На правах рукописи

Красовский Дмитрий Владимирович

АЛГОРИТМЫ РЕШЕНИЯ ЗАДАЧИ СОСТАВЛЕНИЯ ОПТИМАЛЬНОГО РАСПИСАНИЯ БЕЗ ПРЕРЫВАНИЙ

Специальность 05 13 18 – математическое моделирование, численные методы и комплексы программ

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата физико-математических наук

Москва – 2007

Работа выполнена на кафедре математических основ управления Московского физико-технического института (государственного университета)

Научный руководитель
кандидат физико-математических наук, доцент
Фуругян Меран Габибуллаевич

Официальные оппоненты
доктор физико-математических наук
Андрианов Александр Николаевич
доктор технических наук, профессор
Сигал Израиль Хаимович

Ведущая организация
Научно-исследовательский Институт Системных Исследований Российской Академии Наук

Защита состоится "25" мая 2007 г в 14 30 час
на заседании диссертационного совета К 212 156 02 при Московском физико-техническом институте (государственном университете) по адресу 141700, Московская обл , г Долгопрудный, Институтский пер , 9, 903 КПМ

С диссертацией можно ознакомиться в библиотеке МФТИ(ГУ)

Автореферат разослан "23" апреля 2007 г

Ученый секретарь
диссертационного совета
к.ф -м н. О С Федько

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность и характеристики задачи

Задачи составления расписания возникают при проектировании и эксплуатации автоматизированных систем обработки информации и управления Такие автоматизированные системы функционируют при непосредственном взаимодействии с внешней средой, и должны за кратчайшее время вернуть среде результаты обработки в виде корректирующих воздействий или в виде сообщений пользователю Необходимость в корректных и полных математических моделях и быстрых точных алгоритмах, составляющих многопроцессорные расписания, часто возникает в задачах распределенных вычислений в реальном времени и задачах оперативного управления на основе обработки поступающих в реальном времени данных

Для иллюстрации большой практической значимости рассматриваемых задач и важности эффективных алгоритмов их решения можно привести следующие примеры системы противоракетной обороны, системы управления ядерными реакторами на АЭС, бортовые системы при испытаниях и управлении летательными аппаратами и космическими кораблями, организация расписаний в современных аэропортах и многие другие

В наиболее общей формулировке задача составления расписания состоит в следующем С помощью некоторого множества ресурсов или обслуживающих устройств должна быть выполнена некоторая фиксированная система заданий Цель заключается в том, чтобы при заданных свойствах заданий и ресурсов и наложенных на них ограничениях найти эффективный алгоритм упорядочения заданий, оптимизирующий желаемую меру эффективности В качестве мер эффективности обычно рассматриваются *длина* расписания и среднее время пребывания заданий в системе Модели этих задач являются детерминированными в том смысле, что вся информация, на основе которой принимаются решения об упорядочении, известна заранее

При построении расписания работы детерминированной системы обслуживания актуальна задача понижения размерности, состоящая в преобразовании исходной системы путем группирования требований и назначения укрупненным требованиям всех тех характеристик, которые содержала исходная система обслуживания Такое преобразование системы называется *агрегирование*

Помимо того, что понижение размерности системы обслуживания ускоряет в дальнейшем процедуру построения допустимого расписания ее работы, оно необходимо и для решения других проблем В современных вычислительных системах со стороны операционной системы (ОС) нередко выдвигаются количественные ограничения, которые необходимо учитывать при подготовке пакета прикладных программ

Ограничениями ОС являются количество приоритетных уровней доступных диспетчеру ОС, уровень мультипрограммирования, объемы

оперативной памяти, выделяемые программной единице и т п Агрегирование детерминированной системы обслуживания позволяет снизить количество управляемых программных единиц, тем самым выполнить соответствующие требования ОС

Понижение размерности системы приводит также к уменьшению накладных расходов на организацию обслуживания, которые, например, согласно некоторым исследованиям в вычислительных системах реального времени достигают 70% (в отдельных случаях и выше) времени работы центрального процессора.

Кроме методов агрегирования, в последнее время все большую популярность стали получать методы параллельного составления расписаний, когда одни параллельные вычислительные системы используются для составления расписания заданий для других параллельных систем На данный момент существует достаточно большое количество различных архитектур параллельных вычислительных систем, и каждая из них требует специально адаптированных алгоритмов для эффективного использования В данной работе разработан новый алгоритм для одной из параллельных архитектур – параллельный вычислительный кластер

При исследовании задачи мы полагаем, что на процесс составления расписания влияют ограничивающие факторы возможностей вычислительной системы, на которой осуществляется упорядочение Основными ограничивающими факторами являются время выполнения процесса составления расписания и объем используемой вычислительной памяти Будем называть задачами большой размерности те, решение которых нарушает хотя бы одно из ограничений используемой вычислительной системы

Цели и новизна работы

Целями настоящей диссертационной работы являются

- построение математических моделей многопроцессорных систем без дополнительных ограничений по ресурсам, выполнение заданий в которых осуществляется без прерываний,
- разработка и анализ точных и эвристических алгоритмов решения задачи поиска оптимального расписания в этих системах,
- разработка нового подхода к решению задачи составления расписания путем агрегирования заданий системы и решения полученной таким образом задачи меньшей размерности,
- разработка параллельных алгоритмов решения задачи составления расписания большой размерности с высоким коэффициентом эффективности и линейной функцией изоэффективности,
- получение аналитических ограничений на точность расписания, получаемого эвристическими алгоритмами, и на время выполнения алгоритмов, разработанных в рамках данной работы,

- подтверждение полученных результатов экспериментальными данными и создание банка вычислительных результатов, которые могут быть в дальнейшем использованы другими исследователями для сравнения практической эффективности новых алгоритмов

Оценки алгоритмической сложности для многих задач рассматриваемого класса известны уже давно, однако, в литературе встречается сравнительно мало специализированных алгоритмов решения этих задач и экспериментальных результатов их работы. В связи с этим, наравне с созданием новых алгоритмов, в рамках работы проведено большое число вычислительных экспериментов и, для сравнения, приведены результаты переборного алгоритма и широко используемых для других классов задач жадных методик и алгоритма имитации отжига.

Методы исследований

В данной диссертационной работе используются методы теории расписаний, теории графов, оптимизации, дискретной математики, теории алгоритмов и теории сложности, а также математического моделирования.

В основу работы легли аналитические исследования. Для каждого из утверждений приведены доказательства. Также, для подтверждения выводов, были разработаны комплексы вычислительных программ, проведено большое количество экспериментов, результаты усреднены для получения статистически достоверных данных.

