τ_3 поочередно встраиваются задания T_1 и T_2, причем место размещения в плане определено по минимальному среднему времени пребывания заданий в системе. Например, задание τ_1 может быть размещено в плане следующим образом: (T_4, T_3, T_1) или (T_4, T_1, T_3), или (T_1, T_4, T_3), при этом значения среднего времени пребывания заданий в системе составляют 12,6, 9,8 и 11,3 отн. ед. соответственно. В результате наименьшему значению критерия соответствует план (T_4, T_1, T_3) — данный план выберем для дальнейшей работы.

На следующем этапе задание T_2 разместим в выбранном плане в сочетаниях: (T_4, T_1, T_3, T_2), (T_4, T_1, T_2, T_3), (T_4, T_2, T_1, T_3), (T_2, T_4, T_1, T_3), при этом получены значения среднего времени пребывания заданий в системе — 10,5, 9,38, 9,88, 10,38 отн. ед. соответственно. На данном этапе выберем план (T_4, T_1, T_2, T_3).

В результате сравнения путем полного перебора определен оптимальный план, который совпал с ранее полученным по предложенному алгоритму.

Таблица 2. Длительности (τ_i) выполнения стадий заданий на каждом процессоре
Table 2. Duration of task implementation stages on each processor

Задание	Длительность (τ_i) выполнения стадии задания, отн. ед.								
	$\tau_{i,1}$	$\tau_{i,2}$	$\tau_{i,3}$	$\tau_{i,4}$	$\tau_{i,5}$	$\tau_{i,6}$	$\tau_{i,7}$	$\tau_{i,8}$	$\tau_{i,9}$
T_1	1	0	3	0	2	0	3	0	0
T_2	2	0	0	1	0	3	0	2	0
T_3	0	2	3	0	0	1,5	0	0	5
T_4	0	1	0	1	2	0	0	0	4
$\bar{\tau}$ (медианы)	0,5	0,5	1,5	0,5	1	0,75	0	0	2

Заключение

В работе исследованы вопросы организации вычислений в распределенной вычислительной системе реального времени. Предложен двухэтапный алгоритм энергоэффективного flow shop планирования с учетом двух критериев — минимума потребляемой мощности и минимума среднего времени пребывания задания в системе. Алгоритм отличается от известных алгоритмов flow shop планирования тем, что позволяет осуществить планирование в системе с множественными выходами в то время, как известные алгоритмы в этих условиях не применимы.

Отметим, что предложенный алгоритм обладает полиномиальной сложностью, поэтому благодаря относительной простоте алгоритма возможно его применение при планировании и перепланировании в реальном времени сложных систем.

Литература

1. Nawaz M., Enscore Jr. E.E., Ham I. A heuristic algorithm for the m-Machine, n-Job flow-shop sequencing problem // Omega – International Journal of Management Science. 1983. N 11. N 1. P. 91–95. https://doi.org/10.1016/0305-0483(83)90088-9
2. Лазарев А.А., Гафаров Е.Р. Теория расписаний. Задачи и алгоритмы. М.: МГУ, 2011. 222 с.
3. 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.
4. Liu J.W.S. Real-Time Systems. Englewood Cliffs, NJ: Prentice Hall, 2000. 600 p.
5. Cottet F., Delacroix J., Kaiser J., Mammeri Z. Scheduling in Real-Time Systems. John Wiley & Sons, 2002. 282 p.
6. Колесов Н.В., Толмачева М.В., Юхта П.В. Системы реального времени. Планирование, анализ, диагностирование. СПб.: ОАО «Концерн «ЦНИИ «Электроприбор», 2014. 185 с.
7. Gruzlikov A.M., Kolesov N.V., Skorodumov Iu.M., Tolmacheva M.V. Using solvable classes in flowshop scheduling // The International Journal of Advanced Manufacturing Technology. 2017. V. 88. N 5-8. P. 1535–1546. https://doi.org/10.1007/s00170-016-8828-5
8. Грузликов А.М., Колесов Н.В., Скородумов Ю.М., Толмачева М.В. Планирование заданий в распределенных системах реального времени // Известия РАН. Теория и системы управления. 2017. № 2. С. 67–76. https://doi.org/10.7868/S000233881702010X
9. Panda P.R., Silpa B.V.N., Shrivastava A., Gummidipudi K. Power-efficient System Design. Springer, New York, 2010. 260 p. https://doi.org/10.1007/978-1-4419-6388-8
10. Xu R., Melhem R., Moss D. Energy-aware scheduling for streaming applications on chip multiprocessors // Proc. of the 28th International Real-Time Systems Symposium (RTSS). 2007. P. 25–38. https://doi.org/10.1109/rtss.2007.49
11. Sun H., He Y., Hsu W.-J., Fan R. Energy-efficient multiprocessor scheduling for flow time and makespan // Theoretical Computer Science. 2014. V. 550. P. 1–20. https://doi.org/10.1016/j.tcs.2014.07.007
12. Грузликов А.М., Колесов Н.В., Костыгов Д.В., Ошуев В.В. Энергоэффективное планирование в распределенных вычислительных системах реального времени // Известия РАН. Теория и системы управления. 2019. № 3. С. 66–76. https://doi.org/10.1134/S0002338819030090
13. 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
14. Колесов Н.В., Грузликов А.М., Скородумов Ю.М., Толмачева М.В. Смешанное планирование заданий в распределенных системах реального времени // Вестник компьютерных и информационных технологий. 2016. № 5. С. 34–40. https://doi.org/10.14489/vkit.2016.05.pp.034-040
15. Грузликов А.М., Колесов Н.В., Литуненко Е.Г., Скородумов Ю.М. Оптимизация информационных обменов в сети автономных абонентов // Известия РАН. Теория и системы управления. 2022. № 6. С. 56–64. https://doi.org/10.31857/s0002338822060105

