МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В.Ломоносова

ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И КИБЕРНЕТИКИ

На правах рукописи

Воронов Василий Юрьевич

МЕТОДЫ РАЗРАБОТКИ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ НА ОСНОВЕ МАШИННОГО ОБУЧЕНИЯ

Специальность 05.13.11 — математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

А В Т О Р Е Ф Е Р А Т
диссертации на соискание ученой степени
кандидата физико-математических наук

Москва — 2009 г.

Работа выполнена на кафедре автоматизации систем вычислительных комплексов факультета вычислительной математики и кибернетики Московского государственного университета имени М. В. Ломоносова.

Научный руководитель:	кандидат физико-математических наук, доцент Попова Нина Николаевна
Официальные оппоненты:	доктор технических наук, профессор Гергель Виктор Павлович,
	кандидат физико-математических наук Малинаускас Костас Костович
Ведущая организация:	Межведомственный суперкомпьютерный центр РАН

Защита диссертации состоится «18» декабря 2009 г. в 11 часов на заседании диссертационного совета Д 501.001.44 при Московском государственном университете имени М. В. Ломоносова по адресу: 119991, ГСП-1 Москва, Ленинские горы, МГУ имени М.В. Ломоносова, 2-й учебный корпус, факультет ВМК, аудитория 685.

С диссертацией можно ознакомиться в библиотеке факультета ВМК МГУ. С текстом автореферата можно ознакомиться на официальном сайте ВМК МГУ имени М. В. Ломоносова http://cs.msu.ru в разделе «Наука» — «Работа диссертационных советов» — «Д 501.001.44».

Автореферат разослан «18» ноября 2009 г.

Ученый секретарь
диссертационного совета
профессор

Н. П. Трифонов

Общая характеристика работы

Актуальность темы. Развитие средств высокопроизводительной вычислительной техники в настоящее время по оценкам специалистов существенно опережает развитие программного обеспечения для таких систем. Актуальной задачей является разработка методов и алгоритмов для построения высокоэффективного системного и прикладного программного обеспечения для многопроцессорных, массивно-параллельных и кластерных вычислительных систем.

Актуальность задачи обуславливается многими факторами, важнейшими из которых являются, прежде всего, сложность процесса разработки параллельных программ и невысокая эффективность использования параллельных систем. Проводимые исследования показывают, что средние оценки производительности многопроцессорных вычислительных систем при решении различных классов задач не превышают 20% от заявляемой, пиковой производительности[1].

Рост производительности многопроцессорных вычислительных систем (МВС) в настоящее время происходит за счет роста числа вычислительных ядер процессора и увеличения числа процессоров, что повышает требования к масштабируемости параллельных программ. Предлагаемый в диссертационной работе подход для построения параллельных программ основывается на следующих предположениях.

Пусть в параллельной программе, реализующей заданный алгоритм, может быть выделен наиболее трудоемкий этап, для выполнения которого может использоваться один из заданного набора подалгоритмов. Назовем такие подалгоритмы *решателями*. Примерами решателей могут служить функции математических библиотек. Выбор используемого решателя и определение значений его параметров влияют на эффективность параллельной программы. В условиях многопроцессорных систем такой выбор должен выполняться как с учетом особенностей входных данных решателя, так и с учетом параметров вычислительной системы, на которой предполагается выполнение параллельной программы. Важным параметром, связанным с эффективностью параллельной программы, является число процессоров, необходимых для выполнения параллельной программы на рассматриваемой, целевой МВС. Про-

[1] L. Oliker et al. Scientific Application Performance on Candidate PetaScale Platforms // Technical report LBNL–62952. 2007. Ernest Orlando Lawrence Berkeley National Laboratory, Berkeley, USA

блема определения необходимого для эффективного выполнения параллельной программы числа процессоров особенно актуальна для вычислительных систем, имеющих большое число (тысячи и более) процессоров.

Работы, направленные на повышение эффективности параллельных программ, активно ведутся как у нас в стране, так и за рубежом. Одним из лидирующих направлений исследований в этой области является проблема автоматической настройки эффективности параллельных программ. Различные аспекты проблемы автоматической настройки эффективности парал\lлеьных программ рассматриваются в работах Дж. Деммеля, В. Эйкхота, Дж. Донгарра, К. Елик, М. Фриго, Т. Катагири и др. В системах, реализующих данный подход (ATLAS, OSKI) автоматическая настройка эффективности осуществляется на основе сбора и анализа статистики о выполнении параллельной программы.

Методы машинного обучения активно применяются в настоящее время в различных научных областях. В диссертационной работе исследуется возможность применения этих методов для решения проблемы повышения эффективности параллельных программ.

Цель работы. Цель данной работы состоит в разработке и исследовании методов построения эффективных параллельных программ на основе машинного обучения. Задачами диссертационного исследования являются:

1. Разработка методов выбора решателя и настройки его параметров для эффективного выполнения параллельной программы на многопроцессорных и многоядерных вычислительных системах;
2. Разработка метода нахождения числа процессоров, необходимых для эффективного выполнения параллельной программы на заданной многопроцессорной вычислительной системе;
3. Разработка инструментальной программной системы, реализующей предложенный подход;
4. Исследование применимости и эффективности предложенных методов на примерах решения конкретных прикладных задач.

