На правах рукописи

ЦЫМБЛЕР Михаил Леонидович

МЕТОДЫ ПОСТРОЕНИЯ ПРОГРАММНОГО КОМПЛЕКСА ДЛЯ УПРАВЛЕНИЯ ДАННЫМИ В ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ С МАССОВЫМ ПАРАЛЛЕЛИЗМОМ

Специальность 05.13.18 – теоретические основы математического моделирования, численные методы и комплексы программ

Автореферат диссертации на соискание ученой степени кандидата физико-математических наук

Челябинск-2000

Работа выполнена на кафедре математического обеспечения ЭВМ Челябинского государственного университета.

Научный руководитель:
кандидат физико-математических наук, доцент СОКОЛИНСКИЙ Леонид Борисович

Официальные оппоненты:
доктор физико-математических наук, профессор ВОЕВОДИН Владимир Валентинович
кандидат технических наук, доцент САМОФАЛОВ Виктор Владимирович

Ведущая организация:
Уральский государственный университет

Защита состоится "____" _____ 2000 г. в ____ часов
на заседании диссертационного совета Д 064.19.03 по присуждению ученой степени кандидата наук при Челябинском государственном университете
по адресу 454021 г. Челябинск ул. Братьев Кашириных, 129.

С диссертацией можно ознакомиться в библиотеке Челябинского государственного университета.

Автореферат разослан "____" _____ 2000 г.

Ученый секретарь
диссертационного совета,
д-р физ.-мат. наук В.И. Ухоботов

Общая характеристика работы

Актуальность темы. Одним из актуальных направлений системного программирования является создание системного программного обеспечения для современных многопроцессорных вычислительных комплексов с массивно-параллельной архитектурой.

Одной из основных сфер применения вычислительных систем с массовым параллелизмом в настоящее время являются фундаментальные научные и прикладные задачи, эффективное решение которых возможно только с использованием мощных вычислительных ресурсов. Примером могут служить задачи аэродинамики для самолетостроения и создания реактивных двигателей, моделирование управляемого термоядерного синтеза, распознавание изображений при навигации движущихся объектов, системы поддержки принятия решений, предсказание климата и глобальных изменений в земной коре и др. Многие из упомянутых задач требуют обработки больших объемов данных, хранящихся на внешних носителях. Вследствие этого актуальной является тема разработки методов построения программных комплексов для управления данными в системах с массовым параллелизмом.

В настоящее время одной из перспективных отечественных разработок в сфере многопроцессорных вычислительных систем является многопроцессорный вычислительный комплекс МВС-100/1000, имеющий массивно-параллельную архитектуру, и используемый в ряде академических институтов и университетов России для решения широкого спектра фундаментальных научных и прикладных задач. Вследствие этого актуальной темой является разработка комплекса системных программ для управления данными в многопроцессорной вычислительной системе МВС-100/1000.

Цель и задачи исследования. Целью данной работы является исследование и разработка методов построения комплекса системных программ для управления данными в вычислительных системах с массовым параллелизмом. Данная цель предполагает решение следующих задач:

- исследование и анализ существующих методов организации хранения и передачи данных в многопроцессорных вычислительных системах с массовым параллелизмом;
- разработка новых методов управления данными в вычислительных системах с массовым параллелизмом, учитывающих особенности архитектуры современных многопроцессорных систем типа МВС-100/1000;
- разработка, реализация и отладка программного комплекса для организации хранения и передачи данных в многопроцессорной вычислительной системе МВС-100/1000.

Методы исследования. Проведенные в работе исследования базируются на использовании методов модульного и объектно-ориентированного программирования, теории математического моделирования и аппарата баз данных.

Научная новизна работы заключается в следующем:

- предложен новый подход к организации систем управления данными для массивно-параллельных платформ, основанный на введении трех уровней иерархии: аппаратного, физического и логического;
- разработан новый механизм буферизации страниц, базирующийся на введении статического и динамического рейтингов страниц и использующий избыточный индекс буферного пула;

- на основе предложенных подходов разработана и реализована система управления данными для многопроцессорного вычислительного комплекса МВС-100, не имеющая в настоящее время отечественных аналогов.

Практическая ценность работы. Многопроцессорные вычислительные комплексы семейства МВС-100/1000 являются в настоящее время одной из перспективных отечественных разработок в сфере вычислительных систем с массовым параллелизмом. Системы МВС-100/1000 эксплуатируются в ряде академических институтов и университетов России. Данные системы способны показывать производительность, сравнимую с лучшими зарубежными суперкомпьютерами, оставаясь существенно более дешевыми по сравнению с импортными аналогами. Поэтому разработанные методы построения программных комплексов для управления данными в системах с массовым параллелизмом и реализованный на их основе программный комплекс для многопроцессорной вычислительной системы МВС-100/1000 имеют большую практическую ценность.

