На правах рукописи

САМАРЕВ Роман Станиславович

Методы и модели проектирования параллельных СУБД

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

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата технических наук

Москва - 2007

Работа выполнена на кафедре «Компьютерные системы и сети» (ИУ-6) факультета «Информатика и системы управления» в Московском Государственном Техническом Университете имени Н.Э. Баумана.

Научный руководитель: доктор технических наук, профессор
Сюзев Владимир Васильевич

Официальные оппоненты: доктор технических наук, профессор
Кузнецов Сергей Дмитриевич

кандидат технических наук
Коротков Сергей Викторович

Ведущая организация: ЗАО «Концерн ВНИИНС»
(Всероссийский научно-исследовательский институт автоматизации управления в непромышленной сфере)

Защита состоится «18» октября 2007 г. в 16 часов 00 минут на заседании диссертационного совета Д 212.141.10 по защите диссертаций при Московском Государственном Техническом Университете имени Н.Э. Баумана по адресу: 107005, г. Москва, 2-я Бауманская ул., д. 5.

С диссертацией можно ознакомиться в библиотеке МГТУ им. Н.Э. Баумана.

Отзыв на автореферат в одном экземпляре, заверенный печатью, просим направлять в адрес совета университета.

Автореферат разослан «__» _____ 2007 г.

Ученый секретарь
диссертационного совета С.Р. Иванов
кандидат технических наук, доцент _____

Общая характеристика работы

Актуальность проблемы. Интенсивное развитие вычислительной техники в последние годы, привело к массовому использованию недорогих многопроцессорных высокопроизводительных вычислительных систем (ВС) архитектуры x86. Тем не менее, применяемые унаследованные (ранее разработанные) СУБД, часто не используют предоставляемые ресурсы ВС, т.к. не рассчитаны на параллельное выполнение одиночных запросов. В итоге, при малой плотности запросов, СУБД использует лишь минимально необходимое количество ресурсов ВС, так что, в конечном счете, добавление новых процессоров или параллельных дисковых накопителей не приводит к какому-либо улучшению её характеристик. Расчет на пиковую нагрузку СУБД приводит к тому, что в среднем ВС будет серьезно недогружена.

Другой проблемой использования СУБД является то, что монопольное использование ВС не всегда возможно. Часто на ВС, на которой работает СУБД, работают и другие программы, также требующие значительных ресурсов. При малой плотности запросов к СУБД, влияние других программ может быть незначительным, однако, например в случае использования СУБД в сочетании с Web-сервером, вполне возможна ситуация полной загрузки ВС. Операционная система, типа MS Windows на ВС с вытесняющей многозадачностью, не способна оптимально разрешить ситуацию одновременного использования ресурсов ВС, типа ЦП, ОЗУ или НЖМД при большом количестве одновременно работающих процессов/потоков. Т.е. линейное увеличение количества параллельных процессов приводит к нелинейному снижению производительности системы в целом, поскольку доля служебных операций по переключению потоков, вытеснению памяти, позиционированию головок НЖМД на нужный сектор занимает значительное время.

В настоящее время широко распространены параллельные СУБД для вычислительных систем без общей памяти, однако, массовое использование архитектуры x86 не позволяет использовать такие параллельные СУБД без изменений. Кластер, построенный путем объединения отдельных x86-систем, можно рассматривать как единую систему, с несколькими независимыми узлами, не имеющими общей памяти, но каждый узел может быть многопроцессорным, однако в этом случае его возможности не используются в полной мере.

Итак, оптимальной СУБД с точки зрения производительности и времени отклика будет СУБД, реализующая внутризапросный параллелизм, причем доступные в данный момент времени ресурсы ВС используются максимально, но с учетом того, что превышение некоторого предела загрузки ВС приведет к релейному падению производительности системы в целом.

Актуальность темы данной работы обусловлена тем, что, несмотря на наличие большого количества работ в области параллельных баз данных, они, в основном, ориентированы на использование специализированных ВС (Соколинский Л.Б., DeWitt D., Graefe G.). Отличием данной работы является то, что объектом исследования является объектная система управления базами данных (ОСУБД), предназначенная для работы на ВС с общими ресурсами с архитектурой x86, для которой обеспечивается внутризапросный параллелизм в условиях не монопольного использования ресурсов ВС.

