На правах рукописи

Воеводин Вадим Владимирович

МЕТОДЫ АНАЛИЗА ЛОКАЛЬНОСТИ ДАННЫХ В ЗАДАЧЕ ЭФФЕКТИВНОГО ОТОБРАЖЕНИЯ ПРОГРАММ И АЛГОРИТМОВ НА АРХИТЕКТУРУ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

05.13.11 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

Автореферат
диссертации на соискание учёной степени
кандидата физико-математических наук

Москва – 2011

Работа выполнена в лаборатории параллельных информационных технологий Научно-исследовательского вычислительного центра Московского государственного университета имени М.В. Ломоносова.

Научный руководитель: доктор физико-математических наук, профессор
Машечкин Игорь Валерьевич

Официальные оппоненты: доктор физико-математических наук, профессор
Крюков Виктор Алексеевич,
Институт прикладной математики им. М.В. Келдыша РАН

доктор технических наук, профессор
Гергель Виктор Павлович,
Нижегородский государственный университет им. Н.И. Лобачевского, факультет вычислительной математики и кибернетики

Ведущая организация: Межведомственный суперкомпьютерный центр РАН

Защита состоится 23 декабря 2011 года в 15 часов на заседании диссертационного совета Д 501.002.09 при Московском государственном университете имени М.В. Ломоносова по адресу: 119991, г. Москва, Ленинские горы, д.1, стр. 4, НИВЦ МГУ, конференц-зал.

С диссертацией можно ознакомиться в библиотеке НИВЦ МГУ.

Автореферат разослан 21 ноября 2011 года.

Учёный секретарь
диссертационного совета Суворов В.В

Общая характеристика работы

Актуальность исследования. Развитие высокопроизводительной вычислительной техники, постоянно происходившее все последние годы, сопровождается целым набором важных особенностей. Определить их не сложно, для чего можно воспользоваться различными средствами, например, аккуратно проанализировать список Top500 самых мощных компьютеров мира. Анализ вышедших редакций списка сразу указывает на невероятно устойчивый рост производительности суперкомпьютеров. Каждые 10-12 лет их производительность увеличивается на три порядка, что верно как для систем с рекордной производительностью из первой десятки, так и для аутсайдеров списка. В 1997 году машина ASCI Red достигла терафлопсного порога производительности, в 2008 года суперкомпьютер IBM RoadRunner первым перешагнул петафлопсный рубеж, и у вычислительного сообщества нет больших сомнений в том, что экзафлопсный уровень будет преодолен около 2019-2020 годов.

Вместе с этим, обратной стороной медали постоянного роста производительности суперкомпьютерных систем является резкое снижение эффективности выполнения параллельных программ. Даже на оптимизированном тесте Linpack с хорошо известной структурой, являющегося основой для составления списка Top500, эффективность многих вычислительных систем не превышает 50%, поэтому и эффективность в 3-5% у многих реальных приложений не вызывает большого удивления. Компьютеры теоретически могут многое, но достичь максимальных показателей на практике крайне сложно, а зачастую просто невозможно.

Проблема низкой эффективности приложений была замечена давно, но в последние годы разница между пиковыми и реальными показателями производительности стала особенно заметной. Причин этому несколько, одна из них – это стремительный рост степени параллелизма в архитектуре современных вычислительных систем. Это отчетливо видно, если проанализировать те суперкомпьютеры, которые первыми достигали очередного уровня производительности. Первая мегафлопсная система, CDC 6600 – это 1 процессор, первая гигафлопсная система, Cray 2 – это уже 8 процессоров, первая терафлопсная – 9152 процессора, петафлопсная – 122400 процессоров (ядер). На перспективу эта тенденция явно будет продолжаться, и ожидается, что экзафлопсные вычислительные системы будут объединять в своем составе сотни миллионов независимо работающих ядер. Важно и то,

что идеи параллельной обработки распространяются по всем уровням их архитектуры. Векторная обработка, суперскалярность, конвейерность, VLIW-архитектура, гипертрейдинг, многоядерность, распределенная обработка – все это, как и многое другое, используется в современных суперкомпьютерах, являясь проявлением различных сторон параллелизма. Все эти элементы, безусловно, полезны и увеличивают пиковую производительность машин, но, вместе с этим, если в методе, использованном для решения конкретной задачи, нет нужной степени параллелизма, то его реализация на параллельной вычислительной системе заведомо не будет эффективной.