Разработанный программный комплекс внедрен в опытную эксплуатацию в ряде российских организаций. Полученные в диссертационной работе результаты структурированы и включены в Web-репозиторий проекта Омега, доступный через Internet по адресу http://www.csu.ru/~sok/OmegaProject.

Данная работа выполнялась при финансовой поддержке Российского фонда фундаментальных исследований (проекты 97-07-90148, 00-07-90077).

Апробация работы. Основные результаты диссертационной работы неоднократно докладывались автором на международных и всероссийских научных конференциях и семинарах, в том числе:

- на Всероссийской научной конференции "Высокопроизводительные вычисления и их приложения" (Черноголовка, 30 октября – 2 ноября 2000 г.)
- на Международной конференции по программированию и информационным технологиям CSIT'2000 (Уфа, 18-23 сентября 2000 г.)
- на Международной конференции "Распределенные системы: оптимизация и приложения в экономике и науках об окружающей среде" DSO'2000 (Екатеринбург, 30 мая – 2 июня 2000 г.);
- на Всероссийской научной конференции "Научный сервис в сети Internet" (Новороссийск, 20-25 сентября 1999 г.);
- на XI-й Всероссийской конференции "Математическое программирование и приложения" (Екатеринбург, 22-26 февраля 1999 г.);
- на Международной конференции по программированию и информационным технологиям CSIT'99 (Москва, 18-22 января 1999 г.)

Публикации. Основные результаты диссертации представлены в *восьми* опубликованных работах.

Структура и объем работы. Диссертационная работа состоит из введения, шести глав, заключения и списка литературы.

Содержание работы

Во введении приводится обоснование актуальности темы, формулируются цели работы и кратко излагается содержание диссертации.

В первой главе, "Анализ архитектурных особенностей вычислительных систем с массовым параллелизмом", приводится классификация архитектур многопроцессорных вычислительных систем, дается описание архитектуры многопроцессорной вычислительной системы МВС-100/1000 и предлагается концепция трехуровневой иерархической организации систем управления данными для многопроцессорных вычислительных систем с массовым параллелизмом.

Массивно-параллельная вычислительная система строится из процессорных модулей, которые соединяются в систему с помощью высокоскоростной соединительной сети. Процессорный массив объединяется с внешней дисковой памятью и устройствами ввода-вывода и работает под общим управлением host-машины (персонального компьютера или рабочей станции).

В классическом случае процессорный модуль представляет собой вычислительный процессор с приватной памятью. В этом случае процессор, помимо вычислительной нагрузки и нагрузки по вводу-выводу данных, занимается обеспечением транзитных передач данных по соединительной сети. При большом количестве процессоров нагрузка по обеспечению транзитных передач данных становится доминирующей, и большая часть процессорного времени тратится на передачу "чужих" данных. Это может существенно снизить эффективность использования многопроцессорной системы. Возможным решением данной проблемы являются массивно-параллельные системы с архитектурой, предполагающей включение в процессорный модуль дополнительного (коммуникационного) процессора, основной задачей которого является организация межпроцессорных обменов. Системы с подобной организацией способны обеспечить более высокую степень масштабируемости по сравнению с классической массивно-параллельной архитектурой.

Указанный подход применен в архитектуре отечественных многопроцессорных систем семейства *МВС-100/1000*. МВС строится по модульному принципу. Процессорный модуль МВС имеет архитектуру с разделяемой памятью и включает в себя вычислительный процессор и коммуникационный процессор. Взаимодействие вычислительного и коммуникационного процессоров осуществляется через общую оперативную память. Коммуникационный процессор оснащен высокоскоростными внешними каналами (линками) для соединения с другими процессорными модулями. Вычислительная система может объединять в себе до тысячи процессорных модулей.

Анализ задач, решаемых на МВС-100/1000, показывает, что имеется большой класс задач, связанных с обработкой больших объемов данных, при распараллеливании которых обычно применяется следующий подход. Вычислительная задача разбивается на подзадачи, и для решения подзадач в процессорном массиве системы выделяются непересекающиеся группы процессорных модулей, называемых полями. Подзадача, в свою очередь, разбивается на процессы, и каждый процесс запускается на отдельном процессорном модуле поля.

При реализации задач указанного класса прикладные программисты, выполняя разбиение множества процессоров, доступных задаче, на поля, и назначая роли процессоров внутри поля, как правило, не учитывают физическую топологию системы. При этом для реализации межпроцессорных коммуникаций прикладные программисты используют системные сервисы общего назначения, которые часто оказываются неадекватными и неэффективными для небольших полей с сильносвязной топологией. В связи с этим возникает необходимость в разработке комплекса системных программ, который мог бы использоваться прикладными программистами для эффективного управления данными при реализации задач указанного класса.