Цели и основные задачи работы. Целью работы является разработка методов и моделей проектирования параллельных СУБД и их приложений с задан-

ными характеристиками, а также оценка предельных характеристик СУБД и приложений. Для достижения поставленной цели решаются следующие задачи:

1. Анализ параллельных архитектур СУБД и методов повышения эффективности использования системных ресурсов. Выбор критериев производительности СУБД.
2. Разработка алгебраической модели параллельной ОСУБД. Метод оценки характеристик производительности систем на основе СУБД.
3. Моделирование и исследование свойств модели.
4. Разработка параллелизатора запросов для асинхронного выполнения микроопераций в СУБД.
5. Разработка метода измерения производительности систем на основе СУБД.
6. Создание инженерных методик реорганизации СУБД в соответствии с заданными характеристиками с использованием предложенных моделей. Исследование характеристик производительности систем на основе СУБД с использованием этих методик. Реализация методик для улучшения характеристик сервера ОСУБД.

Методы исследований. Примененные в работе модели используют классический теоретико-множественный аппарат, теорию вероятностей, СМО, методы математической статистики и стохастические алгебры процессов. Вычислительный эксперимент проводился при помощи модифицированного автором средства решения моделей “PEPA” (англ.) – PEPAWorkbench, а также с использованием системы математического моделирования MATLAB.

Эксперименты проводились на реальных базах документов, полученных в результате эксплуатации информационно-поисковой системы (ИПС), в частности, накоплена база новостных сообщений в ИПС «Обзор СМИ», эксплуатируемой в Управлении информационного и документального обеспечения Аппарата Совета Федерации Федерального Собрания Российской Федерации.

Научная новизна. В работе получены следующие новые результаты:

На основе стохастической алгебры процессов “PEPA” (англ.) разработан метод, позволяющий проводить расширенный анализ сложной информационной системы на основе алгебраических моделей, включая анализ по каждому параллельному процессу в отдельности, а также весовое расширение “PEPA”, позволяющее анализировать модель с позиции потребления различных ресурсов системы, описываемых как многомерный вектор.

Разработана комплексная модель сервера объектной системы управления базами данных ODB-Jupiter, а также модели отдельных подсистем с использованием алгебраического подхода.

Разработан метод параллельного выполнения внутренних операций сервера БД с учетом доступных ресурсов ВС на момент начала обработки. Реализован программный модуль унифицированного параллельного выполнения операций.

Практическая ценность и реализация результатов

Разработанный метод моделирования информационных систем ориентирован на моделирование процессов с внутренним параллельным расщеплением, однако может быть использован в широком классе других систем. Разработанный параллелизатор обеспечивает асинхронное выполнение заявок, их диспетчеризацию и унифицированную обработку очередей заявок. Его программная реализация обеспечивает простоту реализации произвольных обработчиков этих заявок и их контроль допустимого использования ресурсов ВС, предотвращающий состоя-

ние релейной деградации производительности ВС в целом. Модуль параллелизатора встроен в сервер ИПС «Обзор СМИ», применяемой в Совете Федерации РФ в качестве центрального модуля распределения рабочей загрузки сервера СУБД. Также, параллелизатор встроен в систему резервного копирования и восстановления информации, используемой в ФГУП «Концерн Системпром». Разработана обобщенная методика реорганизации СУБД с последовательной обработкой в СУБД с параллельной не конвейерной обработкой, что позволяет существенно улучшить эксплуатационные характеристики унаследованных СУБД.

Апробация работы

Содержание отдельных разделов и диссертации в целом было доложено:

- На заседаниях аттестационной комиссии при ежегодной аттестации аспирантов кафедры «Компьютерные системы и сети» МГТУ им. Баумана.
- На семинарах и заседаниях кафедры «Компьютерные системы и сети» МГТУ им. Баумана.
- На межвузовской юбилейной научно-технической конференции аспирантов и студентов «Современные информационные технологии» 15 ноября 2000 г.
- На конференции «Корпоративные базы данных 2003» 17 апреля 2003 г.
- На семинаре московской секции ACM SIGMOD 25 января 2007 г.

Публикации

Основные результаты работы опубликованы в 9 печатных работах.

Личный вклад автора

Все основные научные результаты, методика оценки характеристик производительности ИС на основе СУБД, алгебраические модели СУБД и отдельных подсистем, метод параллельного выполнения операций, разработанные на их основе алгоритмы и программные средства, экспериментальные исследования, приведенные в диссертации, получены автором лично.

Содержание работы

Диссертационная работа состоит из введения, пяти глав, заключения, списка литературы. Общий объем диссертации 253 страницы, включая 53 рисунка, список литературы и приложения. Библиография включает 140 наименование, из них 71 из иностранных источников.