Практическая ценность работы

Разработанные в диссертации алгоритмы построения оптимальных и ε-приближенных расписаний могут быть рекомендованы к использованию при проектировании, анализе и эксплуатации аппаратно-программных комплексов систем реального времени, применяемых при разработке, испытаниях и эксплуатации сложных технических объектов, а также в системах оперативного мониторинга

Апробация работы

Результаты диссертации и материалы исследований докладывались, обсуждались и получили одобрение специалистов на

- XLV, XLVII и IL научных конференциях Московского физико-технического института (государственного университета), (Долгопрудный, 2002, 2004, 2006),
- X, XI и XII международных конференциях "Проблемы управления безопасностью сложных систем" (Москва, 2002, 2003, 2004),
- IV международной конференции по исследованию операций (Москва, 2004),
- II Всероссийской научной конференции «Методы и средства обработки информации» (Москва, 2005);

- научных семинарах кафедры математических основ управления Московского физико-технического института (государственного университета),
- научных семинарах сектора проектирования систем реального времени Вычислительного центра им А А Дородницына Российской Академии Наук

Публикации

По материалам диссертации опубликовано 12 печатных работ, в том числе две – в ведущих научных журналах, рекомендованных ВАК РФ Список работ приведен в конце автореферата

Структура и объем работы

Диссертация состоит из введения, семи глав, заключения и списка литературы Общий объем работы составляет 109 страниц

СОДЕРЖАНИЕ РАБОТЫ

Во введении формулируется тема диссертации, обосновывается ее актуальность, формулируется ее цель, научная новизна, полученные результаты и структура диссертации

В первой главе приводится постановка задачи Модель процесса упорядочения, в терминах которой формулируются все последующие задачи, представляется в виде совокупности моделей, описывающих ресурсы и систему заданий

Ресурсы

В рассматриваемой модели ресурсы состоят из набора процессоров $P=\{P_1, \ldots, P_m\}$ В зависимости от особенностей задачи они являются либо идентичными, либо одинаковыми только по функциональным возможностям, но разными по быстродействию, либо разными как по возможностям, так и по быстродействию

Система заданий

Общая система заданий для заданного набора ресурсов может быть определена как система $S = \{T, \|\tau_{ij}\|\}$ следующим образом

- $T = \{T_1, \ldots, T_n\}$ есть набор работ, подлежащих выполнению,
- $\|\tau_{ij}\|$ представляет собой целочисленную матрицу размера $n \times m$, элемент которой $\tau_{ij} > 0$ есть время выполнения работы T_i $(1 \le i \le n)$ на процессоре P_j $(1 \le j \le m)$ Будем полагать, что $\tau_{ij} = \infty$, если работа T_i не может быть выполнена на процессоре P_j, и что для каждого i существует, по крайней мере, одно j, для которого $\tau_{ij} < \infty$ В случае, когда все процессоры идентичны, τ_i обозначает время выполнения T_i на любом процессоре

Показатель эффективности расписаний

Будем рассматривать один из основных показателей эффективности, а именно, *длину расписания*, или *максимальное время завершения*,

$$B = \max_{1 \le j \le m} \{f^j(S)\},$$

где $f^j(S)$ – сумма длительностей работ назначенных на j-й процессор

Основная проблема, таким образом, заключается в нахождении эффективных алгоритмов, позволяющих находить среди всех расписаний такие, для которых эта величина достигает минимума

Во второй главе проводится анализ ранее опубликованных работ по этой теме Рассматриваются встречающиеся в литературе алгоритмы и методы, которые могут быть использованы для решения рассматриваемой задачи·

- случайный и исчерпывающий поиск,
- математическое программирование,
- динамическое программирование,
- метод ветвей и границ,
- быстрые эвристические алгоритмы,
- метод имитации отжига,
- поиск с запретами (Табу-поиск),
- нейронные сети,
- генетические и эволюционные алгоритмы

В результате анализа описанных в литературе результатов были выявлены наиболее перспективные эвристические алгоритмы быстрый жадный алгоритм и метод имитации отжига, которые были в дальнейшем использованы для сравнения и анализа результатов алгоритмов, созданных в рамках данной работ.

В этой главе также приводится исследование опубликованных параллельных стратегий – методов проведения параллельных вычислений, основанных на построении дерева решений При разработке в данной диссертационной работе параллельных алгоритмов была использована одноуровневая стратегия распараллеливания с выделением заданий (в некоторых источниках эта стратегия носит также название «метод выделяемых поддеревьев») В приводимых в литературе данных именно эта стратегия показала наиболее перспективные результаты для машинных архитектур с большими задержками в передаче данных – параллельный вычислительный кластер

В третьей главе описываются разработанные в рамках данной диссертационной работы алгоритмы, а также алгоритмы, которые явились модификациями описанных в литературе методов

В разделах 3 1 – 3.3 описываются алгоритмы, применимые для рассматриваемой задачи в общем виде (процессоры могут различаться как по быстродействию, так и по функциональным возможностям)

Алгоритм «Процессор с ранним окончанием первым» (ПРОП)

Процедура работы этого алгоритма состоит в следующем На k-м шаге распределяемая работа (ее индекс k) назначается на тот процессор, суммарное время выполнения работ на котором с учетом данной работы минимальное Иными словами, минимизируется по j выражение $(B^j_{k-1} + \tau_{kj})$, где B^j_{k-1} – суммарное время выполнения работ, назначенных на j-й процессор на первых k-1 шагах. Алгоритмическая сложность рассматриваемой эвристики составляет $O(nm)$

Вероятностный алгоритм (ВА)

Запишем задачу в следующем виде

$$\sum_{j=1}^{m} x_{ji} = 1, \; i = 1, \quad ,n,$$

$$\sum_{i=1}^{n} x_{ji}\tau_{ij} \le B, j = 1, \quad ,m,$$

$$x_{ji} \in \{0,1\},$$

$$Z=B \to min$$

Поясним смысл такой интерпретации Величина $x_{ji} \in \{0,1\}$ есть показатель того, будет ли выполнена работа T_i на процессоре P_j Так, если x_{ji}= 1, то в соответствующем расписании работа T_i будет выполнена на процессоре P_j; если x_{ji}= 0, то работа T_i не будет выполняться на процессоре P_j При этом первое ограничение означает, что каждая работа должна быть выполнена, а наложение условия целочисленности на x_{ji} означает, что работа будет выполнена только на одном процессоре (расписание без прерываний) Второе условие – это ограничение на длину расписания, а четвертое условие означает, что задача состоит в минимизации длины расписания