В диссертационной работе предлагается подход к организации управления данными в массивно-параллельной системе, который позволяет существенно уменьшить

расходы, связанные с транзитными передачами данных по соединительной сети. Подход предполагает наложение ограничений на топологию процессорного поля. Унификация топологии процессорных полей в пределах одной системы и наложение ограничений на максимальную длину кратчайшего пути между процессорными узлами внутри поля делает возможным разработку более эффективных протоколов обмена данными между узлами процессорного поля.

В соответствии с данным подходом в структуре системы управления данными вводятся три уровня иерархии: аппаратный, физический и логический.

Аппаратный уровень представлен несимметричным процессорным модулем, который состоит из коммуникационного и вычислительного процессоров с разделяемой памятью. Обмен данными между коммуникационным и вычислительным процессорами реализуется аппаратно и микропрограммно.

На *физическом уровне* система представляет собой множество процессорных узлов, объединенных высокоскоростной соединительной сетью. Обмен данными на этом уровне реализуется на основе сервисов, предоставляемых операционной системой типа ОС Router.

На *логическом уровне* множество процессорных узлов системы разбивается на непересекающиеся группы процессоров, называемых кластерами. *Процессорный кластер* состоит из фиксированного числа процессорных узлов, между которыми предполагается интенсивный обмен данными. Обмены данными между узлами разных процессорных кластеров полагаются имеющими относительно низкую интенсивность. Таким образом, на логическом уровне система представляет собой множество процессорных кластеров, объединенных высокоскоростной соединительной сетью. На топологию межпроцессорных соединений кластера могут накладываться определенные ограничения, что позволяет реализовать межпроцессорные обмены внутри кластера более эффективно, чем с помощью сервисов, предоставляемых операционной системой типа ОС Router для МВС-100. Обмен данными на логическом уровне реализуется с помощью двух различных подсистем, одна из которых обеспечивает только внутрикластерные обмены, в то время как другая поддерживает обмены данными между кластерами.

Во второй главе, "Методы организации хранения и передачи данных в многопроцессорных вычислительных системах с массовым параллелизмом", дается классификация данных, обрабатываемых в вычислительной системе с массивно-параллельной архитектурой, в зависимости от объема, времени их жизни и частоты обращения к ним, и предлагаются методы организации управления данными, учитывающие специфику архитектуры современных вычислительных систем с массовым параллелизмом.

Данные могут быть разделены на следующие три категории: персистентные данные, временные данные и сообщения.

Персистентные данные имеют большой объем (не помещаются в оперативную память одного процессорного узла), характеризуются многократными к ним обращениями и существуют до и после выполнения задачи.

Временные данные имеют большой объем (не помещаются в оперативную память процессорного узла), характеризуются многократными к ним обращениями и существуют только в период выполнения задачи.

Сообщения имеют небольшой объем (помещаются в оперативную память процессорного узла), характеризуются однократными к ним обращениями и существуют только в период выполнения задачи.

В соответствии с этим комплекс системных программ для управления данными в вычислительной системе с массовым параллелизмом должен включать в себя систему передачи сообщений и систему хранения персистентных и временных данных.

Система передачи сообщений строится из двух подсистем: модуля межкластерного обмена сообщениями и модуля внутрикластерного обмена сообщениями. *Модуль межкластерного обмена сообщениями* обеспечивает передачу сообщений между узлами, принадлежащими различным процессорным кластерам. *Модуль внутрикластерного обмена сообщениями* обеспечивает средства для передачи сообщений в пределах одного процессорного кластера.

Система хранения персистентных и временных данных обеспечивает унифицированный интерфейс обработки персистентных и временных данных. При этом могут использоваться следующие устройства хранения данных или их комбинации: жесткий диск host-компьютера, дисковая подсистема вычислительной системы и электронная дисковая подсистема, реализуемая на базе стандартного процессорного модуля путем использования его оперативной памяти в качестве хранилища данных.

Система хранения данных строится на базе архитектуры клиент-сервер. В соответствии с этим среди процессорных узлов выделяются *узлы-клиенты* и *узлы-серверы*. Серверная часть системы хранения запускается на узле-сервере и обслуживает запросы клиентских частей, запускаемых на узлах-клиентах.

Система хранения предполагает две реализации узла-сервера: аппаратную и программную. *Аппаратная реализация узла-сервера* строится на базе модуля дисковой подсистемы, встраиваемой в процессорный кластер. *Программная реализация узла-сервера* строится на базе стандартного процессорного модуля, оперативная память которого используется в качестве хранилища данных.

В третьей главе, "Структура программного комплекса Омега для управления данными в многопроцессорной вычислительной системе МВС-100", приводится структура комплекса, и дается описание и принципы реализации его компонент.

В соответствии с подходами, предложенными во второй главе, нами был разработан комплекс системных программ для управления данными в многопроцессорной вычислительной системе МВС-100, получивший название Омега. Программный комплекс Омега включает в себя модуль топологии, менеджер нитей, систему передачи сообщений и систему хранения данных.