Разработанные методы и созданная на их основе инструментальная среда позволят проводить выбор и предварительную оценку методов и поддерживающих их программных средств на этапе проектирования параллельных прикладных программ.

Научная новизна. Все основные результаты работы новые и заключаются в следующем:

1. Предложены новые методы построения параллельных программ, основанные на алгоритмах машинного обучения, позволяющие учитывать особенности входных данных и архитектур целевых многопроцессорных систем. Предложен и исследован новый метод выбора решателя и настройки его параметров для эффективного выполнения параллельной программы на многопроцессорных и многоядерных вычислительных системах. Разработан новый метод для определения числа процессоров, необходимых для эффективного выполнения параллельной программы на заданной многопроцессорной вычислительной системе.
2. Разработаны новые методы и алгоритмы построения инструментальной программной системы для разработки эффективных параллельных программ на основе машинного обучения.

Общие методы исследований. При решении указанных задач используются теория алгоритмов, методы машинного обучения, методы параллельного программирования.

Практическая значимость. Полученные в диссертации результаты имеют большое практическое значение и могут быть использованы для оптимизации использования ресурсов и поддержки работы пользователей МВС. Предложенные методы разрабоки параллельных программ могут применяться для различных классов алгоритмов. В рамках разработанной программной системы реализованы методы построения параллельных программ на основе решения разреженных СЛАУ, где реализации решителей взяты из параллельных математических библиотек PETSc, HYPRE. Разработанная программная система была установлена и прошла апробацию на массивно-параллельной вычислительной системе Blue Gene/P факультета ВМК МГУ, а также на высокопроизводительном кластере СКИФ-МГУ «Чебышев».

Апробация работы. Основные результаты настоящей работы обсуждались на научно-исследовательском семинаре кафедры Автоматизации систем вычислительных комплексов факультета вычислительной математики и кибернетики МГУ под руководством чл.-корр. РАН Л. Н. Королева, совместном семинаре факультета ВМК МГУ и IBM Zurich Research Laboratory, а также докладывались и обсуждались на следующих конференциях:

1. Научная конференция «Тихоновские чтения» (г. Москва, ноябрь 2004 г.),
2. Всероссийские научные конференции «Научный Сервис в сети Интернет» (г. Новороссийск, сентябрь 2004 г., сентябрь 2005 г., сентябрь 2008 г.),
3. Летняя школа по научным вычислениям совместно с Waterford Institute of Technology (г. Москва, август 2006 г.),
4. Международная школа Ph.D. Winter School 2008 (г. Москва, ноябрь 2008 г.),
5. Международная конференция «International Conference on Computing (CIC 2008)» (г. Мехико, Мексика, декабрь 2008 г.),
6. Международная конференция «International Conference on Computing in Engineering, Science and Information (ICCEIS 2009)» (г. Лос-Анджелес, США, апрель 2009 г.),
7. Международная конференция «Numerical Analysis and Scientific Computing Applications (NASCA 2009)» (г. Агадир, Марокко, май 2009 г.),
8. 8-я международная конференция «European Numerical Mathematics and Advanced Applications (ENUMATH-09)» (г. Уппсала, Швеция, июль 2009 г.),
9. Международная конференция «International Conference on High Performance Computing, Networking and Communication Systems (HPCNCS-09)» (г. Орландо, США, июль 2009 г.),
10. Международная конференция «International Conference on Parallel Computing (ParCo-2009)» (г. Лион, Франция, сентябрь 2009 г.).

Публикации. По теме работы имеется 11 публикаций, две из которых в рецензируемых журналах из списка ВАК. Полный список публикаций приведен в конце автореферата.

Структура и объем работы. Текст работы состоит из введения, трех глав, заключения, списка литературы и приложений. Диссертация изложена на 144 страницах, содержит 21 рисунок и 24 таблицы. Список литературы содержит 109 наименований, включая работы автора.

Содержание работы.

Во введении сформулирована проблематика построения параллельных программ. Приведен анализ работ и основных результатов, связанных с темой диссертации. Отмечены особенности существующих подходов и программных систем в данной области. Сформулированы цели диссертационной работы и

аргументирована ее научная новизна. Сформулированы выносимые на защиту научные положения, кратко изложено содержание работы.

В первой главе предложен и обоснован подход к построению параллельных программ, направленный на повышение эффективности использования ресурсов МВС. Рассматривается класс параллельных программ, в которых для выполнения наиболее вычислительно трудоемкого этапа может применяться один из заданного множества алгоритмов (называемого *решателями*). Алгоритм решателя выполняет вычисления над входными данными, имеет множество управляющих параметров и выходные данные в качестве результата. Каждый из управляющих параметров решателя может быть действительной величиной, целочисленной величиной либо одним из значений заданного множества.