Другой, не менее серьезной причиной снижения эффективности программ является сложная структура подсистем памяти в современных компьютерах. Давно подмечено, что скорость работы процессора намного превышает скорость работы оперативной памяти, из-за чего при выполнении операций чтения/записи процессор вынужден простаивать в ожидании завершения работы с данными. Для согласования скоростей работы процессора и памяти введены регистры, различные уровни кэш-памяти, появилось понятие иерархии памяти. Латентность при обращении к кэш-памяти 1-го уровня в современных процессорах составляет 3-4 такта, при обращении к кэш-памяти 2-го уровня – около 15 тактов, при обращении к оперативной памяти – 150-350 тактов. Чем выше степень локальности команд и данных в программе, тем чаще происходят обращения к более высокому уровню иерархии памяти, тем выше эффективность программы и быстрее она выполняется.

Что происходит в реальных программах? Вопрос крайне важный, но далеко не однозначный. Многое определяется использованным для решения исходной задачи методом, алгоритмом, конкретной технологией программирования, стилем написания программ. В конечном итоге, все определяется тем, насколько хорошо соответствуют свойства программ и алгоритмов особенностям архитектуры вычислительных систем. Еще в 80-х годах прошлого столетия академик Г.И. Марчук сформулировал задачу отображения структуры программ и алгоритмов на архитектуру компьютеров, являющейся первичной при исследовании эффективности. С начала 90-х годов задачей отображения активно занимался академик В.В. Воеводин, который ввел и предложил методы исследования информационной структуры программ и алгоритмов – понятия, крайне необходимого для введения в задачу отображения идей параллелизма.

Многие известные российские и зарубежные ученые внесли вклад в решение задачи отображения. Задача рассматривалась с разных сторон, делались разные акценты в исследованиях. Кто-то больше внимания уделял архитектуре компьютеров, другие объектом исследований делали структуру программ, третьи изучали особенности алгоритмов, одни группы занимались теорией оптимизации программ, другие создавали инструментальные средства исследования их структуры. Но, несмотря на значительный возраст самой проблемы, в настоящее время можно с уверенностью говорить не о найденном решении задачи отображения, а о решении множества более частных задач.

Целью данной диссертационной работы является исследование и структуризация задачи отображения программ и алгоритмов на параллельные вычислительные системы, разработка подхода к ее решению. Подход должен учитывать необходимость согласования свойств алгоритмов, структуры программ и особенностей архитектуры параллельных вычислительных систем.

Необходимо провести детальный анализ свойств работы программ с памятью и выполнить исследование структуры подсистем памяти современных вычислительных платформ. На основании этого, необходимо выделить набор характеристик, описывающих взаимодействие программ с памятью. Характеристики набора должны отражать иерархическую природу подсистем памяти, позволяя исследовать как общие, интегральные свойства работы программ с памятью в целом, так и описывать локальные особенности ключевых фрагментов.

Научные результаты, выносимые на защиту:

1. Предложен и апробирован подход к структуризации и решению задачи эффективного отображения алгоритмов на вычислительные системы, опирающийся на совместный анализ свойств и характеристик трех основных объектов: алгоритмов, программ и архитектур, рассматриваемых с точки зрения потенциала параллелизма и степени локальности данных.
2. Предложен метод исследования степени локальности данных в программах с помощью набора формальных характеристик, отражающих иерархическую структуру памяти современных компьютеров. Это позволяет оценивать как интегральные свойства локальности использования данных в целом, так и проводить детальный анализ эффективности работы с памятью на уровне отдельных фрагментов.
3. Определены характеристики множества реальных типовых алгоритмических структур. На основе большого числа вычислительных экспериментов показана зависимость эффективности выполнения программ от степени несоответствия свойств программ особенностям архитектуры. Показана важность исследования профиля обращений программ в память, предложены методы и технологии экспериментального и визуального анализа профиля реальных программ.