Модуль топологии инкапсулирует аппаратные особенности топологии МВС-100. В модуле топологии определяется общее число процессорных узлов, доступных задаче, число процессорных кластеров, число узлов в кластере и местоположение модуля дисковой подсистемы.

Менеджер нитей (легковесных процессов) обеспечивает возможность организации параллельных нитей управления на базе модели "производитель-потребитель". В данной модели процессы ассоциируются с потоками данных и являются средством структуризации программы. Данная модель позволяет обеспечить автоматическую синхронизацию и эффективную диспетчеризацию параллельных процессов.

Система передачи сообщений обеспечивает средства асинхронного обмена данными между любыми двумя процессорными узлами вычислительной системы. В соответствии с иерархической организацией системы Омега система передачи сообщений состоит из двух подсистем: маршрутизатора и кондуктора.

Маршрутизатор обеспечивает асинхронную передачу сообщений между узлами, принадлежащими различным процессорным кластерам. Маршрутизатор поддерживает любое количество асинхронных виртуальных каналов (для каждого обмена создается свой канал) между любыми двумя процессорными узлами. Для идентификации

каналов используется понятие порта: канал, соединяющий два узла, имеет одинаковые номера портов с обеих сторон. Протокол обмена сообщениями маршрутизатора характеризуется тем, что инициализация передачи сообщения от одного узла другому требует только двух элементарных обменов (элементарный обмен примерно эквивалентен передаче одного байта информации). В реализации передачи сообщений посредством каналов используется копирование данных типа "память-в-память" и динамическое выделение памяти. Однако это не сказывается на производительности маршрутизатора критическим образом, так как межкластерные обмены полагаются относительно редкими.

Кондуктор обеспечивает средства для асинхронной передачи сообщений в пределах одного процессорного кластера. Интерфейс кондуктора близок к интерфейсу маршрутизатора, однако кондуктор использует принципиально иной внутренний протокол. В соответствии с этим протоколом инициализация передачи сообщения от одного узла другому требует трех элементарных обменов, и при передаче сообщения не использует копирование данных типа "память-в-память". Данный протокол оказывается достаточно эффективным для внутрикластерных обменов сообщениями, поскольку длина кратчайшего пути между узлами мала (например, не больше двух), что обеспечивает относительно низкую стоимость одного элементарного обмена. Другой особенностью реализации кондуктора является то, что при создании нового виртуального канала не требуется динамического выделения памяти.

Система хранения данных обеспечивает унифицированный интерфейс обработки персистентных и временных данных на базе следующих устройств хранения: жесткий диск host-компьютера, дисковая подсистема вычислительной системы и электронная дисковая подсистема.

В четвертой главе, "Методы реализации системы хранения данных", приводится структура системы хранения данных, дается описание и принципы реализации ее компонент.

Система хранения данных включает в себя электронную дисковую подсистему и систему управления файлами.

Электронная дисковая подсистема (ЭДП) реализует виртуальный модуль дисковой подсистемы и предоставляет соответствующие низкоуровневые сервисы для чтения-записи данных. ЭДП может быть установлена на любом специально выделенном узле процессорного кластера.

Система управления файлами (СУФ) поддерживает понятие файла, как именованного набора записей фиксированной длины. Файлы используются для унифицированного представления и обработки персистентных и временных данных. СУФ представлена иерархией следующих подсистем: менеджер дисков, менеджер наборов и менеджер файлов.

Менеджер дисков обеспечивает страничную организацию диска и предоставляет средства для асинхронного чтения и записи страниц диска. Менеджер дисков состоит из клиентской и серверной части. Клиенты и сервер обмениваются сообщениями, которые состоят из двух частей: заголовок и информационная часть info. Заголовок сообщения – это запись с фиксированной структурой. Информационная часть не имеет структуры; это байтовый массив, длина которого совпадает с размером страницы диска. Передача сообщения выполняется в два этапа: сначала передается заголовок, а затем – info. Часть info может отсутствовать (не передаваться). Приводятся специально разработанные протоколы обмена данными между клиентами и сервером менеджера дисков.

Менеджер наборов обеспечивает представление данных, хранящихся на диске, в виде совокупности наборов (связных списков) страниц. Менеджер наборов позволяет

создавать и удалять наборы страниц, добавлять страницы в набор, удалять страницы из набора, осуществлять последовательный просмотр набора страниц, а также прямую выборку и выборку с упреждением страницы с указанным номером. Менеджер наборов обеспечивает буферизацию данных на основе единого буферного пула. Методы управления буферным пулом, применяемые в системе Омега, подробно рассматриваются в главе 5.

Менеджер файлов обеспечивает представление данных в виде файлов – именованных множеств неструктурированных записей одинаковой длины (запись имеет только одно информационное поле info). Менеджер файлов обеспечивает стандартные функции для управления данными: создание, удаление, открытие и закрытие файла, выборка, добавление и удаление записей, организация последовательного просмотра записей файла.