Сформулированы следующие задачи построения параллельной программы. Задача *выбора решателя и настройки его параметров* в параллельной программе, выполняющейся на целевой МВС при фиксированном числе процессоров n для снижения сложности выполняемой программы P согласно введенному критерию. В качестве критерия P, определенного как сложность, используется произведение времени выполнения решателя на объем требуемой оперативной памяти. Предложенный критерий соответствует наиболее важным ресурсам МВС, используемым при выполнении параллельной программы. Вторая задача заключается в *определении числа процессоров* $n \in (n_1, n_2]$ *для решателя* S на целевой МВС, для которого обеспечивается повышение эффективности распараллеливания решателя.

В разделе 1.1 представлен метод выбора решателя и настройки его параметров. Пусть задано множество целевых МВС $\{H_1, \ldots, H_k\}$. Рассматривается множество решателей $\mathcal{S}$ параллельной программы, имеющих входные данные A.

Определение 1 ***Признаком*** f_i *входных данных* A *называется значение некоторой действительной функции* $f_i = G_i(A)$.

Обозначим вектор признаков входных данных $F(A) = \{f_1, \ldots, f_l\}^T$.

Определение 2 *Пусть требуется выполнить параллельную программу с входными данными* A *на* n *процессорах целевой МВС* H *с применением решателя* $S \in \mathcal{S}$. ***Функцией сложности*** *решателя назовем зависимость*

$$C : (n, F, S, H) \to P, \tag{1}$$

где F–признаки входных данных A.

Путем проведения вычислительного эксперимента на целевой МВС для различных типов решателей и значений их параметров S и последующего сбора результатов выполнения формируется множество $Tr = \{(n, F, S, H, P)\}$, называемое *обучающей выборкой*. Для элементов обучающей выборки строится функция сложности

$$\mathbf{C}(n, F, S, H) = P.$$

После построения функции сложности выполняется определение решателя и его параметров $S_i \in \mathcal{S}$, для которого достигается минимум

$$\min_{S_i \in \mathcal{S}} \mathbf{C}(n, F, S_i, H) \tag{2}$$

Для поиска минимума предложен генетический алгоритм.

Раздел 1.2 посвящен методу выбора числа процессоров для решателя параллельной программы. Критерием выбора числа процессоров является величина эффективности распараллеливания решателя

$$\frac{T(n_1, F, S, H)}{T(n, F, S, H)(n - n_1)} \tag{3}$$

при выполнении программы на n_1 и $n > n_1$ процессорах, где T–время выполнения решателя. Предложен метод определения числа процессоров из заданного диапазона $n^* \in (n_1, n_2]$, на котором для заданного решателя достигается наибольшее значение (3). Предложенный метод основан на решении задачи бинарной классификации. Пусть $\{(n, F, S, H, T)\}$–выборка, полученная путем проведения вычислительного эксперимента на целевой МВС H для данного решателя S, числа процессоров $n \in [n_1, n_2]$, входных данных из заданного множества $A \in \mathcal{A}$. Каждой паре значений $T_1 = T(n_1, F, H)$,$T_n = T(n, F, H)$ для заданного порога эффективности распараллеливания $\varepsilon \in [0, 1]$ ставится в соответствие одна из двух меток класса $y = \{y^+, y^-\}$:

$$y^+ = \{n \in (n_1, n_2] : \frac{T(n_1, F, H)}{T(n, F, H)(n - n_1)} \geq \varepsilon\}$$
$$y^- = \{n \in (n_1, n_2] : \frac{T(n_1, F, H)}{T(n, F, H)(n - n_1)} < \varepsilon\};$$

Полученные элементы $\{F, H, n_1, n, \varepsilon, y\}$ добавляются в обучающую выборку

Tr. В соответствии с указанным алгоритмом обучающая выборка Tr формируется для каждого значения порога эффективности из множества $\mathbf{E} = \{\varepsilon_i : 0 < \varepsilon_1 < \cdots < \varepsilon_k\}$.

Для элементов сформированной обучающей выборки Tr строится бинарный классификатор:

$$f : (F, n_1, n, \varepsilon) \to \{y^+, y^-\}.$$

Далее для каждого значения $n \in (n_1, n_2]$ определяется максимальное значение $\varepsilon_n \in \mathbf{E}$, для которого $f(F, n_1, n, \varepsilon_n) = y^+$. Результат выбора числа процессоров $n^* = \arg\max_{n \in (n_1, n_2]}(\varepsilon_n)$.

Точность предложенного алгоритма на тестовой выборке определяется величиной точности классификатора f и выбором пороговых значений $\mathbf{E}$.

В разделе 1.3 обосновывается использование методов классификации и регрессии опорных векторов[2] для построения бинарного классификатора f и функции сложности $\mathbf{C}$. Вид построенных с помощью методов опорных векторов классификатора и функции сложности (далее называемых *моделями опорных векторов*) зависит от выбора конкретного типа метода опорных векторов, типа функции ядра и значений его параметров, значений других параметров (параметры C, γ, ν).