Научная новизна диссертационной работы заключается в разработке подхода к решению задачи отображения программ и алгоритмов на параллельные вычислительные системы. Подход опирается на согласование свойств алгоритмов, структуры программ и особенностей архитектуры параллельных вычислительных систем. Выделен набор характеристик, описывающих свойства алгоритмов и влияющих на эффективность решения задачи отображения, определены характеристики множества реальных типовых алгоритмических структур. Введен набор формальных характеристик, отражающий иерархическую структуру памяти современных компьютеров, позволяющих оценивать как интегральные свойства локальности использования данных в программах в целом, так и проводить детальный анализ эффективности работы с памятью на уровне отдельных фрагментов. Введенные характеристики позволяют оценивать и степень локальности данных в отдельных программах, и проводить сравнительный анализ свойств работы с памятью множества программ.

Теоретическая и практическая значимость работы определяется ее исходной ориентацией на решение одной из основных задач вычислительной практики – повышение эффективности работы параллельных приложений и суперкомпьютерных систем. Созданный подход может быть использован в качестве основы для разработки программно-аппаратной инфраструктуры, обеспечивающей эффективное решение задачи отображения. На основе разработанного подхода и большого числа вычислительных экспериментов составлены рекомендации по приведению свойств программ к особенностям архитектуры. Проведены аналитические и экспериментальные исследования свойств алгоритмов, структуры программ и особенностей архи-

тектуры параллельных вычислительных систем, показавшие эффективность предложенного подхода.

Соответствие диссертации паспорту научной специальности. Содержание и результаты работы соответствуют паспорту специальности 05.13.11, а именно, включают разработку новых методов и моделей анализа задачи эффективного отображения алгоритмов и программ на архитектуру вычислительных систем, а также разработку методов и моделей анализа эффективности работы с памятью.

Апробация работы. Основные положения работы прошли обсуждение на научных семинарах НИВЦ МГУ и кафедры АСВК факультета ВМК МГУ, а также на спецсеминаре «Вопросы распределенной обработки информации» факультета ВМК МГУ. Результаты работы представлялись на международной научной конференции «Параллельные вычислительные технологии-2010» (г Уфа, март-апрель 2010 г.), на Третьей Международной научной конференции «Суперкомпьютерные системы и их применение» (г. Минск, май 2010 г.), на X Международной конференции «Высокопроизводительные параллельные вычисления на кластерных системах» (г. Пермь, ноябрь 2010 г.), на международной научной конференции «Параллельные вычислительные технологии-2011» (г Москва, март-апрель 2011 г.), на XI Всероссийской конференции «Высокопроизводительные параллельные вычисления на кластерных системах» (г. Нижний Новгород, октябрь-ноябрь 2011 г.).

Результаты работы были апробированы в рамках выполнения работ по гранту РФФИ (№ 09-07-00168-а), а также при выполнении государственного контракта с Министерством образования и науки Российской Федерации № 07.514.12.4001, выполняемого в рамках скоординированного конкурса между Россией и Евросоюзом.

Публикации. По теме диссертации опубликовано 5 научных работ, из них 3 в журналах из списка ВАК.

Структура и объем работы. Общий объем диссертации – 144 страницы. Диссертация состоит из введения, 4 глав и заключения, 42 источников использованной литературы и одного приложения, 11 таблиц и 28 рисунков.

Содержание работы

Во Введении приведено обоснование актуальности данной работы и ее научной новизны, сформулированы основные цели и приведено краткое описание структуры диссертации.

Первая глава посвящена описанию существующих подходов к решению задачи эффективного отображения алгоритмов на параллельные вычислительные платформы, а также обзору существующих инструментов, связанных с ее решением.

Двумя основными проблемами эффективного отображения программ и алгоритмов являются сложность подсистем работы с памятью и значительный объем заложенного в аппаратуре параллелизма. Наличие этих проблем привело к тому, что сегодня очень небольшое число приложений в области высокопроизводительных вычислений достигает эффективности большей, чем несколько процентов.

К настоящему моменту проведено достаточно обширное исследование задачи эффективного отображения алгоритмов на вычислительные платформы. Первые продвижения были сделаны еще в 80-х годах академиком Г.И. Марчуком. Далее они были продолжены в 90-х академиком В.В. Воеводиным, которым было разработано новое направление – использование информационной структуры алгоритмов и программ для изучения их свойств на основе понятия графа алгоритма.