Клиент менеджера дисков, менеджер наборов и менеджер файлов реализуют клиентскую часть системы хранения данных. Серверная часть системы хранения реализуется в виде сервера менеджера дисков и драйвера реальной либо электронной дисковой подсистемы. Подобный подход обеспечивает мобильность комплекса Омега при переходе с платформы МВС-100 на платформу МВС-1000 и другие массивно-параллельные платформы.

В пятой главе, "Методы управления буферным пулом в системе хранения данных", анализируется проблематика буферизации данных, описывается оригинальный метод вытеснения страниц из буферного пула, и приводятся результаты численных экспериментов, исследующих эффективность предложенного подхода.

При интенсивных обменах данными между узлами-клиентами и узлами-серверами пропускная способность линков становится узким местом. Данная проблема решается с помощью буферизации данных. *Буферизация* заключается в выделении буферного пула в основной памяти узла-клиента. Если запрашиваемая страница данных уже находится в буфере клиента (ситуация *попадания*), то повторного считывания страницы не происходит, что позволяет значительно сократить объем данных, передаваемых от сервера к клиенту.

При обработке данных большого объема, как правило, не удается обеспечить буферный пул, вмещающий весь массив данных, необходимых задаче. В этом случае приходится *вытеснять* из буферного пула страницы, которые в данный момент не используются. При использовании механизма вытеснения страниц возможна ситуация *неудачи*, когда повторно запрашиваемая страница данных отсутствует в буфере. Отсюда возникает проблема подбора эффективной *стратегии вытеснения страниц*, которая минимизировала бы число неудач.

В данной главе приводится обзор известных стратегий вытеснения (общие стратегии LRU, LFU, FIFO, LIFO и их модификации). С использованием общих стратегий вытеснения связан ряд проблем. Первая из них заключается в том, что стратегия вытеснения может показывать в среднем удовлетворительную производительность, но быть наихудшей в отдельных случаях доступа к диску. Например, стратегия LRU (Least Recently Used), вытесняющая страницу, к которой дольше всего не было обращений, и широко применяющаяся в реализации операционных систем, является наихудшей для случая последовательного чтения страниц диска, повторяющегося в цикле. Между тем такой случай доступа к данным диска является типичным при выполнении запросов в реляционных базах данных.

Вторая проблема связана с тем, что общие стратегии вытеснения могут быть неадекватными в ряде случаев, поскольку при выборе жертвы руководствуются только одним определенным принципом и не поддерживают избирательное вытеснение стра-

ниц. Примером может быть порядок вытеснения страниц для обеспечения корректного восстановления базы данных после сбоя. Страницы данных журнала транзакций, содержащие флаги "зафиксировано", должны вытесняться на диск строго после страниц, которые были зафиксированы в ходе транзакций – безотносительно к используемой стратегии вытеснения.

При реализации системы Омега нами был использован оригинальный метод вытеснения страниц, получивший название DIR-метода. *DIR-метод* базируется на двух концепциях: введение для образов страниц статического и динамического рейтингов и использование избыточного индекса буферного пула. Механизм статического рейтинга позволяет реализовать избирательное вытеснение страниц. Механизм динамического рейтинга позволяет моделировать различные стратегии вытеснения страниц. Использование избыточного индекса буферного пула позволяет сохранять историю обращений к страницам, анализ которой дает возможность динамического выбора адекватной стратегии вытеснения страниц.

Статический рейтинг – это целое число от 0 до 20. Статический рейтинг является атрибутом *открытого* набора страниц и определяется пользователем при выполнении операции открытия набора. Статический рейтинг страницы остается неизменным, пока набор открыт. При закрытии набора значение статического рейтинга его страниц теряется.

Динамический рейтинг – это некоторая функция от статического рейтинга, принимающая значения в интервале [0;1[. Значение динамического рейтинга может меняться во время нахождения страницы в буфере. Способ вычисления динамического рейтинга задается системой.

Суммарный рейтинг страницы получается как сумма статического и динамического рейтингов. Если необходимо освободить место в буферном пуле, то вытесняется страница, имеющая минимальный суммарный рейтинг. Если таких страниц несколько, то вытесняется страница, дольше всего находящаяся в буферном пуле.

Избыточный индекс буферного пула (DIR) представляет собой таблицу в оперативной памяти, в которой, помимо указателей на образы страниц, находящихся в данный момент в буфере, хранится статистическая информация об использовании этих страниц. Избыточность DIR выражается в том, что после вытеснения страницы из буфера соответствующий элемент в DIR сохраняется. Это позволяет накапливать статистику попаданий и неудач, которая используется для предсказания последующих обращений к страницам диска. Длина DIR определяется как kM, где M – длина буферного пула в страницах, целое $k>1$; k называется кратностью DIR и его точное значение устанавливается экспериментально. Теоретическим максимумом длины DIR является количество страниц на диске.