В разделе «Введение» обосновывается актуальность темы диссертации, рассматривается место задач моделирования и проектирования СУБД и их приложений в современных информационных технологиях, дается краткая характеристика таких задач. Формулируется цель работы и её связь с другими аспектами информационных систем. Кроме того, обосновывается необходимость использования схемы выполнения запросов СУБД с внутренним параллельным расщеплением процесса обработки.

В настоящее время посредством математического моделирования СУБД принято решать следующие задачи:

1. Подбор методов индексации данных по используемым типам данных, понимая под типами в том числе и сложные типы пользователей, а также оптимизация схемы данных для ускорения работы приложений СУБД.

2. По известному набору операций и их прогнозируемой плотности в ИС, определение загрузки функциональных модулей СУБД, реализуемость выполнения в конкретных условиях на конкретной СУБД, а также характеристики производительности ИС в целом.
3. По известной архитектуре ВС и структуре СУБД подбор необходимого оборудования для выполнения требований конкретной ИС.

Как показало исследование, в общем случае, задача подбора оборудования является сложно реализуемой по причине отсутствия единообразного подхода для оценки характеристик оборудования различных производителей. Задача подбора методов индексации актуальна в СУБД, предоставляющих достаточные средства для их реализации. Таким образом, наиболее актуальной задачей является вторая задача.

В главе «Анализ существующих методов построения параллельных СУБД и ИС на их основе, анализ методов моделирования» рассматриваются общие аспекты построения СУБД, необходимые для решения вопроса оценки производительности, методы формализации данных и операций в СУБД.

Создание СУБД предполагает не только разработку функциональных модулей, как таковых, но и формальное математическое описание модели хранения данных и манипулирования с ними. В работе отмечены реляционная модель данных Кода, её развитие К. Дейтом и объектная модель. Близко связанными с моделями хранения являются формальные методы манипулирования данными и языки запросов, построенные на основе этих методов. Использование формальных методов описания операций позволяет на их основе строить математические модели баз данных (БД). Наиболее широко применяются алгебраические методы описания операций. Среди них можно выделить работы Е.Ф. Кодда по реляционной алгебре, её современную формализацию Е.М. Бениаминовым в виде многосортной алгебры, К. Бири по формальному описанию БД объектно-ориентированных СУБД средствами многосортной алгебры, работы с использованием других алгебраических методов, такие как HERM-алгебра, основанная на машине абстрактных состояний Ю. Гуревича в работах Б. Талхайма (Bernhard Thalheim). При некоторой модификации также применимы методы императивной спецификации динамических систем А.В. Замулина. Использование алгебраических подходов позволяет провести формальное описание логического плана выполнения запроса (уровень языка запроса), его трансляцию в физический план выполнения (уровень внутренней обработки запросов) и, как следствие, иметь возможность проведения анализа его оптимизации и выполнения по микрооперациям физического плана.

В работе рассматривается классификация М. Стоунбрейкера для многопроцессорных вычислительных комплексов (МВК) и расширение Л.Б. Соколинского.

В СУБД различаются методы хранения данных на внешних накопителях. Так, выделяют метод прямого хранения данных (и модификации), часто применяемый в объектных СУБД, метод нормализованного хранения данных (и модификации), характерный для реляционных СУБД.

Существенными моментами функционирования СУБД являются методы работы с транзакциями и методы выполнения блокировок данных. В настоящее время производители СУБД используют различные методы, в зависимости от целевой области применения СУБД. Рассматриваются модель транзакций в соответствии с требованиями ODMG 3.0 и расширенный вариант на примере ОСУБД Versant, реализующую модель длинных транзакций, позволяющих выполнять распределенную изолированную обработку данных.

Для эффективного использования ресурсов МВК необходимо, чтобы СУБД реализовывала алгоритмы параллельной обработки в соответствие с особенностями их архитектуры. Рассматривается классификация методов параллельного выполнения запросов различного уровня обработки.

Существенным аспектом при проектировании СУБД является то, каким образом происходит взаимодействие клиентов и серверов СУБД. Рассматривается базовая классификация клиент-серверного взаимодействия, а также расширения этой классификации применительно к объектным СУБД.

Рассматриваются основные математические методы моделирования, а также некоторые методы, применяемые для моделирования конкретных элементов информационных систем. Проведен анализ существующих стохастических методов моделирования, таких как марковские цепи, сети Петри, их стохастические расширения, сети рандеву (SRN), сети деятельностей (SAN), а также различные модификации алгебр процессов.