При работе алгоритма сначала находится решение релаксационной задачи линейного программирования, в которой $x_{ji} \in [0, 1]$ Пусть получена матрица с элементами x_{ji}^* Затем найдем такую целочисленную матрицу $\bar{x}$, аппроксимирующую x_{ji}^*, что $P(\bar{x}_{ji}=1) = x_{ji}^*$ и выполняется первое из наложенных ограничений Это можно сделать следующим образом Будем заполнять элементы матрицы $\|\bar{x}_{ji}\|$ по строкам для каждого j будем брать последовательно элементы $\bar{x}_{ji}$ с i от 1 до n, полагая $\bar{x}_{ji}$ равным 1 с вероятностью x_{ji}^* Если некоторый элемент $\bar{x}_{ji}$, полученный таким образом, окажется равным 1, то остальные элементы строки (элементы с тем же значением j) полагаем равными 0

В работе доказано, что с вероятностью $P_n(k) \ge \left(1-\left(1-\frac{1}{m}\right)^{mk}\right)^n$ алгоритмическая сложность алгоритма есть $O(nmk)$, где k – свободный параметр алгоритма В этой оценке не учитывается алгоритмическая сложность решения релаксационной задачи, которая зависит от выбранного

алгоритма Исследование этого вопроса лежит вне рамок данной работы, однако, существование полиномиальных методов решения нецелочисленных задач линейного программирования (например, алгоритм Кармаркара, метод эллипсоидов и др), позволяет утверждать, что описанный алгоритм является полиномиальным

На рис 1 приведен фазовый потрет функции вероятности $P_n(k)$ Как видно из графиков, при значениях k, близких к 10, вероятность назначения всех работ достаточно велика, что хорошо коррелирует с результатами вычислительных экспериментов

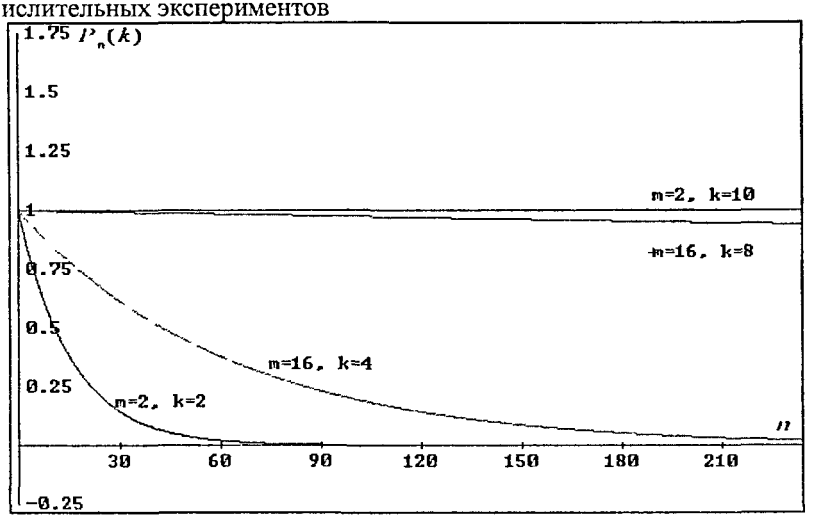


Рис 1 Фазовый потрет $P_n(k)$

Псевдополиномиальный алгоритм с поиском в ширину (ПАПШ)

Дадим следующую интерпретацию рассматриваемой задачи Необходимо решить оптимизационную задачу вида $\left\| \sum_{i=1}^{n} \vec{p}_i \right\| \to \min$[1], где $\vec{p}_i = (0, \ ,\tau_{ik}, \ ,0)$ – m-мерный вектор В теории оптимизации задача в подобной формулировке называется «задачей об упаковке» Однако несложно заметить, что «задача об упаковке» имеет однозначное соответствие с рассматриваемой задачей достаточно провести соответствие вектора $\vec{p}_i = (0, \ ,\tau_{ik}, \ ,0)$ с работой T_i, где k – номер процессора, на который назначена эта работа Работу алгоритма, решающего поставленную задачу, опишем следующим образом Вычислим величину B, которую будем называть директивным сроком Число B можно получить, например, с помощью описанных выше алгоритмов ПРОП или ВА Затем будем

[1] Для произвольного вектора $\vec{a} = (a_1, a_2, \ ,a_m)$, $\left\| \vec{a} \right\| = \max_k a_k$ – величина максимальной компоненты

рассматривать m-мерный куб со стороной B (в векторной интерпретации) Необходимо выбрать вектора $\vec{p}_i$ так, чтобы каждая компонента вектора $\sum_{i=1}^{n} \vec{p}_i$ не превосходила B Под выбором вектора $\vec{p}_i$ будем понимать выбор процессора, на который назначена работа T_i, что однозначно задает вектор Выбор осуществляем путем построения множества точек m-мерного пространства, для каждой из которых на l-м уровне проверяем, принадлежит ли точка m-мерному кубу со стороной B (рис 2) Если условие не выполнено, то точка исключается из дальнейшего рассмотрения

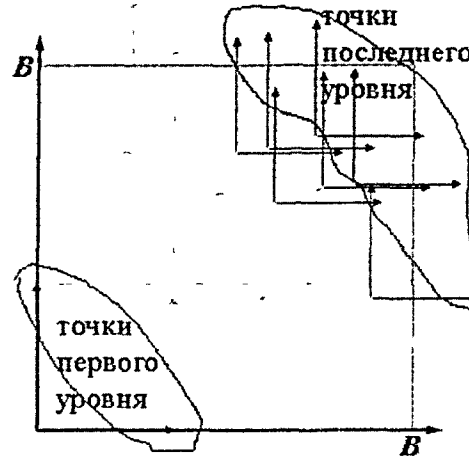


Рис 2 Векторная интерпретация псевдополиномиального алгоритма

В работе доказано, что вычислительная сложность алгоритма есть $O(m^2B^m)$ При фиксированном числе процессоров m алгоритм является псевдополиномиальным с алгоритмической сложностью $O(B^m)$

Псевдополиномиальный алгоритм с поиском в глубину (ПАПГ)