Схема алгоритма операции выборки страницы, использующего DIR, выглядит следующим образом. Если дескриптор страницы отсутствует в DIR, то нужно вставить новый элемент в DIR. При этом типична ситуация, когда в DIR все позиции уже заняты. Тогда определяется "жертва" DIR, и на ее место записывается новый элемент. Если образ страницы есть в буфере, то алгоритм заканчивает работу. В противном случае ищется свободный блок в буферном пуле. Если свободный блок есть, то в него загружается с диска соответствующая страница, и алгоритм заканчивает работу. Если свободных блоков в буфере нет, то по суммарному рейтингу определяется страница-жертва, которая вытесняется на диск, и на ее место загружается требуемая страница.

Следует отметить, что в данной схеме вытеснение применяется не только к страницам, образы которых в данный момент находятся в буфере, но и к страницам, образы которых отсутствуют в буфере. Причем, для этого могут использоваться различные стратегии вытеснения.

В DIR хранится статистика использования страниц, которые находились или находятся в буфере. Примерами основных статистических атрибутов элемента DIR являются счетчик попаданий *HC (hit counter)*, время последнего попадания *HT (hit time)*, счетчик неудач *FC (fault counter)* и время последней неудачи *FT (fault time)*. Указанные статистические атрибуты позволяют моделировать с использованием DIR-метода практически любую общую стратегию вытеснения.

На базе DIR-метода были разработаны несколько оригинальных стратегий, две из которых приведены в Таблице 1 (здесь *NORM* означает функцию нормирования, приводящую значение к диапазону [0;1[).

Таблица 1. Примеры стратегий вытеснения, использующих DIR

Стратегия вытеснения	Подсчет динамического рейтинга страницы
DIR_{FC}	NORM(*HC+FC/k*)
DIR_{FT}	NORM(*HT+FT/k*)

Одним из факторов, влияющих на эффективность системы хранения данных, является ее способность обнаруживать циклы подкачки страниц. Затраты времени на обнаружение таких циклов всегда несравнимо меньше времени, которое будет затрачено на вытеснение страниц, составляющих цикл. При достаточных размерах DIR путем анализа истории загрузки страниц в буфер могут быть обнаружены циклы подкачки страниц. Эти циклы могут быть оптимизированы с помощью механизма динамического рейтинга.

Для исследования эффективности предложенного метода буферизации страниц были проведены *численные эксперименты*. При проведении экспериментов ставились следующие цели:

- сравнить эффективность общих стратегий вытеснения страниц и стратегий, использующих DIR;
- сравнить эффективность различных стратегий, использующих DIR, при одинаковом значении кратности длины DIR;
- определить оптимальное значение кратности длины DIR.

Была разработана программа, моделирующая последовательность L случайных обращений к страницам диска 1,…,N с Зипфовым распределением частот обращений (то есть вероятность доступа к странице с номером, не превосходящем i, равна $(i/N)^{\log a/\log b}$, где a=0.8 и b=0.2). В качестве L и N брались значения 3000 и 1000 соответственно.

Для сравнения эффективности были взяты стратегия LRU (которая среди общих стратегий считается лучшей) и стратегия DIR_{FC}. Для буферного пула небольшого размера (10-30 страниц) эффективность DIR_{FC} превосходит эффективность LRU на 15%. Затем это превосходство снижается до 10% при размере буферного пула 30-100 страниц и 5% при буфере в 100-200 страниц. При большем размере буфера обе стратегии показывают практически одинаковую эффективность (ожидаемый результат, поскольку при больших размерах буфера вытеснение страниц практически отсутствует).

Для сравнения стратегий, использующих DIR, были взяты две стратегии: DIR_{FC} и DIR_{FT}. Эффективность стратегии DIR_{FC} для буферного пула размером 10-60 страниц в среднем выше, чем у DIR_{FT}, на 10%. При большем размере буфера обе стратегии показывают практически одинаковую эффективность.

Эксперименты, в которых варьировалось значение кратности длины DIR, показали, что наибольшей эффективности стратегий можно достичь при значении кратности k=0.02D, где D – общее число страниц на диске.

В шестой главе, "Технологические аспекты разработки программного комплекса для управления данными", описывается технология коллективной разработки программного комплекса Омега, структура программных средств поддержки коллективной разработки и разработанные компоненты расширения среды программирования (отладчик и профилировщик).

Для организации коллективной разработки подсистем, входящих в программный комплекс Омега, нами была предложена *технология коллективной разработки* программ для МВС-100, включающая в себя концептуальные, организационные и программные средства.

Концептуальные средства коллективной разработки системы Омега составляют репозиторий проекта, библиотека проекта и справочник по функциям проекта. *Репозиторий проекта* хранит все исходные тексты подсистем и все изменения, которые были в них произведены после включения в репозиторий. *Библиотека проекта (Ω-Lib)* хранит объектный код функций подсистем проекта. *Справочник по функциям Ω-Lib* представляет собой гипертекстовый документ в формате HTML, который может просматриваться любым WWW-обозревателем. Репозиторий проекта, библиотеку проекта и Справочник по функциям Ω-Lib поддерживает специально выделенный разработчик – *технолог проекта*.