Среди методов моделирования производительности ИС в целом и серверов СУБД в частности, можно выделить:

1. Модели, основанные на коммуникации клиентов и серверов данных. Чаще всего строятся и анализируются средствами СМО. Данный класс моделей позволяет оценить производительность ИС в предположении, что поток запросов клиентов однороден, кроме того возможна оценка времени их выполнения в среднем. Модели позволяют оценить использование различных каналов связи, методов кэширования данных, однако столь высокий уровень абстракции не позволяет получить оценку влияния внутренних особенностей обработки запросов сервером СУБД.
2. Модели хранения данных на ВЗУ, что позволяет оценить минимальное время доступа к данным и производительность ИС в целом. В зависимости от назначения модели, могут учитываться как физические характеристики накопителей данных, такие как время позиционирования головок НЖМД, скорость считывания и схема разметки дисков или только методы размещения данных на них. В настоящее время, модели с использованием внутренних характеристик НЖМД мало применимы, поскольку большинство таких характеристик накопителей от разных производителей не является адекватно сравнимыми.
3. Модели имитационного представления внутренней обработки данных описывают сервер СУБД с позиции последовательности обработки запросов. Могут учитывать транзакционную обработку, разнородность запросов и, как следствие, различные методы их выполнения. Данный класс моделей может быть реализован любыми математическими средствами.
4. Методы, на основе моделей запросов к СУБД применимы в том случае, если проводится анализ характеристик ИС в целом, а не отдельного сервера БД или тракта клиент-сервер. Выделяются методы, основанные на аналитическом описании процесса обработки запросов эмпирическими формулами, исходя из априорного предположения об однотипности функционирования коммерческих СУБД и, как следствие, на однотипности выполняемого набора операций проекции, слияния и выборки данных. К этому же классу методов относятся работы, в где проводится анализ исходных запросов в алгебраической форме, анализ способов их преобразования в физический план конкретной СУБД и выполнение, при помощи модели, имитирующей внутреннюю обработку запросов конкретной СУБД. Эти модели могут быть реализованы как традиционными средствами СМО, так и в форме алгебр процессов.

5. Условно в отдельную группу можно выделить модели, в которых проводится анализ производительности сервера СУБД на основе имеющихся сведений о порядке обработки данных в транзакции применительно к конкретной ИС. В печатных источниках выявлены модели, в которых анализ производительности проводится в зависимости от количества элементов данных внутри транзакции, так и модели блокировки данных применительно к транзакциям OLTP систем.

Приводится краткий обзор некоторых инструментальных средств моделирования, реализующих перечисленные модели.

В результате проведенных исследований можно заключить, что методы, ориентированные на моделирование характеристик отдельных подсистем или сервера СУБД в целом, не позволяют достоверно оценивать поведение ИС на основе СУБД, поскольку не учитывают взаимного влияния выполняемых запросов (в интерпретации не элементарных операций, а последовательности некоторых действий) различных пользователей. Для получения достоверных характеристик ИС в целом необходимо проводить анализ логических и физических планов выполнения запросов, а также учитывать особенности выполнения микроопераций физического плана сервером СУБД.

Полная модель ИС на основе СУБД, таким образом, включает в себя следующие частные модели:

- Модель взаимодействия клиента и сервера СУБД, включая многозвенные клиент-серверные архитектуры;
- Модель преобразователя исходных запросов в логический план выполнения по схеме данных;
- Модель оптимизатора логического плана;
- Модель планировщика физического плана выполнения;
- Модели функциональных обработчиков запросов.

В качестве математического метода моделирования целесообразно использовать алгебраические методы. В данной работе основным методом выбрана алгебра процессов “PEPA” (Performance Evaluation Process Algebra), позволяющая выполнять моделирование параллельных систем, имеющих определенные точки синхронизации, использующая формальный метод описания модели.

Алгебра процессов “PEPA” является удобным средством моделирования взаимодействующих процессов. Тем не менее, она имеет ряд ограничений, затрудняющих ее использование. В числе таких ограничений можно назвать отсутствие возможности анализа каждого процесса алгебры в отдельности, что не позволяет строго определить время выполнения того или иного процесса, производительность того или иного процесса в случае, если несколько разных процессов содержат в себе одинаковые типы действий. Следует также отметить то, что, предоставляя структурное описание процессов, алгебра “PEPA” потенциально может быть использована для введения в модель дополнительных данных.