Псевдополиномиальный алгоритм с поиском в глубину отличается от рассмотренного в предыдущем разделе алгоритма следующим Вместо построения полного дерева решений, укладывающегося в m-мерный куб со стороной B, находится одно решение, удовлетворяющее директивному сроку Затем производится уточнение директивного срока, например путем половинного деления Начальный интервал возможных значений B $(\underline{B}, \overline{B})$, где $\underline{B} = \max\left\{\max_i\left(\min_j \tau_{ij}\right), \dfrac{\sum_i \min_j \tau_{ij}}{m}\right\}$, а $\overline{B}$ – значение, полученное каким-либо быстрым алгоритмом (например, ПРОП или ВА) Затем вычисляется величина $B_1=(\overline{B}+\underline{B})/2$, которая используется в качестве следующего значения директивного срока. В случае если решение с таким директивным

сроком найдено, то директивный срок для последующей итерации вычисляется по формуле $B_2=(B_1+\underline{B})/2$, в обратном случае по формуле $B_2=(\overline{B}+B_1)/2$, и т д Таким образом, этот алгоритм является итерационным В худшем случае каждая итерация алгоритма может иметь такую же сложность, как и алгоритм с поиском в ширину, однако в среднем каждая итерация этого алгоритма работает существенно быстрее (это подтверждается экспериментальными данными) Представленный алгоритм обладает также тем преимуществом, что с его помощью можно находить как точные решения (для целочисленных задач), так и приближенные решения с заданной точностью Для этого следует остановить работу алгоритма при разности между верхней и нижней оценками, не превосходящей требуемой точности

В разделе 3 4 приводится описание алгоритмов решения задачи составления расписания на идентичных процессорах, основанных на общей методике агрегирования Этот подход содержит следующие основные элементы последовательное разбиение задачи на подзадачи упорядочения существенно меньшей размерности, формирование дерева подзадач, решение подзадач, формирование решения задачи из решения подзадач в соответствии с построенным деревом подзадач

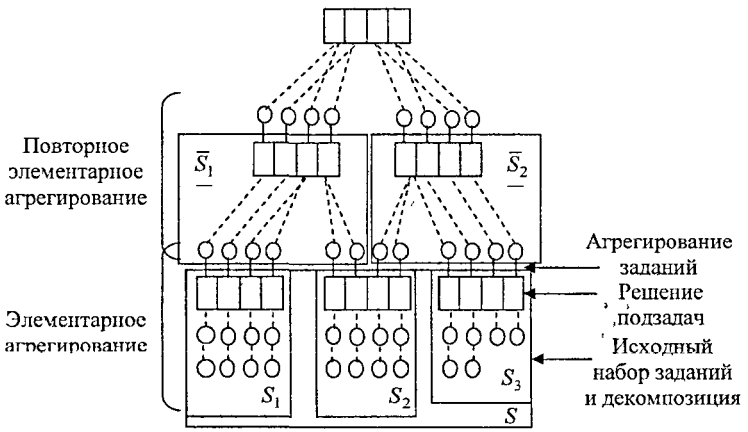


Рис 4 Процесс агрегирования

В диссертационной работе применялся алгоритм декомпозиции исходной задачи, удовлетворяющий следующим условиям

1. Времена решения каждой из подзадач должны быть примерно равны, т о количество работ в каждой из подзадач должно быть примерно одинаковым

2 Длительности работ в одной подзадаче должны быть близки

$$\delta = \sum_{i=1}^{k} \max_{T_i\ T_p \in S^i} | \tau_i - \tau_p | \text{ - мало}$$

3 Время проведения декомпозиции достаточно мало

Работы сортировались по длительности Полученный таким образом набор разбивался на равные по количеству работ поднаборы (по числу подзадач, которое являлось параметром декомпозиции) В случае если количество работ в задаче не было кратно числу подзадач, работы «на стыке» добавлялись в поднабор в соответствии с минимизацией δ

Переборный Агрегирующий Алгоритм (ПАА)

Алгоритм предполагает упорядочение работ в подзадачах и агрегированной задаче путем перебора Перебор может быть полным, либо осуществляться с помощью метода ветвей и границ Этот метод имеет одно существенное ограничение Пусть задачу упорядочения n работ по m процессорам мы хотим разбить на s подзадач Для того чтобы получить подзадачи заметно меньшей размерности, необходимо брать s большим Однако при решении агрегированной задачи нам будет необходимо решить задачу упорядочения ms работ по m процессорам, вычислительная сложность которой может оказаться слишком большой за счет величины s

Комбинированный Агрегирующий Алгоритм (КАА)

Процедура применения этого алгоритма следующая Каждая из подзадач меньшей размерности упорядочивается путем перебора Решение же агрегированной задачи строится с помощью эвристического алгоритма Таким образом, решается проблема, возникающая при применении переборного агрегирующего алгоритма

Многоуровневый Агрегирующий Алгоритм (МАА)

Проводится декомпозиция задачи упорядочения n работ по m процессорам на $s^{(1)}$ подзадач После решения подзадач (переборным алгоритмом) и агрегирования получим задачу упорядочения $ms^{(1)}$ работ по m процессорам Полученная задача снова декомпозируется на $s^{(2)}$ подзадачи, которые решаются аналогично Когда, после некоторого количества повторений, мы получим задачу упорядочения $ms^{(N-1)}$ работ по m процессорам, время решения которой приемлемо, назначение работ проводится в последний раз Таким образом, в этом методе число управляющих переменных составляет N Возможность их варьирования позволяет уменьшить погрешность метода, но увеличивает его алгоритмическую сложность В работе приведены результаты экспериментов, в которых $s^{(0)}=s$, $s^{(i+1)}=[s^{(i)}/2]$

В четвертой главе приводятся аналитические результаты, полученные для алгоритмов с гарантированной точностью Доказываются следующие утверждения

Утверждение 4 1 1

С помощью ПАПГ можно найти решение с погрешностью $\varepsilon = \frac{B - B^*}{B^*}$ за время $O\left(\left(\log\left(\frac{\overline{B} - \underline{B}}{\varepsilon \underline{B}}\right) + 1\right) m^2 \left(\overline{B} - \varepsilon \underline{B}\right)^m\right)$, если $\varepsilon \underline{B} > 1$ и за время $O\left(\log\left(\overline{B} - \underline{B}\right) m^2 \overline{B}^m\right)$, если $\varepsilon \underline{B} \le 1$, где B – длина расписания, найденного ПАПГ, B^* – длина оптимального расписания, $\overline{B}$ и $\underline{B}$ – первоначальные верхняя и нижняя оценки

Утверждение 4 1 2

За время $O\left(\log\log\left(\frac{\overline{B}}{\underline{B}}\right) m^2 \overline{B}^m\right)$ с помощью ПАПГ можно найти расписание длиной B, такое, что $\frac{1}{2} B \le B^* \le B$, где B^* – длина оптимального расписания

