

# ЭНЕРГОЭФФЕКТИВНОЕ ПЛАНИРОВАНИЕ В РАСПРЕДЕЛЕННЫХ СИСТЕМАХ<sup>1</sup>

**А.М. Грузликов**

*АО «Концерн «ЦНИИ «Электроприбор»*  
Россия, 197046, Санкт-Петербург, ул. М. Посадская, д. 30.  
E-mail: [agruzlikov@yandex.ru](mailto:agruzlikov@yandex.ru)

**Н.В. Колесов**

*АО «Концерн «ЦНИИ «Электроприбор»*  
Россия, 197046, Санкт-Петербург, ул. М. Посадская, д. 30.  
E-mail: [kolesovnv@mail.ru](mailto:kolesovnv@mail.ru)

**В.В. Ошув**

*АО «Концерн «ЦНИИ «Электроприбор»*  
Россия, 197046, Санкт-Петербург, ул. М. Посадская, д. 30.  
E-mail: [oshuev.valery@yandex.ru](mailto:oshuev.valery@yandex.ru)

**Д.В. Костыгов**

*АО «Концерн «ЦНИИ «Электроприбор»*  
Россия, 197046, Санкт-Петербург, ул. М. Посадская, д. 30.  
E-mail: [dkost92@mail.ru](mailto:dkost92@mail.ru)

**Ключевые слова:** планирование, вычислительные системы реального времени, энергоэффективность.

**Аннотация:** Рассматривается подход к планированию вычислительного процесса в распределенных системах реального времени, в котором предусмотрен этап определения архитектуры системы с целью минимизации потребляемой системой мощности. Проблема формулируется как планирование заданий, каждое из которых состоит из ряда задач (по числу процессоров), исполняемых на разных процессорах и связанных отношением предшествования. В основе подхода лежат алгоритмы с низкой вычислительной сложностью для субоптимального планирования равноприоритетных заданий.

## 1. Введение

Проблема планирования каких-либо действий возникает в разных приложениях и в разнообразных постановках. В настоящей работе обсуждается планирование в бортовых системах [1-5], а именно, планирование вычислений в распределенных системах реального времени (СРВ). Если быть более точным, то в фокусе настоящей работы лежит планирование распределенных вычислений в многоядерной системе на кристалле. При этом если на одном ядре оказываются две или более задач, не связанных отношением предшествования, попутно возникает вопрос об очередности их выполнения, которая определяется наилучшим образом с точки зрения заданного критерия. Таковыми могут быть, 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

---

<sup>1</sup> Работа поддержана Российским фондом фундаментальных исследований (проект № 19-08-00052).

было снижение энергопотребления. В последние десятилетия ей уделяется особое внимание, в том числе и в связи с обсуждением ее в отношении систем на кристалле [4-7], когда снижение энергопотребления (рассеиваемой мощности) достигается за счет снижения тактовой частоты и напряжения питания. Ниже минимум рассеиваемой мощности будет одним из критериев, на основании которых будут планироваться вычисления.

Далее будем рассматривать широко обсуждаемую в литературе проблему (flow shop)-планирования [7-10], в качестве практической интерпретации которой часто называют обработку информации в многоканальных системах. В предлагаемом алгоритме планирования на первом этапе будет использоваться критерий минимума рассеиваемой мощности, а на втором – минимум общего времени выполнения плана.