В разделе 1.4 рассматриваются особенности реализации разработанных методов построения параллельных программ. Поскольку методы опорных векторов формируют параметрическое семейство моделей, для повышения точности применяются алгоритмы *отбора моделей* с целью выбора конкретного вида классификатора или функции сложности. Предложен эвристический алгоритм отбора моделей опорных векторов путем выбора значений параметров, допускающий распараелелливание. Алгоритм заключается в выборе начального множества значений рассматриваемых параметров V^0, оценке точности модели опорных векторов для каждого элемента $v \in V^0$ и выбора среди них комбинации значений v_0^*, соответствующих модели с наибольшей точностью. На основе значений v_0^* формируется множество V^1 значений параметров модели, которые расположены в окрестности v_0^*, и шаг повторяется. В результате после k итераций результатом алгоритма будут значения параметров v_k^*. После этого модель с наибольшей точностью на обучающей выборке выбирается из моделей с параметрами $\{v_0^*, v_1^*, \ldots, v_k^*\}$.

Для повышения точности построения функции сложности предложено

[2]V. Vapnik, C. Cortes. Support Vector Networks // Machine Learning. 1995. Vol. 20, P. 273–297.

разбивать множество решателей $\mathcal{S}$ на m непересекающихся подмножеств $\mathcal{S} = \bigcup_{k=1,\ldots,m} \mathcal{S}_k$. Для каждого подмножества методом регрессии опорных векторов строится функция сложности $\mathbf{C}_k$, в результате решение (1) заменяется решением m задач на множествах меньшей размерности. В результате декомпозиции каждая функция сложности $\mathbf{C}_k(n, F, S)$ соответствует одному из подмножеств решателей на одной из целевых МВС. Это позволяет снизить время построения функции и применения генетического алгоритма.

В разделе 1.5 сформулированы выводы данной главы.

Вторая глава посвящена методам построения программной системы, реализующей предложенный подход. Сформулированы требования к программной системе. Выделены компоненты системы, реализация которых не зависит от рассматриваемого класса параллельных алгоритмов и решателей, и компоненты, реализация которых зависит от специфики целевых МВС и рассматриваемого класса параллельных программ и их решателей. Предложена распределенная архитектура программной системы, состоящая из *ядра*, функционирующего на сервере программной системы, и функционирующих на целевых МВС *клиентов*.

В разделе 2.1 рассматривается архитектура программной системы. Программная система состоит из следующих модулей:

- Модуль извлечения признаков из входных данных параллельной программы. Модуль устанавливается на каждой целевой МВС;
- Модуль генерации обучающей выборки. Осуществляет сбор результатов и статистики выполнения параллельной программы, обеспечивает использование в параллельной программе реализаций решателей. Модуль устанавливается на каждой целевой МВС.
- Ядро программной системы, в котором реализованы предложенные методы. Содержит интерфейс взаимодействия с пользователем, реализации методов опорных векторов, реализацию генетического алгоритма, реализации алгоритмов отбора моделей опорных векторов и понижения размерности признакового пространства, средства взаимодействия с хранилищем и клиентами целевых МВС. Модуль устанавливается на сервере системы;
- Хранилище. Служит для накопления статистики о выполнении параллельных программ. Модуль устанавливается на сервере системы;

Архитектура программной системы представлена на рис. 1.