Утверждение 4 2

С вероятностью $P_n(k) \ge \left(1 - \left(1 - \frac{1}{m}\right)^{mk}\right)^n$ за время $O(nmk)$ с помощью вероятностного алгоритма можно с вероятностью, большей $1 - \frac{1}{n}$, найти решение с погрешностью $\varepsilon = \frac{B - B^*}{B^*} \le D\left(B^*, \frac{1}{n}\right)$, где

- $D\left(B^*, \frac{1}{n}\right) \le (e-1)\sqrt{\frac{\ln n}{B^*}}$ при $B^* > \ln n$,
- $D\left(B^*, \frac{1}{n}\right) \le \frac{e \ln n}{B^* \ln\left(\frac{e \ln n}{B^*}\right)}$ при $B^* \le \ln n$

Утверждение 4 3 1

Погрешность ε расписания, получаемого алгоритмом ПРОП, составляет $\varepsilon = \frac{B - \underline{B}}{\underline{B}} \le m - 1$, где m – число процессоров, B – длина расписания, полученного алгоритмом ПРОП,

$\underline{B} = \max\left\{\max_i\left(\min_j \tau_{ij}\right), \dfrac{\sum_i \min_j \tau_{ij}}{m}\right\}$ – нижняя оценка длины оптимального расписания

Утверждение 4 3 2

При решении задачи составления расписания на идентичных процессорах, погрешность расписания ε, получаемого алгоритмом ПРОП, составляет $\varepsilon = \dfrac{B - B^*}{B^*} \le 1$, где m – число процессоров, B – расписание, полученное алгоритмом ПРОП, B^*– длина оптимального расписания

Утверждение 4 4

При решении задачи составления расписания работ на идентичных процессорах с использованием алгоритма СДРП погрешность ε получаемого расписания составляет $\varepsilon = \dfrac{B - B^*}{B^*} \le 1$, где B – длина полученного расписания, B^* – длина оптимального расписания

В пятой главе приводятся аналитические результаты для некоторых частных случаев рассматриваемой задачи

При доказательстве утверждения 5 1 предполагается, что длительности выполнения работ задаются зависимостью $\tau_i = \tau_1 + (i-1)d$, $d > 0$, $1 \le i \le n$

Утверждение 5 1

При использовании алгоритма СДРП длительность полученного расписания задается формулой

$B = [n/2m](2\tau_n + d - 2md[n/2m])$, при n=$2km$, где k- целое число,

$B = [n/2m](2\tau_n + d - 2md[n/2m]) + \tau_n - 2md[n/2m]$, при $n - [n/2m]2m \le m$,

$B = [n/2m](2\tau_n + d - 2md[n/2m]) + 2\tau_n - 2md(2[n/2m]+1) + d$, при $n - [n/2m]2m > m$

Следствие Пусть $\Delta B = B - B^*$, где $B^* = \dfrac{\sum_i \tau_i}{m} = \dfrac{2\tau_n - d(n-1)}{2m} n$ Тогда

ΔB=0, при n=$2mk$,

$\Delta B = ([n/2m] - n/2m)(2\tau_n + d) + \tau_n - 2m[n/2m]d([n/2m]+1) + dn^2/(2m)$,

при $n - [n/2m]2m \le m$,

$\Delta B = 2([n/2m] - n/2m + 1)\tau_n + ([n/2m] - n/2m)d - 6md[n/2m] +$

$+dn^2/(2m) - 2md + d$, при $n - [n/2m]2m > m$

при $n - [n/2m]2m > m$

При доказательстве утверждения 5 2 предполагается, что длительности выполнения работ задаются зависимостью $\tau_i + \underline{d} \le \tau_{i+1} \le \tau_i + \overline{d}$, где $\overline{d}, \underline{d}$ – натуральные числа

Утверждение 5 2

При использовании алгоритма СДРП при выполнении условия $2[n/2m](\overline{d}-\underline{d})<\tau_1$ длина полученного расписания удовлетворяет соотношению $B(\underline{d})\le B\le B(\overline{d})$, где $B(\underline{d})$ и $B(\overline{d})$ длины расписаний, полученных алгоритмом СДРП при решении задачи 5 1 с разностью арифметической прогрессии $d=\underline{d}$ и $d=\overline{d}$ соответственно

Утверждение 5 3

В случае фиксированного числа произвольных процессоров m задача составления расписания без прерываний с пустым отношением предшествования ($Rm||C_{\max}$) является псевдополиномиально разрешимой.

Необходимо отметить, что до данного момента не был определен характер NP-трудности рассматриваемой задачи (не было определено, является ли задача NP-трудной в сильном или NP-трудной в обычном смысле, т е псевдополиномиально разрешимой) Наиболее общая формулировка задачи составления расписания, для которой был известен псевдополиномиальный алгоритм, описывает распределение работ по фиксированному числу процессоров, которые могут различаться по быстродействию, но не по функциональным возможностям, при этом i-ая работа становится доступна к назначению после некоторого момента времени r_i ($Qm|r_i|C_{\max}$) Формулировка задачи, используемая в утверждении 5 3 предполагает более общий случай произвольных процессоров, однако, не учитывает времена, когда работы становятся доступными к назначению Определение того факта, что NP-трудная задача является псевдополиномиально разрешимой важно с практической точки зрения. Еще Гэри и Джонсон (1979) отмечают «Псевдополиномиальный алгоритм показывает «экспоненциальное поведение» только в случаях, содержащих «экспоненциально большие» величины, что достаточно редко встречается в задачах, в которых мы заинтересованы Таким образом, алгоритмы такого типа могут служить нашим целям также хорошо, как и полиномиальные алгоритмы»

В шестой главе приводится описание двух параллельных алгоритмов составления расписания

Комбинированный псевдополиномиальный алгоритм

Для решения задачи составления расписаний большой размерности предлагается следующий алгоритм С помощью эвристического алгоритма строится расписание, длина которого $\overline{B}_0$ берется за оценку сверху длины итогового расписания На следующем шаге проводится декомпозиция задачи на подзадачи в соответствии с полученным расписанием Для этого проводится разбиение процессоров на группы с некоторым задаваемым параметром M (M равно количеству групп, на которое разбивается множество процессоров) таким образом, что в первую группу попадает $[m/2M]$ наиболее загруженных процессоров (суммарное время выполнения работ, на которых наибольшее), и столько же наименее загруженных Аналогично строятся остальные группы из оставшихся процессоров Затем производится