В настоящее время рассматриваемая область получила новый виток развития. Причиной этому послужил следующий факт: все основные существующие технологии параллельного программирования работают для программ, которые могут быть распараллелены до уровня сотен и тысяч параллельных процессов или потоков, однако уже в ближайшем будущем вычислительные системы будут обладать миллионами ядер. Необходимо разрабатывать новые парадигмы программирования, в которых обязательно должно учитываться строение аппаратных платформ, поскольку иначе эффективность выполнения программ будет продолжать снижаться. А для этого необходимо проводить исследование эффективности всего процесса отображения, начиная от уровня алгоритма и заканчивая архитектурой вычислительных систем. Подобные исследования начинают проводиться в рамках различных крупных проектов, однако проблема еще далека от решения.

Помимо описанных исследований, на текущий момент существует множество различных программных средств, которые помога-

ют проводить анализ эффективности приложений и выполнять их оптимизацию. Они используют различные подходы к исследованию эффективности и позволяют с разных сторон подойти к этому вопросу. Одни системы проводят анализ эффективности работы с памятью, другие – анализ эффективности использования доступного ресурса параллелизма. Некоторые средства изучают коммуникационный профиль программ, другие – профиль обращений к данным. Различаются и используемые для этого подходы: одни системы работают на основе изучения свойств программы в статике, другие опираются на динамику, может использоваться анализ показаний аппаратных счетчиков или данных моделирования, и т.д.

Однако все данные программные средства предлагают подходы к решению только отдельных частных задач и помогают проводить исследование лишь отдельных составляющих всей задачи отображения. Они не позволяют взглянуть на всю проблему эффективного отображения в целом.

Проведенный обзор показал, что, несмотря на проводимые исследования в данной области, на текущий момент задачу отображения нельзя считать решенной. Вместе с этим выполненный обзор был в работе использован в качестве основы для структуризации задачи отображения и разбиения ее на отдельные составные части и этапы. Более того, хотя существующие программные системы не позволяют проводить исследование процесса отображения алгоритмов на вычислительные платформы в целом, они во многих случаях позволяют осуществлять качественный детальный анализ конкретных свойств программы. Поэтому при создании в будущем комплексной программной инфраструктуры для решения задачи эффективного отображения эти системы могут быть использованы в качестве готовых компонент по изучению отдельных частей этой задачи.

Вторая глава содержит описание предлагаемого подхода к решению задачи эффективного отображения алгоритмов на вычислительные платформы.

Задача отображения состоит из тех же этапов, что и задача создания обычной программы. Сначала происходит выбор подходящей технологии программирования, затем идет написание программы, выбор подключаемых библиотек, компилятора и его опций, после чего производится выполнение программы на целевой вычислительной системе. Однако нас интересует вопрос эффективности, поэтому мы стремимся на каждом этапе определить те его отличительные свойства, которые позволяют оценивать именно эффективность отображения. Такие свойства будем называть характеристиками. В соответствии с описанными в первой главе двумя основными проблемами эффективного отображения, при исследовании характеристик внимание уделяется и свойствам работы с памятью, и использованию ресурса параллелизма. Обе проблемы важны, однако главный акцент в представляемой диссертации делается именно на анализе свойств работы программ с памятью. Сформированная в процессе исследования схема процесса отображения с примерами характеристик приведена на рис. 1.

В процессе отображения можно выделить три основные составляющие – алгоритм, программа и аппаратура. В рамках данной работы мы предполагаем, что алгоритм и аппаратура фиксированы, то есть необходимо создать реализацию выбранного алгоритма, которая будет эффективно выполняться на выбранной аппаратуре. Это не накладывает ограничений, поскольку при необходимости можно выбрать другой алгоритм или другую аппаратную платформу, и для них провести новое исследование. Характеристики этих двух составляющих фиксированы. Соответственно, для получения эффективного отображения необходимо создать такую программу, характеристики которой, с одной стороны, будут определяться характеристиками алгоритма, а с другой, должны соответствовать характеристикам аппаратуры.