## 2. Постановка задачи

Рассмотрим постановку задачи (flow shop)-планирования в распределенной вычислительной системе. Система включает множество  $C = \{C_i \mid i = \overline{1, n}\}$  процессоров (ядер). Предполагается, что рассматриваемое множество задач разбито на независимые группы задач (далее задания), связанных отношением предшествования. Планированию подлежат  $m$  независимых равноприоритетных заданий  $\tau = \{\tau_j \mid j = \overline{1, m}\}$ , обрабатывающих входные данные, которые поступают с периодом  $T$ . Каждое  $j$ -е задание состоит из  $n$  задач  $\tau_{j,i}$  длительностью  $e_{j,i}$   $i = \overline{1, n}$ . Будем предполагать, что значения длительностей известно точно. Все процессоры системы имеют одинаковую производительность. При (flow shop)-планировании предполагается, что проблема назначения уже решена. В данной работе это означает, что имеется  $m$  изоморфизмов  $\varphi_j : G_j(E_j, T_j) \rightarrow F(Q, C)$   $j = \overline{1, m}$ , где  $G_j(E_j, T_j)$  – граф межзадачных связей  $j$ -го задания,  $E_j$  – множество ребер,  $T_j$  – множество вершин (задач),  $F(Q, C)$  – граф межпроцессорных связей,  $Q$  – множество ребер,  $C$  – множество процессоров. Процессор  $C_i$  будем называть  $i$ -й стадией системы. Заметим, что в простейшем случае граф  $G_j(E_j, T_j)$  является линейным, т.е. представляет собой цепочку. В этом случае система является конвейером, а процессор  $C_i$  будет его стадией. Далее для рассматриваемой системы будем использовать обозначение  $S(C, \tau)$ .

В настоящей работе рассматривается расширенная проблема (flow shop)-планирования, которую назовем проблемой энергоэффективного (flow shop)-планирования. Ее особенность состоит в том, что, если традиционно задачи любой  $i$ -й стадии решаются на одном процессоре, то в предлагаемом подходе число процессоров  $i$ -й стадии определяется, исходя из соображений энергоэффективности, и может принимать значения от 1 до  $m$ . Таким образом, на этапе назначения задач будет варьироваться архитектура  $A$  системы с одновременным изменением тактовой частоты и напряжения питания процессоров с целью минимизации потребляемой мощности  $P$ :

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

## 3. Предварительные сведения

Основой для предлагаемых решений являются известные эффективные алгоритмы (flow shop)-планирования, а именно, NEH-алгоритм [9] и разработанный авторами РКС-алгоритм [10]. Коротко опишем эти алгоритмы.

*NEH-алгоритм* (Nawaz, Enscore, Ham). Алгоритм основывается на ограниченном переборе возможных вариантов упорядочивания заданий. Суть его заключается в следующем. Предварительно список планируемых заданий упорядочивается, например, по возрастанию их длительностей. Далее в формируемом плане на первой позиции размещается первое задание из упорядоченного списка. Далее второе задание из этого списка размещается в плане сначала на первой позиции, а затем на второй. Для обоих вариантов вычисляется значение критерия (общее время выполнения плана), а затем принимается вариант с минимальным значением критерия. На следующем шаге действия аналогичны.

*РКС-алгоритм*. Следующий из рассматриваемых алгоритмов основан на использовании разрешимых классов распределенных (flow shop)-систем. Для всех разрешимых классов известны [10] оптимальные алгоритмы (flow shop)-планирования. Эти алгоритмы отличает существенная простота, поскольку в них отсутствует перебор вариантов плана. Очевидно, что на практике для распределенной системы общего вида жесткие условия ее принадлежности к тому или иному разрешимому классу чаще всего не выполняются. В результате исчезают гарантии оптимальности упомянутых выше алгоритмов. В связи с этим предложен пусть приближенный, но справедливый для любой из рассматриваемых систем рекурсивный алгоритм планирования (алгоритм разрешимых классов систем – РКС-алгоритм), выполняемый за число шагов, не большее, чем число заданий. На каждом шаге рекурсии используется алгоритм планирования, соответствующий тому разрешимому классу, к которому наиболее близка рассматриваемая на данном шаге система  $S' = \{C, \tau'\}$ , где  $\tau'$  – множество неразмещенных заданий ( $\tau' \subseteq \tau$ ). 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.

#### 4. 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. Сразу отметим, что случай с минимизацией энергии вместо мощности аналогичен рассматриваемому ниже, поскольку энергия, потребляемая системой, равна произведению мощности на время работы системы.