Организация технологического цикла коллективной разработки выглядит следующим образом. Для обращения к функциям подсистем библиотеки программист получает из репозитория тексты соответствующих заголовочных файлов. Затем программист строит загрузочные модули тестов разрабатываемой им подсистемы, при этом в качестве параметра компиляции указывается библиотека Ω-Lib. Далее загрузочные модули тестов выполняются на МВС. По полученным отладочным дампам анализируются и локализуются ошибки. В исходные тексты вносятся исправления, и цикл повторяется. После того, как все тесты новой подсистемы прошли без ошибок, ее исходные тексты передаются технологу проекта. При этом данные тексты должны содержать спецификации функций подсистемы, пригодные для включения в Справочник по функциям Ω-Lib.

Технолог выполняет следующую стандартную последовательность действий. Он помещает полученные исходные тексты подсистемы в свою рабочую копию проекта. После этого технолог собирает и запускает комплексный тест системы. Если при выполнении теста возникли ошибки, подсистема возвращается программисту на доработку. Если комплексный тест прошел нормально, то технолог помещает исходные тексты подсистемы в репозиторий. Затем создается новый вариант библиотеки Ω-Lib, включающий в себя функции подсистемы, полученной от разработчика. В завершении цикла генерируются HTML страницы, содержащие справочную информацию по функциям новой подсистемы, которые добавляются в Справочник по функциям Ω-Lib. Если при дальнейшей разработке проекта в подсистеме обнаруживается ошибка, то программист копирует в свой рабочий каталог библиотеку Ω-Lib и удаляет из этой копии библиотеки все модули своей подсистемы, получая библиотеку Ω'-Lib. После этого программист возобновляет работу над своей подсистемой, только теперь в технологическом цикле вместо Ω-Lib он использует Ω'-Lib.

Программные средства технологии разработки комплекса Омега делятся на две группы: средства поддержки коллективной разработки и средства расширения среды программирования для МВС-100.

Средства поддержки коллективной разработки обеспечивают ведение репозитория, библиотеки проекта и справочника по функциям проекта. Репозиторий поддерживается с помощью специальной системы контроля версий. Библиотека проекта поддерживается с помощью компилятора и программы-библиотекаря. Для генерации спра-

вочника по функциям используется документатор исходных текстов программ. Документатор анализирует комментарии специального вида и переводит их в формат HTML. В качестве программных средств поддержки коллективной разработки были использованы система контроля версий CVS, документатор DOC++ и компилятор C фирмы Portland Group (PGCC). Все три программных продукта являются разработками независимых фирм.

Средства расширения среды программирования предоставляют инструментарий для отладки и профилирования программ, отсутствующий в среде PGCC для МВС-100. Наличие средств отладки и профилирования программ является одним из важных факторов обеспечения эффективной разработки программ для многопроцессорных систем. Для обеспечения данных возможностей были разработаны отладчик и профилировщик.

Отладчик выполняет трассировку программы и отладочную печать. Отладчик позволяет задавать номера процессоров, для которых следует выводить отладочные сообщения. В отладочные сообщения автоматически могут включаться номера процессоров. Отладчик поддерживает два режима работы: режим выдачи отладочной информации в виде дампа и режим трассировки, обеспечивающий пошаговое выполнение программы.

Профилировщик позволяет учитывать общий объем данных, вовлекаемых в обмен с диском и в передачу по линкам. Профилировщик использует концепцию профилировочных тегов. *Профилировочный тег* является некоторым идентификатором, позволяющим пользователю суммировать результаты профилирования, полученные при выполнении одной задачи на нескольких процессорных узлах. Каждый обмен с диском и каждая передача сообщения, выполняемая некоторой нитью, увеличивают соответствующие счетчики всех своих профилировочных записей, для которых включен режим профилирования.

Предложенное расширение среды программирования PGCC путем добавления отладчика и профилировщика позволяет существенно повысить эффективность процесса отладки приложений в многопроцессорной среде, и обеспечить возможность профилирования программы с целью оптимизации передачи сообщений по линкам и обменов с дисками.

В заключении перечислены основные результаты работы.