Далее рассматриваются характеристики каждой из трех составляющих. Для изучения свойств алгоритма в работе используется его представление в виде графа алгоритма. На его основе выделяется набор характеристик, которые позволяют оценивать основные свойства алгоритма. К таким характеристикам относятся: общее число операций, критический путь графа алгоритма, максимальная и средняя ширина яруса в ярусно-параллельной форме, среднее число исходящих ребер на ярусе, свойства ребер, зависящих от входных данных, и ряд других.

Эти характеристики позволяют определить ограничения, которые алгоритм накладывает на будущую реализацию. Например, критический путь графа позволяет оценить параллельную сложность алгоритма, то есть минимальное число шагов, которое необходимо выполнить последовательно при любой реализации данного алгоритма; максимальная ширина графа показывает, какое максимальное число операций алгоритма может выполняться параллельно; и т.д.

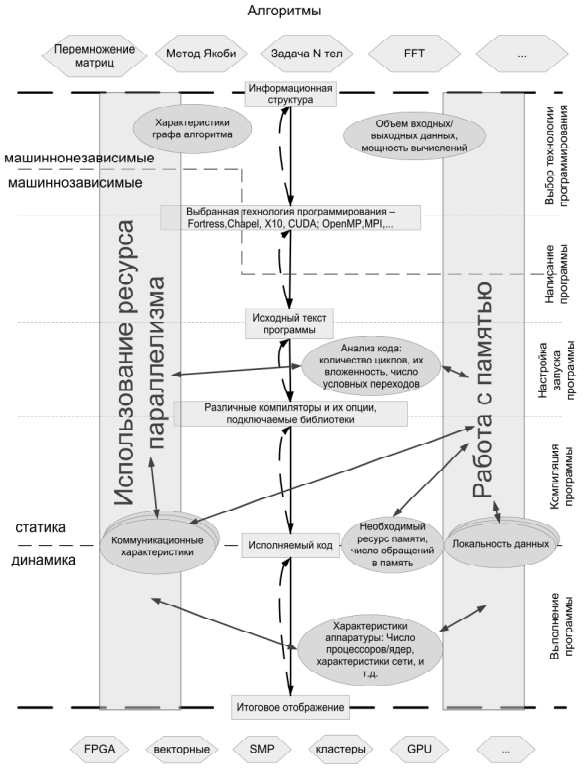

Рис. 1. Общая схема процесса отображения алгоритмов на архитектуру вычислительных систем

Задача характеристик программ состоит в том, чтобы помогать определять степень соответствия между программой и аппаратурой, с учетом ограничений, наложенных алгоритмом.

На уровне программ возникает деление на характеристики статики и динамики. К первому типу относятся характеристики, описывающие, например, параметры циклов или свойства используемых массивов, полученные на основе анализа исходного текста программы. Ко второму – вычисленные во время выполнения программы характеристики, определяющие, например, свойства последовательности обращений в память в процессе или объем передаваемых данных между процессами.

Характеристики аппаратуры нужны для того, чтобы уметь оценивать эффективность полученной реализации алгоритма. По своей сути, эти характеристики составляют описание используемой вычислительной системы. В данной работе мы сосредоточили внимание на кластерных системах, поскольку к настоящему моменту в области суперкомпьютерных вычислений данный класс является доминирующим. Характеристики кластерных систем также разбиваются на две группы – влияющие на эффективность работы с памятью и эффективность использования ресурса параллелизма. К характеристикам первой группы можно отнести, например, свойства оперативной или кэш-памяти, ко второй группе – общее число узлов в системе или степень суперскалярности процессоров. В рамках каждой группы проводится дальнейшее деление на три уровня – характеристики всего кластера, вычислительного узла и процессора, что приводит к иерархическому описанию кластерных систем.

Подобные характеристики крайне необходимы, поскольку помогают определять, почему программа выполняется недостаточно эффективно. Например, причиной снижения эффективности может быть большое число промахов в TLB-буфере, неэффективная работа аппаратного префетчера или недостаточный объем кэш-памяти, и совместное исследование характеристик программы и аппаратуры позволяет это выявить.

Третья глава посвящена подробному исследованию характеристик программ, описывающих эффективность работы с памятью.

Для того чтобы обеспечивать высокую эффективность работы с памятью в программе, необходимо решать следующие задачи:

- нужно уметь обнаруживать факт низкой эффективности,
- нужно уметь определять причины возникновения низкой эффективности,
- нужно уметь устранять эти причины, то есть проводить соответствующую оптимизацию программы.