декомпозиция множества работ Оно также разбивается на M групп, и в каждую группу попадают те работы, которые были назначены эвристическим алгоритмом на процессоры из соответствующей группы Каждая из полученных таким образом подзадач параллельно решается с помощью описанного выше псевдополиномиального алгоритма

Параллельный псевдополиномиальный алгоритм с поиском в глубину

Опишем работу псевдополиномиального алгоритма поиска в глубину с использованием параллельного вычислительного кластера из q+1 процессоров Будем использовать один из процессоров кластера в качестве управляющего, а в качестве стратегии распределения работ использовать метод выделяемых вершин Управляющий процессор строит дерево решения до уровня $n_0 = \lceil \log_m q \rceil$ (наименьшее целое, большее или равное $\log_m q$), т.е производит назначение первых n_0 работ на процессоры При этом количество концевых вершин дерева решений X (соответствующих возможным расписаниям распределенных работ) больше или равно q Будем предполагать следующее

– Порядок выбора первых работ не важен и не влияет на получение окончательного решения

– В случае, если $X > q$, из полученных концевых вершин выбираются q наиболее перспективных для дальнейшего решения Способ определения перспективности следующий Для каждой из X вершин определяется задача упорядочения оставшихся n–n_0 работ по m процессорам Для каждой из задач определяется нижняя и верхняя оценки длины расписания Нижняя оценка вычисляется путем решения релаксационной задачи линейного программирования (эта оценка заведомо не хуже аналитической, вычисляемой по формуле $\underline{B} = \max\left\{\max_i\left(\min_j \tau_{ij}\right), \frac{\sum_i \min_j \tau_{ij}}{m}\right\}$). Верхняя оценка определяется как меньшее из двух решений, найденных быстрыми эвристическими алгоритмами ПРОП и ВА.

– Вершины, соответствующие решениям с минимальными разностями верхней и нижней оценок, являются более перспективными Этот факт связан с тем, что при использовании итерационного псевдополиномиального алгоритма с поиском в глубину, чем ближе границы поиска, тем быстрее работает алгоритм

– Значение рекорда (длины текущего лучшего расписания) есть меньшее из полученных верхних оценок Вершины, которые соответствуют расписаниям с нижней оценкой, большей рекорда, исключаются из множества активных (далее не рассматриваются)

– В случае, если мощность множества активных вершин меньше q, проводится дополнительное ветвление до уровня $n_0 + 1$, вычисляется новый рекорд и снова строится множества активных вершин

Полученные q наиболее перспективных активных вершин (или все полученные, если $X=q$) выделяются управляющим процессором рабочим процессорам для дальнейшего построения поддеревьев. Рабочие процессоры строят решение задач с использованием псевдополиномиального алгоритма с поиском в глубину. При получении решения одним из рабочих процессов посылается сигнал завершения с уточнением рекорда (если таковое имело место)

Остальные процессы обновляют информацию о рекорде В случае, если не все активные вершины были выделены рабочим процессорам для построения поддеревьев, параллельно с работой рабочих процессоров управляющий процессор проводит дальнейшее ветвление оставшихся активных вершин (и выделение в работу по завершению какого-либо из рабочих процессов) с тем, чтобы по возможности иметь все время q активных вершин Выдача вершин для построения поддеревьев имеет приоритет над дальнейшим ветвлением, поэтому в итоге все активные вершины будут выданы в работу и обработаны Таким образом, процесс имеет завершение

Управляющий процессор также отвечает за хранение информации о текущем рекорде и соответствующем расписании Поэтому по окончании работы всех рабочих процессоров, управляющий процессор может возвратить информацию об оптимальном расписании

В седьмой главе собраны и структурированы все экспериментальные результаты, полученные с помощью разработанных в рамках данной диссертационной работы комплексами вычислительных программ Приводится анализ результатов работы созданных алгоритмов Из приведенных в работе результатов видно, что на широком круге вычислительных задач все агрегирующие алгоритмы дают результаты по погрешности лучше, чем метод имитации отжига (МИО) для данного распределения длительностей работ По времени работы, на многих входах, МИО также уступает агрегирующим алгоритмам. Среди агрегирующих алгоритмов худшие результаты по погрешности дает КАА, что ожидаемо, т к решение агрегированной задачи проводится с помощью быстрой эвристики Наиболее перспективными представляются алгоритмы МАА и КПА КПА представляет собой на самом деле метод улучшения оценки расписания, полученного первоначальным быстрым алгоритмом Для тех случаев, когда изначальное приближение близко к оптимуму КПА за приемлемое время находит улучшение расписания Время работы алгоритма МАА наиболее подвержено влиянию размерности входа, однако, практически во всех случаях найденное решение было лучшим Для задач составления расписания на произвольных процессорах приведенные результаты показывают, что среди эвристических алгоритмов наиболее перспективным является ВА при достаточно малой погрешности алгоритм обладает и малым временем работы Два других эвристических алгоритма ПРОП и МИО получают решение примерно с одинаковым качеством, однако, ПРОП находит решение существенно быстрее Все эвристики

обладают особенностью, что с ростом отношения m/n растет и погрешность получаемого решения. Алгоритмы ВА и ПАПГ могут использоваться в виде пакета прикладных алгоритмов для решения практически любых задач: на входах со сравнительно малым числом задач (а следовательно большим отношением m/n) ПАПГ достаточно быстро улучшает решение, полученное ВА, до оптимума или до решения с гарантированной заданной точностью. При решении задач с большим числом работ, когда получение точного решения невозможно, использование ВА оправдано за счет малой погрешности получаемого расписания.

На диаграммах 1, 2 и 3 приводятся время работы алгоритмов и погрешность получаемого расписания при решении задачи составления расписания на идентичных процессорах. На диаграмме 1 приводятся результаты экспериментов при решении задачи, длительности работ в которой задавались формулой $\tau_i = n - i + 1, \quad i = \overline{1,n}$; на диаграмме 2 – формулой $\tau_i = n - i + 1 + [1.2^{n-i+1}], \quad i = \overline{1,n}$; на диаграмме 3 приводятся результаты экспериментов при решении задачи, длительности работ в которой задавались с помощью программного генератора случайных чисел, позволяющего получать псевдослучайные числа с равномерным распределением на отрезке [1, 1000]. На диаграмме 4 приводится время работы алгоритмов, а на диаграмме 5 погрешность получаемого расписания при решении задачи составления расписания на произвольных процессорах (длительности работ также задавались в помощью генератора случайных чисел).