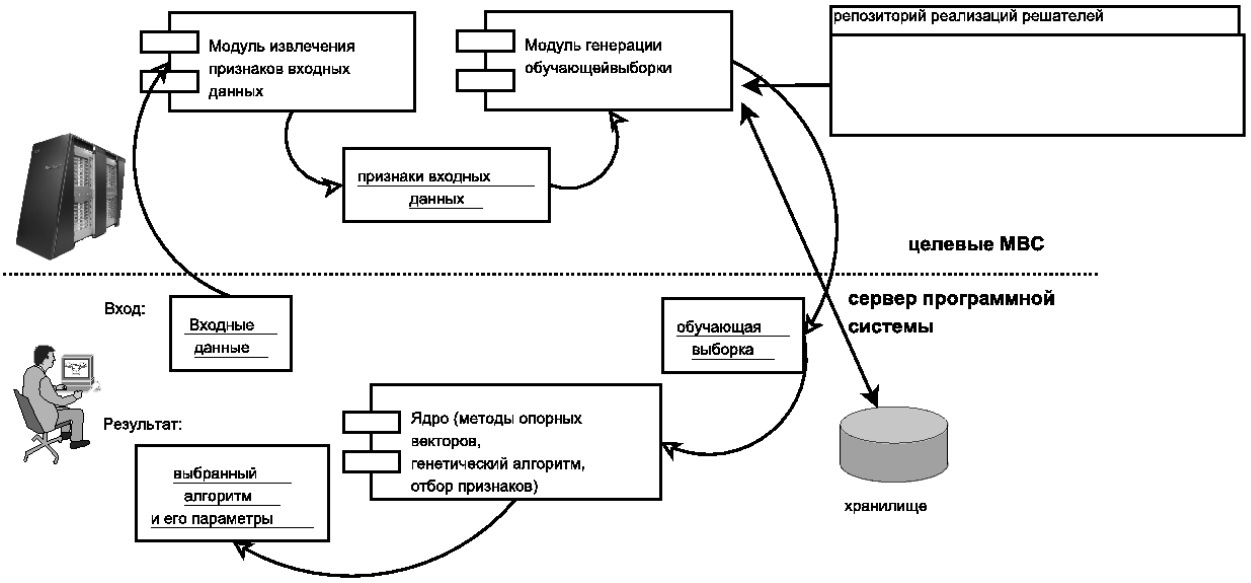

Рис. 1: Архитектура программной системы

Описаны особенности реализации программной системы на примере класса параллельных программ. Рассматривается класс параллельных алгоритмов решения разреженных систем линейных алгебраических уравнений (разреженных СЛАУ), имеющий важные приложения на МВС в научных и инженерных рассчетах. Для поддержки работы с данным классом параллельных программ в программной системе реализованы модули извлечения признаков из разреженных СЛАУ и средства для использования параллельных решателей разреженных СЛАУ из параллельных математических библиотек.

Предложена реализация модуля формирования признакового пространства разреженных СЛАУ на основе системы AnaMod.

В разделе 2.2 рассматриваются вопросы распараллеливания предложенных методов. Наибольшую долю вычислений в предложенных методах занимают этапы выбора моделей классификатора и регрессии, а также генетический алгоритм. В программной системе предложено распараллеливание данных этапов с учетом особенностей рассматриваемых вычислителей, где получили распространение многоядерные процессоры.

Алгоритм отбора моделей обладает естественным параллелизмом по данным, пространство поиска параметров разбивается на подмножества для независимой обработки на каждом ядре, последующего выбора наилучшей модели среди полученных на каждом ядре. Предложена параллельная реализация

генетического алгоритма, основанная на островной модели[3].В рамках параллельной реализации каждый остров особей выделяется на ядро процессора, стадии миграции особей между островами требуют синхронных обменов данными между островами. Практическая реализация указанных параллельных

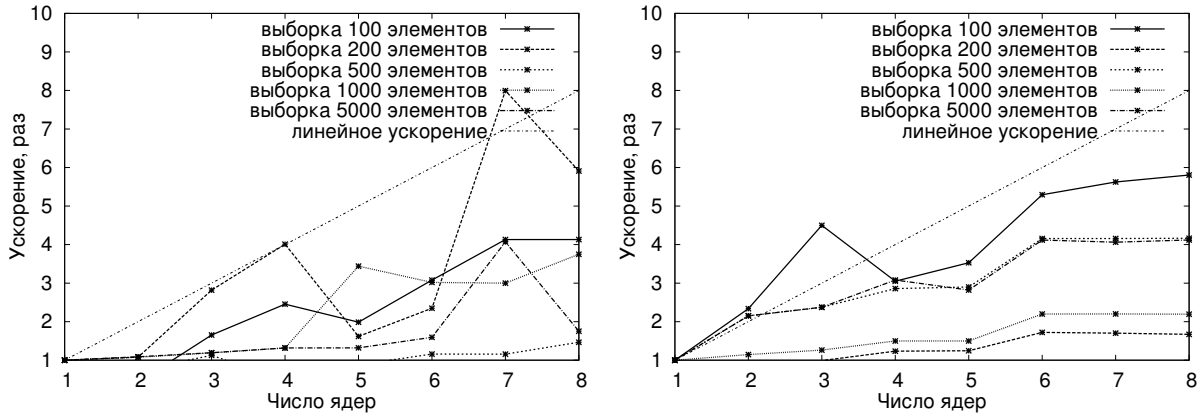

Рис. 2: Ускорение времени выполнения алгоритма отбора моделей опорных векторов для бинарного классификатора (a) и регрессии (b)

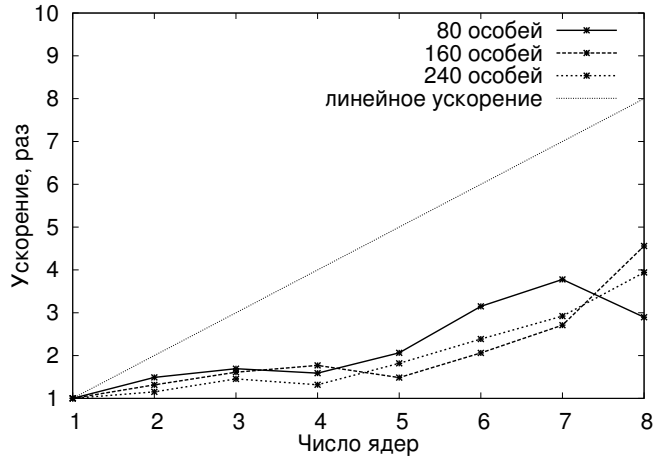

Рис. 3: Ускорение времени выполнения генетического алгоритма при увеличении числа нитей

алгоритмов выполнена на основе интерфейса программирования OpenMP. Анализ параллельных реализаций обоих алгоритмов на восьмиядерной платформе с двумя процессорами Intel 5472 3.2GHz, 16 Gb RAM показывает ускорение при переходе от 1 до 8 нитей до 4,8 раз.