References

1. Nawaz M., Enscore Jr. E.E., Ham I. A Heuristic algorithm for the m-Machine, n-Job flow-shop sequencing problem. *Omega — International Journal of Management Science*, 1983, no. 11, no. 1, pp. 91–95. https://doi.org/10.1016/0305-0483(83)90088-9
2. Lazarev A.A., Gafarov E.R. *Scheduling Theory. Problems and Algorithms*. Moscow, MSU, 2011, 222 p. (in Russian)
3. Malashenko Yu.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
4. Liu J.W.S. *Real-Time Systems*. Englewood Cliffs, NJ, Prentice Hall, 2000, 600 p.
5. Cottet F., Delacroix J., Kaiser J., Mammeri Z. *Scheduling in Real-Time Systems*. John Wiley & Sons, 2002, 282 p.
6. Kolesov N.V., Tolmacheva M.V., Yukhta P.V. *Real Time Systems. Planning, Analysis, Diagnosis*. St. Petersburg, Concern CSRI Elektropribor, 2014, 185 p. (in Russian)
7. Gruzlikov A.M., Kolesov N.V., Skorodumov Iu.M., Tolmacheva M.V. Using solvable classes in flowshop scheduling. *The International Journal of Advanced Manufacturing Technology*, 2017, vol. 88, no. 5-8, pp. 1535–1546. https://doi.org/10.1007/s00170-016-8828-5
8. Gruzlikov A.M., Kolesov N.V., Skorodumov Y.M., Tolmacheva M.V. Task scheduling in distributed real-time systems. *Journal of Computer and Systems Sciences International*, 2017, vol. 56, no. 2, pp. 236–244. https://doi.org/10.1134/s1064230717020101
9. Panda P.R., Silpa B.V.N., Shrivastava A., Gummidipudi K. *Power-efficient System Design*. Springer, New York, 2010, 260 p. https://doi.org/10.1007/978-1-4419-6388-8
10. Xu R., Melhem R., Moss D. Energy-aware scheduling for streaming applications on chip multiprocessors. *Proc. of the 28th International Real-Time Systems Symposium (RTSS)*, 2007, pp. 25–38. https://doi.org/10.1109/rtss.2007.49
11. Sun H., He Y., Hsu W.-J., Fan R. Energy-efficient multiprocessor scheduling for flow time and makespan. *Theoretical Computer Science*, 2014, vol. 550, pp. 1–20. https://doi.org/10.1016/j.tcs.2014.07.007
12. Gruzlikov A.M., Kolesov N.V., Kostygov D.V., Oshuev V.V. Energy-efficient scheduling in distributed real-time computing systems. *Journal of Computer and Systems Sciences International*, 2019, vol. 58, no. 3, pp. 393–403. https://doi.org/10.1134/s1064230719030092
13. 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
14. Kolesov N.V., Gruzlikov A.M., Skorodumov Yu.M., Tolmacheva M.V. Mixed job scheduling in distributed real time systems. *Vestnik Komp'iuternykh i Informatsionnykh Tekhnologi*, 2016, no. 5, pp. 34–40. (in Russian). https://doi.org/10.14489/vkit.2016.05.pp.034-040
15. Gruzlikov A.M., Kolesov N.V., Litunenko E.G., Skorodumov Yu.M. Optimization of information exchange in a network of autonomous participants. *Journal of Computer and Systems Sciences International*, 2022, vol. 61, no. 6, pp. 935–943. https://doi.org/10.1134/S1064230722060107

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 — инженер, аспирант, АО «Концерн «ЦНИИ «Электроприбор», Санкт-Петербург, 197046, Российская Федерация, sc 57796319900, https://orcid.org/0000-0001-5280-0593, lisa.litunenko@gmail.com

Скородумов Юрий Михайлович — кандидат технических наук, начальник лаборатории, АО «Концерн «ЦНИИ «Электроприбор», Санкт-Петербург, 197046, Российская Федерация, sc 56413038700, https://orcid.org/0000-0003-0825-1720, skorum@mail.ru

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

Authors

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

Elizaveta G. Litunenko — PhD Student, Engineer, Concern CSRI Elektropribor, JSC, Saint Petersburg, 197046, Russian Federation, sc 57796319900, https://orcid.org/0000-0001-5280-0593, lisa.litunenko@gmail.com

Iurii M. Skorodumov — PhD, Head of Laboratory, Concern CSRI Elektropribor, JSC, Saint Petersburg, 197046, Russian Federation, sc 56413038700, https://orcid.org/0000-0003-0825-1720, skorum@mail.ru

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

Статья поступила в редакцию 08.06.2023
Одобрена после рецензирования 31.08.2023
Принята к печати 30.09.2023

Received 08.06.2023
Approved after reviewing 31.08.2023
Accepted 30.09.2023