Решение этих задач может быть построено на основе исследования локальности работы с памятью. В данной работе изучение локальности выполняется с помощью анализа потока обращений. Вводится формальное описание потока обращений в память, который представляет собой последовательность используемых в программе переменных, расположенных в том порядке, в котором происходят обращения к этим переменным в программе. Затем указываются его свойства, а также допущения, которые были приняты при проведении исследования: на данный момент в работе рассматриваются последовательные программы, и только операции с массивами, которые являются основой для большинства вычислительных программ. Массивы изучаются независимо друг от друга, что позволяет рассматривать поток обращений как последовательность индексов элементов, к которым происходят обращения.

В будущем планируется переход от исследования потока индексов к исследованию потока физических или виртуальных адресов, что позволит учитывать обращения не только к массивам, но и к любым другим структурам данных. Разработанные методы исследования потока индексов могут в дальнейшем эффективно применяться и к потоку адресов.

После определения потока обращений приводится формальное описание общих характеристик, которые предназначены для качественной оценки эффективности. Эти характеристики оценивают локальность работы с памятью в рамках потока обращений, и могут быть разделены на два класса – описывающие пространственную локальность и временну́ю локальность. В общем случае пространственная локальность отражает среднее расстояние (в терминах адресов) между несколькими последовательными обращениями в память; временна́я локальность показывает частоту обращений по одному адресу в память за время исполнения всей программы.

Для описания пространственной локальности вводятся две характеристики:

- Первая характеристика оценивает среднее расстояние по памяти между соседними обращениями. В терминах введенного потока обращений расстояние измеряется как разница между индексами элементов.
- Вторая характеристика призвана выделять случаи с наиболее высокой пространственной локальностью и описывает среднюю длину цепочек последовательных обращений. Два идущих подряд обращения называются последовательными, если они происходят к одному и тому же элементу или соседним элементам массива.

Временна́я локальность описывается тремя характеристиками:

- среднее время между повторными обращениями к переменным,
- среднее число обращений к переменным,
- степень повторного использования переменных.

Среднее время между повторными обращениями оценивает, насколько часто за время выполнения программы в среднем происходят обращения к одним и тем же элементам. Среднее число обращений показывает, сколько в среднем за время выполнения программы было произведено обращений к одним и тем же элементам. Степень повторного использования служит показателем того, какая доля переменных используется однократно, что является наихудшим случаем с точки зрения временно́й локальности.

Данные характеристики позволяют получить общее представление о свойствах потока обращений и на качественном уровне оценить, обладает ли исследуемый поток хорошей или плохой локальностью.

Общие характеристики не учитывают особенности строения подсистемы памяти и поэтому не во всех случаях могут предоставить информацию нужного уровня подробности. Для определения детальной структуры потока обращений вводятся тонкие характеристики работы программ с памятью, которые принимают во внимание такие важные свойства памяти как иерархичность (подсистема памяти устроена в виде иерархии) и локальность (время обращения к разным областям памяти неодинаково и зависит от удаленности запрашиваемых данных в памяти от предыдущих).

Для определения тонких характеристик используется разбиение потока обращений на кластеры. Кластер представляет собой набор идущих подряд элементов потока, который обладает высокой локальностью. Это позволяет считать, что внутри кластера эффективность работы с памятью достаточно высока, и основные потери эффективности происходят при переходе от кластера к кластеру. С помощью такой кластеризации можно выделять случаи, которые плохо описываются усредненными общими характеристиками. Например, поток обращений устроен таким образом, что к каждому элементу происходят два обращения подряд, после чего повторное обращение происходит через большой промежуток времени. В таком случае среднее время между повторными обращениями будет равно некоторому среднему значению, не отражающему структуры потока. При использовании

кластеризации два обращения подряд будут объединены в один кластер, что позволит отделить эти обращения от остальных и учитывать их отдельно друг от друга.

Характеристики кластеров определяются на основе общих характеристик, однако применяются к потоку обращений, разбитому на кластеры. Это позволяет по-другому оценить локальность и дополнить полученные ранее данные по потоку в целом.