[3]Whitley D., Rana S., Hechendorn R. B. The island Model Genetic algorithm: On separability, population size and convergence // CIT. Journal of computing and information technology. 1999. Vol. 7, N. 1. P. 33–47

Раздел 2.3 посвящен реализации алгоритмов отбора признаков входных данных для повышения точности предложенных методов. Для отбора признаков в методе выбора числа процессоров для решателя предложено использовать метод f-score[4]. В методе выбора и настройки параметров решателя отбор признаков применяется при построении функции сложности **C**. Предложен подход на основе алгоритма частичного сингулярного разложения (truncated SVD) и последующего выбора числа признаков для достижения наибольшей точности построения функции сложности. В результате применения указанных алгоритмов отбора признаков на рассматриваемых выборках данных получено повышение точности методов не менее чем на 4%, время построения методов сократилось в среднем на 17%.

В разделе 2.5 представлены выводы данной главы.

Третья глава посвящена исследованию эффективности предложенных методов на примерах решения практических задач.

В разделе 3.1 представлены результаты применения предложенных методов для набора прикладных задач на основе решения разреженных СЛАУ. Сформирован набор разреженных СЛАУ, взятых из репозитория разреженных СЛАУ[5], возникающих в различных приложениях (структурная механика, квантовая химия, гидродинамика, анализ ДНК, моделирование интегральных схем и др.). Для данных СЛАУ на целевых МВС проведен вычислительный эксперимент и собрана обучающая выборка, необходимая для построения методов. В результате применения построенных методов время решения рассматриваемых задач сокращено в среднем на 7%. На основе полученных результатов сформулированы рекомендации к решению данных задач на целевых МВС с использованием рассматриваемых библиотек.

В разделе 3.2 описывается задача моделирования сетей распределения питания СБИС. Проблема моделирования заключается в определении потенциалов в узлах многоуровневой металлической структуры, которая подключает активные устройства интегральной схемы к источнику питания [6].Характерной особенностью данной задачи является необходимость моделирования сетей питания с числом элементов порядка $10^5 - 10^8$ различных конфигураций и размеров, что требует использования МВС и делает актуальным применение предложеных методов для сокращения времени моделирования. Для

[4]Y.W. Chen, C.J. Lin. Combining SVMs with various feature selection strategies // Studies In Fuzziness And Soft Computing. 2006. Vol. 207. P. 315–324

[5]T. Davies. University of Florida sparse matrix collection // NA digest. 1997. Vol. 97, N. 23, P. 7.

этого выполняется построение методов на обучающей выборке, полученной в результате моделирования одних конфигураций сетей питания для последующего сокращения времени моделирования сетей питания других конфигураций, в том числе для сокращения время моделирования больших моделей сетей питания СБИС на основе построения методов на результатах моделирования сетей меньшей размерности.

В разделе 3.3 рассматриваются особености применения предложенных методов в задаче моделирования сетей питания СБИС.

Для повышения точности предложенных методов и снижения сложности формирования признакового пространства предложено использовать в качестве признаков параметры физической модели сети распределения питания СБИС. В качестве таких признаков используются геометрические и физические характеристики рассматриваемой сети питания СБИС, количество элементов разного типа в сети питания и характеристики их распределения по геометрической модели. Реализация указанных особенностей в программной системе заключается в модификации модуля извлечения признаков и создания в хранилище дополнительной таблицы для физических параметров. Показано, что использование в качестве признакового пространства параметров сетей совместно с численными характеристиками СЛАУ повышает точность формирования классификаторов и регрессии опорных векторов в среднем на 14% на рассматриваемых наборах данных, а также приводит к сокращению времени построения предложенных методов.

В разделе 3.4 обсуждаются результаты применения предложенных методов для параллельной программы моделирования сетей питания СБИС.

Метод выбора числа процессоров для решателя в задаче позволил получить на рассматриваемых наборах сетей питания совпадающие с фактическими значения числа процессоров (рис. 4(a),(b)) с погрешностью определения значения ε_i в среднем 18,7%. Применение метода выбора и настройки решателя для моделирования рассматриваемых наборов сетей питания на МВС Blue Gene/P приводит к сокращению времени вычисления в среднем на 16000 процессор-часов (таблица 1) и в среднем на 3500 процессор-часов на МВС СКИФ-МГУ «Чебышев» (таблица 2) при моделировании 1000 шагов переходного процесса в сети питания.

В разделе 3.5 приводятся выводы данной главы.