Диаграмма 1. Результаты для длительностей работ, заданных арифметической прогрессией

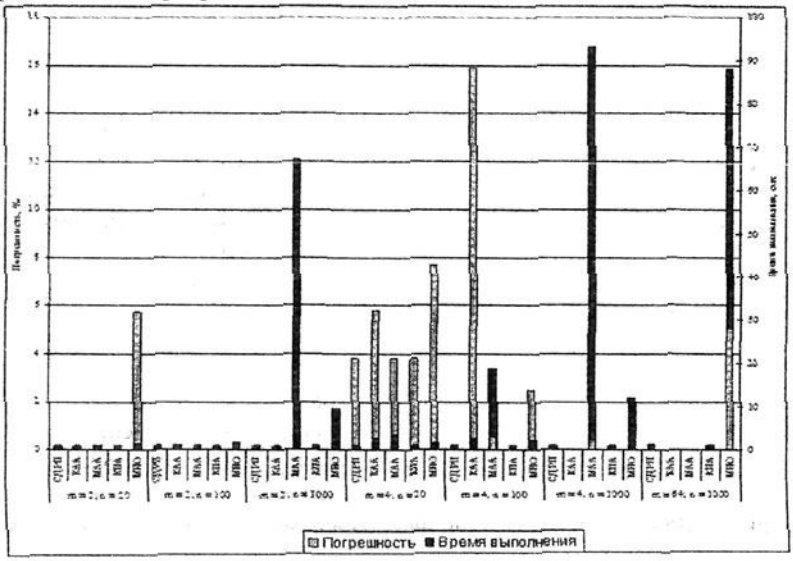


Диаграмма 2. Результаты для длительностей работ, заданных геометрической прогрессией

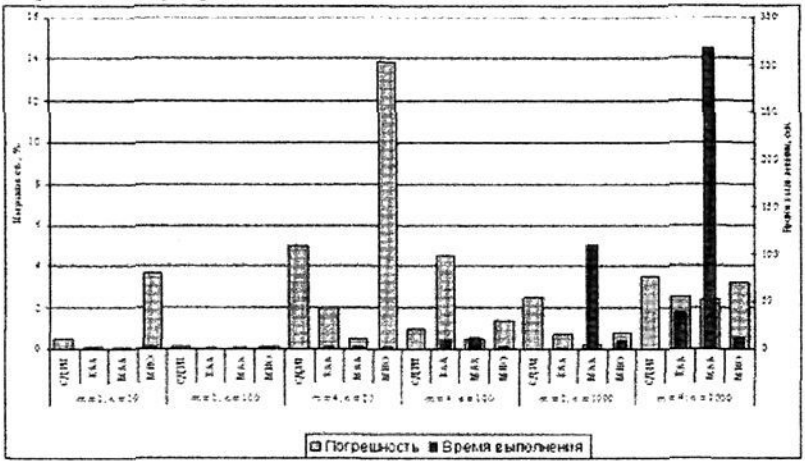


Диаграмма 3. Статистические результаты

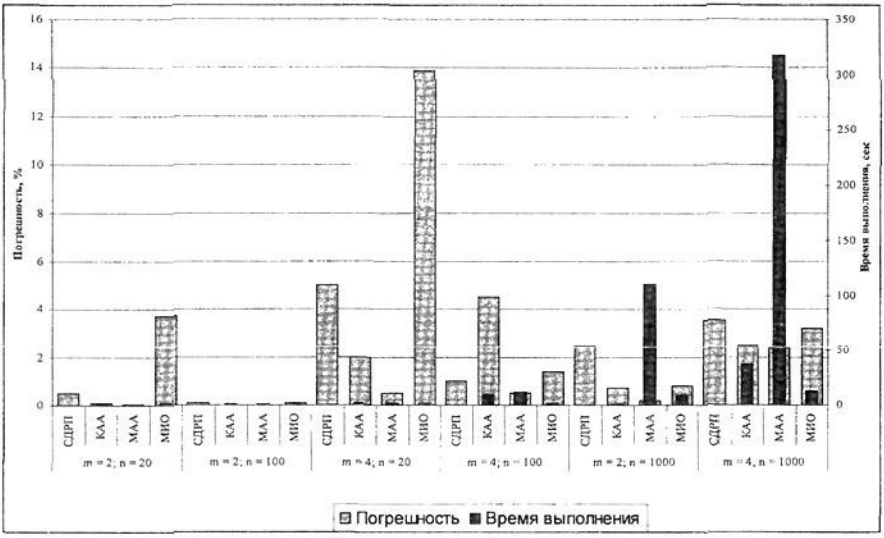
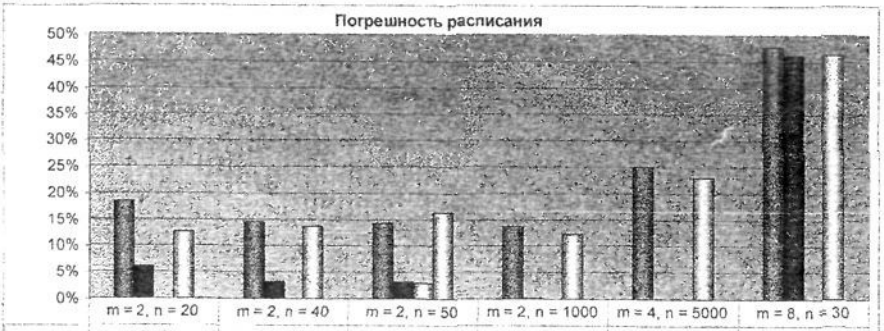


Диаграмма 4. Время работы алгоритмов составления расписания для произвольных процессоров

	m = 2, n = 20	m = 2, n = 40	m = 2, n = 50	m = 2, n = 1000	m = 4, n = 5000	m = 8, n = 30
ПРОП, t	1,2 мс	1,5 мс	2,0 мс	16,0 мс	42,0 мс	1,5 мс
ВА, t	120 мс	183 мс	220 мс	995 мс	55 880 мс	103 мс
ПАПГ, t	20 мс	76 600 мс	320 000 мс			59 900 мс
МИО, t	1 430 мс	2 040 мс	2 560 мс8	720 мс	63 700 мс	1 840 мс

Диаграмма 5. Погрешность алгоритмов составления расписания для произвольных процессоров



	m = 2, n = 20	m = 2, n = 40	m = 2, n = 50	m = 2, n = 1000	m = 4, n = 5000	m = 8, n = 30
ПРОП, e	18,2%	14,6%	14,2%	13,6%	25,1%	47,6%
ВА, e	6,1%	3,2%	3,1%	0,2%	0,1%	46,3%
ПАПГ, e	0,0%	0,0%	2,9%			0,0%
МИО, e	12,6%	13,7%	16,4%	12,3%	22,9%	46,3%

В заключении изложены основные результаты диссертационной работы и указаны возможные направления дальнейших исследований.