Далее выделяется еще одна группа тонких характеристик. Практически во всех программах основная вычислительная часть сконцентрирована в циклах. При этом зачастую разные итерации цикла состоят из одного и того же набора операций, и, соответственно, обладают «подобными» наборами обращений в память. Два набора обращений, или фрагмента, называются подобными, если один может быть получен из другого путем параллельного сдвига по памяти.

На основе проведенной кластеризации вводится набор характеристик, оценивающих подобные фрагменты в потоке. Использование для определения подобных фрагментов потока кластеров, а не начального потока обращений, обусловлено двумя причинами. Во-первых, поиск подобия внутри кластеров проводить не имеет смысла, поскольку по определению внутри кластера эффективность работы с памятью высока, и нет нужды более детально исследовать его строение. Во-вторых, это помогает проводить более глубокое исследование свойств кластеров, что совместно с их характеристиками позволяет получать более детальную информацию. Как и в случае кластеризации, характеристики фрагментов построены на основе общих характеристик, однако применяемых не к потоку обращений, а к потоку фрагментов.

При выделении подобных фрагментов происходит поиск последовательности кластеров, которая повторяется в потоке несколько раз. Знание о таких фрагментах позволяет лучше понимать структуру потока обращений, поскольку подобные фрагменты обладают одинаковыми или очень схожими свойствами.

В отдельных случаях помочь в изучении потока обращений и качественно оценить его структуру может его визуализация. Для потока обращений строится график, по оси абсцисс которого отложена последовательность обращений, а по оси ординат – адрес элемента (или его индекс в массиве), к которому происходит обращение. Анализ подобного графика помогает локализовать узкие места и выделить зависимости в строении структуры, которые не позволяют определить

описанные ранее характеристики. В последнем разделе третьей главы приведено описание данного метода визуализации.

В **четвертой главе** приведены основные практические результаты. На основе сформулированных в предыдущих главах идей проведено исследование различных типовых алгоритмических структур. С помощью характеристик алгоритма проведен сравнительный анализ различных типовых структур и проведено изучение свойств их графов алгоритма, что позволяет делать выводы об особенностях их параллельной реализации.

Например, в табл. 1 представлена вычислительная мощность алгоритма, которая равна общему числу операций, поделенному на суммарный объем входных и выходных данных. Анализ ее значений для различных типовых структур помогает сделать выводы относительно качества будущих реализаций рассматриваемых структур на вычислительных системах с распределенной памятью. Из таблицы видно, что только в случае перемножения матриц вычислительная мощность зависит от размера входных данных (во всех структурах размер матрицы равен NxN), что указывает на потенциально хорошую масштабируемость реализации алгоритма на бóльшее число процессов или потоков при увеличении размера задачи.

Табл. 1. Пример значений характеристик алгоритма

	Перемножение матриц	Разностная схема «крест»	Обратный ход метода Гаусса	Умножение матрицы на вектор
Вычислительная мощность	2N/3	~1	~2	~2
Параллельная сложность алгоритма	2N	2N-1	2N	2N

На уровне работы с памятью построена сводная таблица с указанием временно́й и пространственной локальности для исследуемых типовых алгоритмических структур. Анализ таблицы показывает, что даже использование только общих характеристик позволяет получать полезные результаты, соответствующие реальным соотношениям локальности между рассматриваемыми типовыми структурами.

Подобные результаты можно использовать в качестве удобного инструмента для проведения сравнительного анализа. Взяв за основу значения, например, для хорошо изученного теста Linpack, про ко-

торый известно, что он обладает хорошей локальностью, можно делать выводы относительно локальности исследуемых структур.

Использование общих характеристик помогает получить качественную оценку. Для детальной информации необходимо использовать тонкие характеристики. На примере двух типовых структур – разностной схемы «крест» и метода Якоби – показано, какие выводы можно получить при исследовании тонких характеристик работы с памятью. Без использования исходного текста программы, при помощи анализа исключительно значений характеристик и входных данных для рассматриваемых структур, было составлено практически полное описание строения потока и сделаны выводы относительно наличия и свойств циклов.