В **заключении** перечислены основные результаты диссертационной ра-

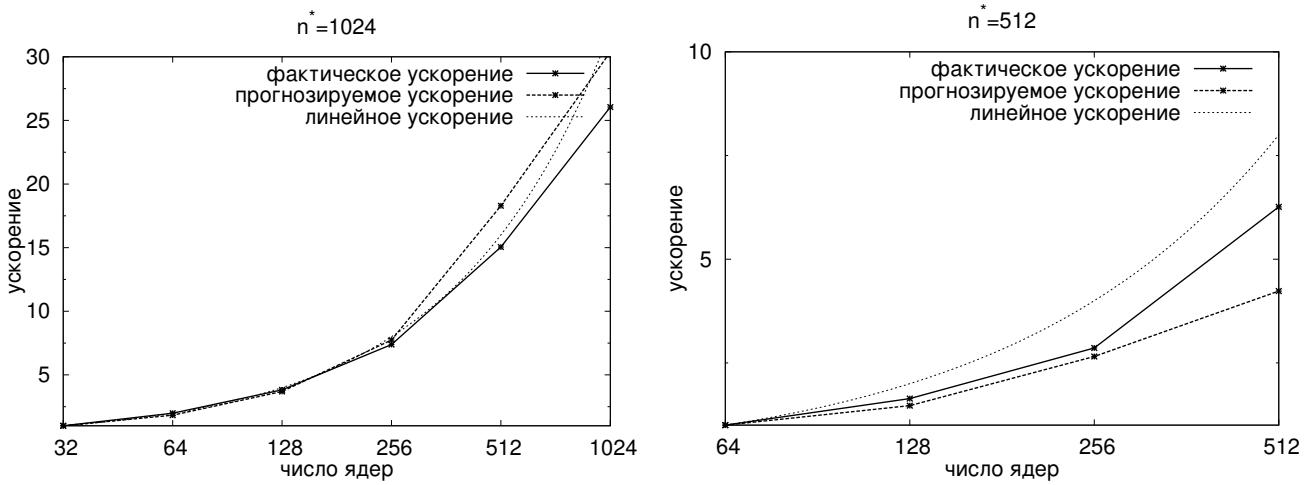

Рис. 4: Результаты применения метода выбора числа процессоров для одного шага переходного процесса на МВС Blue Gene/P (a) и МВС «Чебышев» (b). Для МВС BlueGene/P рассматривается модель сети питания $c10000$, режим запуска программ SMP, построение метода на выборке решения моделей сетей $c9000, c6000, c4000, c3000$. Для МВС «Чебышев» рассматривается модель сети питания $b9000$, построение метода на выборке решения моделей сетей $b6000, b4000, b3000$. n^* обозначает полученное методом число процессоров для решения задачи.

Таблица 1: Результаты применения метода выбора и настройки параметров решателя на Blue Gene/P для n=512 ядер, модель c10000; обучение на статистике решения моделей c3000, c4000, c6000, c9000 для одного шага моделирования переходного процесса

N_{iter}	$P_{predict}$	t_{fact}	$t_{default}$	t_{method}	$t_{economy}$
50	12949743,5	931,17 (10,74%)	1043,3	131,46	112,13 * 512 ≈ 15,9ч
100	12412730,1	957,29 (8,25%)	1043,3	273,55	86,01 * 512 ≈ 12,23ч
200	11716675,0	867,14 (16,88%)	1043,3	568,26	176,16 * 512 ≈ 25,05ч
500	11820381,9	905,31 (13,22%)	1043,3	1417,29	137,99 * 512 ≈ 19,62ч

Обозначения: N_{iter}–кол-во итераций генетического алгоритма,
$P_{predict}$–значение функции сложности для полученного методом решателя S^*, сек. * Мб
t_{fact}–фактическое время моделирования для найденного решателя (в скобках указан процент улучшения по сравнению с решателем по умолчанию), сек.
$t_{default}$–время моделирования для решателя, используемого по умолчанию, сек.
$t_{algorithm}$–время выполнения метода, сек.
$t_{economy}$–сокращение машинного времени в моделировании СБИС при использовании решателя, найденного предложенным методом, в сравнении с решателем по умолчанию, процессор-часы.

Таблица 2: Результаты применения метода выбора и настройки параметров решателя на СКИФ-МГУ «Чебышев» для n=512 ядер, модель b10000; обучение на статистике решения моделей b3000, b4000, b6000, b9000 для одного шага моделирования переходного процесса

N_{iter}	$P_{predict}$	t_{fact}	$t_{default}$	t_{method}	$t_{economy}$
50	8564505	159,31 (9,06%)	175,2	157,21	15,89 * 512 $\approx$ 2,25ч
100	7903257	147,01 (8,25%)	175,2	298,44	28,19 * 512 $\approx$ 4,00ч
200	8139264	151,40 (16,09%)	175,2	641,13	23,8 * 512 $\approx$ 3,38ч
500	7863475	146,27 (16,51%)	175,2	1981,44	28,93 * 512 $\approx$ 4,11ч

Обозначения: N_{iter}–кол-во итераций генетического алгоритма,
$P_{predict}$–значение функции сложности для полученного методом решателя S^*, сек. * Мб
t_{fact}–фактическое время моделирования для найденного решателя (в скобках указан процент улучшения по сравнению с решателем по умолчанию), сек.
$t_{default}$–время моделирования для решателя, используемого по умолчанию, сек.
t_{method}–время выполнения метода, сек.
$t_{economy}$–сокращение машинного времени в моделировании СБИС при использовании решателя, найденного предложенным методом, в сравнении с решателем по умолчанию, процессор-часы.