Основные результаты

1. Предложена методика построения программного комплекса для хранения и передачи данных в вычислительных системах с массовым параллелизмом, учитывающая особенности архитектуры современных многопроцессорных систем.
2. Разработана и реализована система управления файлами для многопроцессорного вычислительного комплекса МВС-100, включающая в себя следующие подсистемы:
   - менеджер дисков, поддерживающий страничное представление диска и обеспечивающий операции чтения-записи данных на основе эффективных протоколов взаимодействия с модулем дисковой подсистемы;
   - менеджер наборов, поддерживающий представление данных в виде совокупности наборов (связных списков) страниц и обеспечивающий буферизацию данных на основе единого буферного пула;
   - менеджер файлов, поддерживающий представление данных в виде файлов (именованных множеств неструктурированных записей одинаковой длины) и обеспечивающий стандартные средства для управления данными;

- электронную дисковую подсистему, поддерживающую виртуальный модуль дисковой подсистемы и обеспечивающую унифицированный интерфейс для работы как с виртуальной, так и с реальной дисковой подсистемой.

3. Разработан метод вытеснения страниц из буферного пула, получивший название DIR-метода. DIR-метод базируется на введении статического и динамического рейтингов страниц и использовании избыточного индекса буферного пула (DIR). DIR-метод позволяет:
   - моделировать практически любую общую стратегию вытеснения страниц;
   - назначать и динамически изменять стратегию вытеснения для отдельных наборов страниц;
   - обнаруживать и оптимизировать циклы подкачки страниц.
4. Предложена методика построения эффективных стратегий вытеснения страниц, базирующихся на DIR-методе. Проведены численные эксперименты, подтверждающие более высокую эффективность DIR-стратегий по сравнению с классическими.
5. Предложена технология разработки больших программных комплексов для многопроцессорной вычислительной системы МВС-100. Данная технология апробирована при разработке программного комплекса Омега и обеспечивает:
   - поддержку коллективной разработки программных комплексов для МВС-100 на этапах кодирования, отладки, тестирования и сопровождения;
   - среду программирования с инструментарием отладки и профилирования параллельных программ на МВС-100.

Основные результаты диссертации опубликованы в следующих работах:

1. *Zymbler M.L., Sokolinsky L.B.* Implementation Principles of File Management System for Omega Parallel DBMS // Proceedings of the 2nd International Workshop on Computer Science and Information Technologies (CSIT'2000), September 18-23, 2000, Ufa, Russia. Ufa: USATU Publishing. 2000. Vol. 1. P. 173-178.
2. *Соколинский Л.Б., Цымблер М.Л.* Принципы реализации системы управления файлами в параллельной СУБД Омега для МВС-100 // Вестник Челябинского университета. Серия математика, механика. 1999. № 2(5). С. 176-199.
3. *Zymbler M.L.* Computer Aided Design Facilities for Prototyping the Omega DBMS // CSIT'99, Proceedings of the 1st International Workshop on Computer Science and Information Technologies, January 18-22, 1999, Moscow, Russia. Moscow: MEPhI Publishing. 1999. Vol. 2. P. 124-131.
4. *Цымблер М.Л., Соколинский Л.Б.* Организация распределенной обработки больших массивов данных в многопроцессорных системах с массовым параллелизмом // Высокопроизводительные вычисления и их приложения: Труды Всероссийской научной конференции (Черноголовка, 30 октября – 2 ноября 2000 г.). - М.: Изд-во МГУ. 2000. С. 186-190.
5. *Цымблер М.Л., Соколинский Л.Б.* Выбор оптимальной стратегии вытеснения страниц в параллельной СУБД Омега для мультипроцессорной системы МВС-100 // Распределенные системы: оптимизация и приложения в экономике и науках об окружающей среде (DSO'2000). Сборник докладов к Международной конференции (Екатеринбург, 30 мая – 2 июня 2000 г.). Екатеринбург: УрО РАН. 2000. С. 337-340.
6. *Цымблер М.Л., Соколинский Л.Б., Федрушков В.В.* Использование Internet-технологий в коллективной разработке больших программных систем // Научный сервис в сети Интернет: Тезисы докладов Всероссийской научной конфе-

ренции (Новороссийск, 20-25 сентября 1999 г.). - М.: Изд-во МГУ. 1999. С. 207-210.
7. *Соколинский Л.Б., Цымблер М.Л.* Использование МВС-100 в качестве машины баз данных // Информационный бюллетень Ассоциации математического программирования. № 8. Екатеринбург: УрО РАН. 1999. С. 251-252.
8. *Соколинский Л.Б., Цымблер М.Л.* Проект создания параллельной СУБД Омега на базе суперкомпьютера МВС-100/1000 // Телематика'98: Тезисы докладов Всероссийской научно-методической конференции (7-10 июня 1998 г., Санкт-Петербург). - СПб: Вузтелекомцентр.1998. С. 154-155.

Подписано в печать 28.09.2000
Формат 60x84 $^{1}/_{16}$. Бумага офсетная.
Печать офсетная. Усл.печ.л. 0,9. Уч.-изд.л. 1,5.
Тираж 100 экз. Заказ №____. Бесплатно.

Челябинский государственный университет.
454021 Челябинск, ул. Братьев Кашириных, 129

Полиграфический участок Издательского центра ЧелГУ.
454021 Челябинск, ул. Молодогвардейцев, 57б