ОСНОВНЫЕ РЕЗУЛЬТАТЫ ДИССЕРТАЦИОННОЙ РАБОТЫ:

1. Разработан новый подход к решению задачи составления расписания на идентичных процессорах, заключающийся в агрегировании первоначальных требований системы и решении полученной таким образом задачи меньшей размерности Результаты большого числа экспериментов, полученных с помощью разработанного пакета прикладных программ, подтверждают высокую эффективность предложенного метода и его практическую важность

2 Среди разработанных агрегирующих алгоритмов выявлены наиболее перспективные, которые за допустимое для большинства практических задач время находят точное или близкое к точному решение

3 Для задачи составления расписания на произвольных процессорах разработаны новые эвристические и точный псевдополиномиальный алгоритмы Результаты вычислительных экспериментов позволили сравнить эти алгоритмы с методом имитации отжига Полученные данные позволяют заключить, что новые алгоритмы эффективнее как по быстродействию, так и по точности получаемого решения

4 Для ряда разработанных алгоритмов аналитически получены верхние оценки погрешности получаемых решений

5 Для случая длительностей работ, задаваемых арифметической прогрессией или близких к ней, аналитически получены длина расписания и отклонение от оптимума, получаемые при использовании разработанного эвристического алгоритма

6 В работе доказано, что задача составления расписания без прерываний на фиксированном числе произвольных процессоров с пустым отношением предшествования является псевдополиномиально разрешимой Этот факт является новым, т к на момент написания диссертации были известны псевдополиномиальные алгоритмы, находящие оптимальное расписание на фиксированном числе процессоров, которые могут различаться по быстродействию, но не по функциональным возможностям Таким образом, многие практические задачи рассматриваемого класса, длительности работ в которых не зависят экспоненциально от числа работ, решаются созданным в работе алгоритмом за полиномиальное время

7 В работе создан ряд параллельных вычислительных алгоритмов, которые могут быть предложены к использованию в широком круге практических задач Полученные в экспериментах на параллельном вычислительном кластере значения эффективности распараллеливания близки к единице, а функция изоэффективности алгоритмов является линейной, что означает их хорошую масштабируемость

Основные результаты исследований, проведенных в рамках диссертации, опубликованы в работах

1 *Гончар Д Р, Гуз Д С, Красовский Д В, Фуругян М Г* Эффективные алгоритмы планирования вычислений в многопроцессорных системах // Проблемы управления безопасностью сложных систем Труды 10-й международной конференции Книга 2 /ИПУ РАН – М, 2002 – С 207-211

2 *Красовский Д В, Фуругян М Г* Комплексное применение методов дискретной оптимизации при решении задач минимаксной теории расписаний // МФТИ – М, 2003 «Моделирование и обработка информации» – С 96-103

3 *Гуз Д С, Красовский Д В, Фуругян М Г* Некоторые алгоритмы составления расписаний в многопроцессорных системах //Проблемы управления безопасностью сложных систем Труды 11-й международной конференции /ИПУ РАН – М, 2003 – С 12-14

4 *Гуз Д С, Красовский Д В, Фуругян М Г* Эффективные алгоритмы планирования вычислений в многопроцессорных системах реального времени / ВЦ РАН – М, 2004 – 65 с

5 *Guz D, Krasovskiy D, Furugian M* Effective scheduling algorithms for multiprocessor real-time systems //Proceedings of IV Moscow International Conference on Operations Research /ВЦ РАН – М, 2004 – С 100-103

6 *Гуз Д С, Красовский Д В, Фуругян М Г* Некоторые алгоритмы анализа многопроцессорных систем реального времени / ВЦ РАН – М, 2005 – 42 с

7. *Красовский Д В* Решение задачи составления оптимального расписания без прерываний на произвольных процессорах с использованием вероятностного алгоритма // Современные проблемы фундаментальных и прикладных наук – общая и прикладная физика Сборник трудов 48-й научной конференции МФТИ / МФТИ – М, 2005, – С 76-78

8 *Гуз Д С, Красовский Д В, Фуругян М Г* Эффективные алгоритмы планирования вычислений в многопроцессорных системах реального времени // Методы и средства обработки информации Труды второй всероссийской научной конференции – Москва, издательский отдел факультета ВМиК МГУ 2005, – С 540-545

9 *Красовский Д В, Фуругян М Г* Агрегирование в задаче составления оптимального расписания для многопроцессорных АСУ // Автоматика и телемеханика – 2006 – №12 – С 205-212

10 *Красовский Д В, Фуругян М Г* Псевдополиномиальные алгоритмы упорядочения работ без прерываний по произвольным процессорам // Вестник Московского университета, сер 15, Вычислительная математика и кибернетика, 2006, № 4, – С 25-29

11 *Красовский Д В* Алгоритм параллельного планирования работы многопроцессорных систем большой размерности // Современные проблемы фундаментальных и прикладных наук – общая и прикладная физика Сборник трудов 49-й научной конференции МФТИ, Т II / МФТИ – М 2006 – С 79-80

12 *Красовский Д В , Фуругян М Г* Некоторые алгоритмы составления многопроцессорных расписаний с использованием параллельных вычислений / ВЦ РАН – М , 2006 – 27 с

ЛИЧНЫЙ ВКЛАД

Идеи некоторых эвристических и точных алгоритмов решения задачи составления оптимального расписания без прерываний с пустым отношением предшествования выдвинуты совместно с М Г Фуругяном В совместных работах автору принадлежит теоретическая часть, постановки задач, формулировки и доказательства утверждений, разработка, испытания и исследования предложенных в диссертации алгоритмов Д Р. Гончару и Д С Гузу принадлежит рассмотрение ряда других алгоритмов управления, анализа и синтеза систем реального времени, не вошедших в данную диссертацию

Красовский Дмитрий Владимирович

АЛГОРИТМЫ РЕШЕНИЯ ЗАДАЧИ СОСТАВЛЕНИЯ ОПТИМАЛЬНОГО РАСПИСАНИЯ БЕЗ ПРЕРЫВАНИЙ

Подписано в печать 12.04 07
Формат 60x84 1/16 Бумага офсетная Усл печ л 1,0.
Тираж 70 экз Заказ № 327

Московский физико-технический институт
(государственный университет)
НИЧ МФТИ
141700, Московская обл., Долгопрудный, Институтский пер , 9