боты.

Основные результаты работы.

1. Разработаны и исследованы новые методы построения параллельных программ, основанные на алгоритмах машинного обучения, позволяющие учитывать особенности входных данных и параметры архитектур целевых многопроцессорных систем. Предложен и исследован новый метод выбора решателя и настройки его параметров для эффективного выполнения параллельной программы на многопроцессорных и многоядерных вычислительных системах. Разработан новый метод для определения числа процессоров, необходимых для эффективного выполнения параллельной программы на заданной многопроцессорной вычислительной системе.
2. Разработана распределенная инструментальная программная система для построения эффективных параллельных программ, реализующая предложенные методы. Система поддерживает разработку параллельных программ для высокопроизводительных вычислительных систем на основе использования высокоуровневых математических библиотек. Разработанная система поддерживает разработку параллельных программ для

вычислительных систем IBM Blue Gene/P и СКИФ МГУ «Чебышев» и имеет возможность подключения других многопроцессорных и многоядерных систем.

3. Проведено исследование эффективности предложенных методов и разработанной инструментальной системы для класса прикладных задач, основанных на решении разреженных систем линейных уравнений с применением математических библиотек PETSc, HYPRE. Получено сокращение времени решения задач в среднем на 7,8%.
4. С применением разработанной системы выполнено решение задачи моделирования сетей питания СБИС. Разработана параллельная программа для проведения моделирования на системах СКИФ МГУ «Чебышев» и Blue Gene/P. Выбраны решатели и параметры их настройки. Предложенный подход в применении к моделям сетей питания СБИС с числом элементов порядка 10^8 позволил сократить время, необходимое для решения задачи, в среднем на 16000 процессор-часов для системы Blue Gene/P и на 3500 процессор-часов для МВС СКИФ-МГУ «Чебышев» при моделировании 1000 шагов переходного процесса в сети питания.

Автор выражает глубокую благодарность своему научному руководителю, доценту Н. Н. Поповой за постановку задачи, постоянное внимание и помощь в работе.

Публикации

1. *Попова Н. Н., Воронов В. Ю., Джосан О. В., Медведев М. А.* Сравнительный анализ эффективности параллельных вычислений с использованием современных параллельных математических библиотек на примере решения систем линейных уравнений // Труды Всероссийской научной конференции «Научный Сервис в Сети Интернет-2004». — М: Изд-во МГУ, 2004.
2. *Медведев М. А., Попова Н. Н., Воронов В. Ю., Игумнов В. Н.* Практический анализ параллельных методов решения СЛАУ для различного класса матриц на MIMD и SMP платформах // Труды Всероссийской научной конференции «Научный Сервис в Сети Интернет-2005». — М: Изд-во МГУ, 2005.
3. *Воронов В. Ю., Попова Н. Н.* Переносимость параллельных научных библиотек на платформы с мультиядерными процессорами на примере пакета PETSc // Труды Всероссийской научной конференции «Научный Сервис в Сети Интернет-2007». — М: Изд-во МГУ, 2007. — С. 184–187.
4. *Воронов В. Ю., Попова Н. Н.* О применении методов машинного обучения для автоматической настройки параметров решателей СЛАУ в параллельной библиотеке научных вычислений PETSc // Труды Всероссийской научной конференции «Научный Сервис в Сети Интернет-2008». — М: Изд-во МГУ, 2008. — С. 233–236.
5. *Воронов В. Ю.* Метод автоматического выбора и настройки разреженных решателей СЛАУ // *Вестник Московского Университета. Серия 15. Вычислительная математика и кибернетика.* — 2009. — Т. 2. — С. 49–56.
6. *Воронов В. Ю., Попова Н. Н.* Моделирование сетей распределения питания СБИС на многоядерном вычислителе // *Вычислительные методы и программирование.* — 2009. — № 2. — С. 51–60.
7. *Voronov V. Y., Popova N. N.* Parallel Power Grid Simulation on Platforms with Multi Core Processors (accepted) // Proceedings of IEEE International Conference on Computing in Engineering, Science and Information (ICCEIS09). — 2009.
8. *Voronov V. Y., Popova N. N.* A Hybrid Simulation of Power Grids using High-Performance Linear Algebra Packages // Abstract book of Numerical Analysis and Scientific Computing with Applications (NASCA-09) conference. — 2009. — P. 90.
9. *Voronov V. Y., Popova N. N.* Use of Threaded Numerical Packages for Parallel Power Grid Simulation // Proceedings of International Conference on High Performance Computing, Networking and Communication Systems (HPCNCS-09). — 2009. — Pp. 39–45.
10. *Voronov V. Y., Popova N. N.* Machine Learning Approach to Automatic Performance Tuning of Power Grid Simulator // Abstract Book of 8th European Numerical Mathematics and Advanced Applications (ENUMATH-09) conference. — 2009. — P. 291.
11. *Voronov V. Y., Popova N. N.* Automatic Performance Tuning Approach for Parallel Applications Based on Linear Solvers // Abstract Book of International Conference on Parallel Computations (PARCO-09). — 2009. — P. 29.