Далее рассмотрены некоторые типы потоков обращений, которые встречаются как в большинстве рассматриваемых типовых структур, так и во множестве реальных задач. Приведена структуризация потоков по нескольким типам: от самых простых, представляющих собой последовательный перебор элементов массива, как, например, при использовании массивов косвенной адресации; до более сложных, представляющих собой композицию нескольких потоков, например, поток теста Linpack.

Для описанных типов потоков указывается, какие выводы об эффективности работы с памятью можно сделать в каждом случае, и приводятся примеры потоков обращений, которые соответствуют рассматриваемому типу.

Результаты анализа характеристик могут быть применены не только для изучения свойств процесса отображения. Например, были проведены эксперименты с известным тестом RandomAccess, который выполняет операции чтения/записи элементов массива, выбираемых в произвольном порядке. Были использованы две системы на основе процессоров Xeon 5130 и Xeon 5160, отличие которых заключается только разной тактовой частоте – 2 ГГц у Xeon 5130 против 3 ГГц у Xeon 5160. Подсистемы памяти в обеих системах были устроены одинаково. Эксперименты показали, что время выполнения теста на обеих системах практически совпадает (табл. 2), и это говорит о том, что в этом тесте узким местом является работа с памятью, которая происходит очень неэффективно, о чем и сигнализировали значения характеристик. Таким образом, если для некоторого приложения необходимо выбрать подходящую вычислительную платформу, а анализ характеристик показывает, что узким местом этого приложения, скорее всего, будет работа с памятью, то стоит выбирать не более мощный процессор, а более быструю память.

Табл. 2. Результаты экспериментов для теста RandomAccess

	Intel Xeon 5130	Intel Xeon 5160
Тактовая частота	2 ГГц	3 ГГц
Время, сек, $N=2^{26}$ элементов	13.33	13.01
Время, сек, $N=2^{27}$ элементов	35.67	35.50

Все представленные в работе результаты вычислительных экспериментов получены с помощью разработанных автором инструментальных средств. Данные средства предназначены как для определения детальной структуры потока, так и для его визуализации.

В **заключении** сформулированы основные результаты работы:

1. Предложен и апробирован подход к структуризации и решению задачи эффективного отображения алгоритмов на вычислительные системы, опирающийся на совместный анализ свойств и характеристик трех основных объектов: алгоритмов, программ и архитектур, рассматриваемых с точки зрения потенциала параллелизма и степени локальности данных.
2. Предложен метод исследования степени локальности данных в программах с помощью набора формальных характеристик, отражающих иерархическую структуру памяти современных компьютеров. Это позволяет оценивать как интегральные свойства локальности использования данных в целом, так и проводить детальный анализ эффективности работы с памятью на уровне отдельных фрагментов.
3. Определены характеристики множества реальных типовых алгоритмических структур. На основе большого числа вычислительных экспериментов показана зависимость эффективности выполнения программ от степени несоответствия свойств программ особенностям архитектуры. Показана важность исследования профиля обращений программ в память, предложены методы и технологии экспериментального и визуального анализа профиля реальных программ.

Основные результаты диссертации опубликованы в следующих изданиях.

Публикации в журналах из Перечня ВАК:

1. Воеводин Вад. В. Характеристики типовых алгоритмических структур // Вестник Нижегородского университета, ч. 1, вып. 2. 2011. С. 181-189.
2. Воеводин Вад. В. Визуализация и анализ профиля обращений в память // Вестник Южно-Уральского университета (серия Математическое моделирование и программирование), №17(234), вып. 8. 2011. С. 76-84.
3. Адинец А.В., Брызгалов П.А., Воеводин Вад.В., Жуматий С.А., Никитенко Д.А. Мониторинг, анализ и визуализация потока заданий на кластерной системе // Вычислительные методы и программирование. Т. 12. 2011. С. 90-93.

Публикации в других научных изданиях:

4. Воеводин Вад. В. Организация работы с памятью в современных процессорах и эффективность выполнения программ // Третья Международная научная конференция "Суперкомпьютерные системы и их применение" (SSA'2010): доклады конференции. Минск: ОИПИ НАН Беларуси. 2010. С. 171-176.
5. Воеводин Вад. В. Характеристики работы с памятью и эффективность работы программ // Высокопроизводительные параллельные вычисления на кластерных системах (HPC-2010): материалы X Международной конференции. Пермь, 2010. С. 114-118.