Известно, что при снижении частоты и напряжения питания может быть достигнуто существенное снижение потребляемой мощности [4-6]. Так если как частота, так и напряжение питания снижаются в  $k$  раз, то потребляемая мощность снижается в  $k^3$  раз. При этом, правда, и время работы ядра увеличивается в  $k$  раз. Если для сохранения его 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  $k^2$  раз. Описанный факт положен в основу алгоритма определения архитектуры системы.

Суть алгоритма 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 по потребляемой мощности, но и по площади  $s_0$  кристалла, отведенной для реализации вычисли-

тельной системы, по напряжению питания и по частоте. Зная размер площади  $s_k$ , занимаемой одним ядром, можно пересчитать ограничение по площади кристалла в допустимое число  $n_d$  дополнительных ядер.

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 существующих ограничений. Очевидно, что экономия мощности будет максимальной, если разгружаться будут наиболее загруженные ядра, а распределение нагрузки в расширенном множестве ядер будет осуществляться сбалансированным образом. Будем называть ядра исходной реализации системы «исходными», процесс добавления в  $i$ -ю стадию  $n_{d,i}$  дополнительных ядер «расщеплением  $i$ -го исходного ядра», а образованное в результате расщепления  $i$ -го исходного ядра множество ядер – «расщепленным множеством  $i$ -го исходного ядра». Архитектуру системы будем представлять вектором состава  $A = (a_1 \ a_2 \ \dots \ a_n)$ , где  $a_i$  – число ядер в расщепленном множестве  $i$ -го исходного ядра, и вектором средней потребляемой мощности  $\bar{P} = (\bar{P}_1 \ \bar{P}_2 \ \dots \ \bar{P}_n)$ , где  $\bar{P}_i$  – средняя потребляемая мощность в расщепленном множестве  $i$ -го исходного ядра.

Итак, предлагается следующий алгоритм.

*Алгоритм определения энергоэффективной архитектуры*

Шаг 1. Сделать начальные присвоения:  $M = n_d$  (число неиспользованных дополнительных ядер),  $A = (1 \ 1 \ \dots \ 1)$ ,  $\bar{P} = (\bar{P}_1 \ \bar{P}_2 \ \dots \ \bar{P}_n)$ , где  $\bar{P}_i$  – средняя потребляемая мощность в  $i$ -м исходном ядре.

Шаг 2. Выбрать в  $\bar{P} = (\bar{P}_1 \ \bar{P}_2 \ \dots \ \bar{P}_n)$  компоненту с максимальным значением  $\bar{P}_{\max}$ . Пусть ее номер равен  $l$ . Ввести такое число  $n_{d,l} \leq M$  дополнительных ядер в расщепленное множество с максимальной средней потребляемой мощностью, чтобы после сбалансированного распределения нагрузки исходного  $l$ -го расщепленного множества между его ядрами и всеми дополнительными ядрами для нового значения  $\bar{P}_l^+$  выполнялось  $\bar{P}_l^+ < \bar{P}_{\max}^+$ , где  $\bar{P}_{\max}^+$  – максимальная по системе средняя потребляемая мощность в одном из ее расщепленных множеств. Пересчитать вектора  $A$  и  $P$ . Если  $M - n_{d,l} = 0$ , то конец, иначе перейти к шагу 2.

Поскольку приведенный алгоритм оперирует не с абсолютными значениями мощностей, а с их соотношением, то с инженерной точки зрения возможен переход к оперированию не потребляемыми в стадиях мощностями, а их вычислительными сложностями. Подчеркнем, что выбор варианта размещения задач стадии целесообразно делать в пользу сбалансированного назначения, поскольку в этом случае вычислительная сложность максимального блока будет минимальной, а, значит, и минимальной будет выбираемая тактовая частота стадии.

## 5. Алгоритм (flow shop)-планирования

В докладе предлагается алгоритм (flow shop)-планирования, названный авторами «гибридным», что объясняется использованием комбинации РКС и НЕН алгоритмов. При этом предлагается процедура, порождающая алгоритм планирования с заданными характеристиками производительности и качества планирования в смысле принятого критерия. В качестве критерия оптимальности при этом будем рассматривать минимум общего времени выполнения плана.

Используемые здесь алгоритмы в каком-то смысле составляют альтернативу друг другу. Действительно, РКС-алгоритм характеризуется большей производительностью, но и более низким качеством поставляемого плана, нежели НЕН-алгоритм, и соответственно наоборот. Предлагается при построении плана использовать оба эти алгоритма в рамках следующей двухэтапной процедуры.

*Алгоритм планирования (гибридный).*

Шаг 1. Упорядочить множество заданий по убыванию времени выполнения.

Шаг 2. Разбить все множество заданий на  $k$  групп и при помощи РКС-алгоритма составим частные расписания для каждой из групп.

Шаг 3. Составить общий план для всех заданий при помощи НЕН-алгоритма, последовательно размещая очередное задание из каждой групп в общем плане, не нарушая их последовательность относительно частных расписаний.

Варьируя числом групп, можно варьировать характеристики получаемого алгоритма планирования. Причем при числе групп, равном мощности исходного множества, получаем НЕН-алгоритм, а при числе групп, равном единице, – РКС-алгоритм. Во всех промежуточных ситуациях порождаются алгоритмы планирования с промежуточными значениями характеристик.

В докладе приводятся результаты моделирования задачи планирования при использовании предложенного подхода. При этом применялась случайная генерация, как графов заданий, так и длительностей составляющих их задач. Было показано, что меняя число групп, можно варьировать характеристики получаемого алгоритма планирования, а именно, время работы алгоритма и близость получаемого результата к оптимальному.

## 6. Заключение

В докладе предлагается и исследуется алгоритм энергоэффективного (flow shop)-планирования, который отличают две особенности. Во-первых, введен этап определения архитектуры системы, с целью минимизации потребляемой системой мощности. Во-вторых, предлагается для получаемой архитектуры использовать алгоритм планирования, основанный на комбинации двух известных и наиболее эффективных алгоритмов (flow shop)-планирования – НЕН-алгоритме и РКС-алгоритме.

## Список литературы

1. Liu J.W.S. Real-Time Systems. Englewood Cliffs, NJ: Prentice Hall, 2000. 600 p.
2. Drozdowski M. Scheduling for Parallel Processing. London: Springer, 2009.
3. Колесов Н.В., Толмачева М.В., Юхта П.В. Системы реального времени. Планирование, анализ, диагностирование СПб.: ОАО «Концерн «ЦНИИ «Электроприбор», 2014. 185 с.
4. Xu R., Melhem R., Moss D. Energy-Aware Scheduling for Streaming Applications on Chip Multiprocessors // 28th IEEE International Real-Time Systems Symposium. 2007. P. 25-38.
5. Panda P.R., Shrivastava A., Silpa B.V.N., Gummidipudi K.. Power-efficient System Design. New York: Springer, 2010. 260 p.
6. Sun H., He Y., Hsu W.-J. Energy-Efficient Multiprocessor Scheduling for Flow Time and Makespan // Theoretical Computer Science. 2014. Vo. 550. P. 1-20.
7. Rudek A., Rudek R. On Flowshop Scheduling Problems with the Aging Effect and Resource Allocation // Int. J. Adv. Manuf. Technol. 2012. Vol. 62, No. 1. P. 135-145.
8. Bocewicz G., Banaszak Z.A. Declarative Approach to Cyclic Steady State Space Refinement: Periodic Process Scheduling // Int. J. Adv. Manuf. Technol. 2013. Vol. 67, No. 1-4. P. 137-155.
9. Nawaz M, Ensore Jr. E.E., Ham I. A Heuristic Algorithm for the m-Machine, n-Job Flow-shop Sequencing Problem // Omega – Intern. J. of Management Science. 1983. Vol. 11. No. 1. P. 91-95.
10. Gruzlikov A. M., Kolesov N. V., Skorodumov Iu. M., Tolmacheva M. V. Using solvable classes in flowshop scheduling // Int J Adv Manuf Technol. 2017. Vol. 88. P. 1535-1546.