В главе рассмотрен объект исследования работы – объектная СУБД ODB-Jupiter и информационно-поисковая система (ИПС) «Обзор СМИ», применительно к которой рассматриваются схема данных и схема обработки запросов, набор выполняемых операций, метод формирования индексных данных. ОСУБД ODB-Jupiter является специализированной документально-ориентированной объектной СУБД для работы в составе ИПС. В дальнейшем, с использованием ИПС, на основе данной СУБД, проводится анализ и моделирование характеристик ИПС и СУБД, а также реорганизация СУБД для улучшения характеристик производительности.

В главе «Разработка метода моделирования производительности ИС, использующей СУБД» на основе ранее выбранной алгебры процессов “PEPA” реализован метод моделирования производительности ИС.

Формальный синтаксис алгебры “PEPA” с квазиоперацией *VM()* и расширенным комбинатором кооперации может быть представлен в форме Бэкуса-Наура следующим образом:

$$P_{dynamic} ::= (\alpha, r).P \mid P + Q; \qquad C_{static} ::= VM_{k1}(P) \underset{L}{\triangleright\!\triangleleft} VM_{k2}(Q) \mid P / L$$

Основные языковые конструкции:

Префикс: компонент ***(α,r).P*** выполняет действие с типом α и интенсивностью *r*. Компонент *(α,r).P* впоследствии становится компонентом *P*. Действие определяется как $a = (\alpha, r)$.

Выбор: компонент ***P+Q*** представляет систему, которая может вести себя как компонент *P* или компонент *Q*. Компонент *P+Q* разрешает все текущие действия *P* и *Q*, т.е. $Act(P + Q) = Act(P) \bigcup Act(Q)$.

Кооперация: $VM_{k1}(P) \underset{L}{\triangleright\!\triangleleft} VM_{k2}(Q)$ называется совместное действие для компонентов *P* и *Q* на множестве кооперации $L \subseteq A$. Таким образом, возможна блокировка, при которой один компонент будет ожидать другой. Если $L=\varnothing$, то компоненты *P* и *Q* выполняются параллельно. VM_{k1} и VM_{k2} есть квазиоперации, не вводимые в аксиоматику алгебры, ассоциирующие виртуальные ВС, на которых выполняются процессы с их статически комбинируемыми описаниями *P* и *Q*.

Скрытие: компонент ***P/L*** ведет себя как компонент *P,* за исключением всех действий, имеющих тип, входящих во множество *L.* Действия, типы которых попадают во множество *L*, объединяются в единое действие с неизвестным типом τ, и их длительности объединяются. Кроме того, эти действия не могут объединяться с другим компонентом операцией кооперации.

В отличие от базовой интерпретации уравнения системы, в которой процессы неразличимы, в данной работе уравнение системы *Sys* есть функция кооперации по множеству динамических компонентов *P*, каждый из которых представляет собой процесс: $Sys ::= f_{\triangleright\!\triangleleft}(\{P_1, P_2 \ldots P_n\})$.

Процесс есть последовательность компонентов $P_k = \{C_1^k \ldots C_i^k\}$, где C_i^k - элементарный компонент процесса i, определяющий действия для перехода к следующему компоненту, причем $C_i^k \in ds(P_k)$, где *ds(C)* есть порождающее множество компонента C.

Алгебра процессов “PEPA” не имеет непосредственного метода решения. Процесс решения разбивается на два этапа – генерация замкнутой марковской цепи с постоянными параметрами и её решение традиционными для марковских цепей методами. Главным определяемым параметром “PEPA” является вероятность состояний марковской цепи (эквивалентных компонентам “PEPA”) в установившемся режиме. Остальные параметры модели определяются с помощью полученной марковской цепи с использованием т.н. метода стоимостных структур и соответствующей интерпретацией полученных значений.

В работе предлагается сохранять на момент генерации замкнутой марковской цепи информацию о позиции процесса в уравнении системы, в рамках которого выполняется действие, вызывающее переход к другому состоянию, что не делается в базовом варианте “PEPA”. Таким образом, интенсивность перехода $q(C_i, C_j)$ между компонентами алгебраической модели C_i и C_j (эквивалентными со-

стояниями марковской цепи) определяется как $q(C_i, C_j) = \sum_k \sum_{a \in Act(C_i^k | C_j^k)} r_\alpha^k$, где C_i^k и C_j^k есть компоненты, соответствующие *k*-му процессу. Форма $q(C_i, C_j)$ соответствует принятой в “PEPA” форме, в то время как $q(C_i^k, C_j^k)$ является её расширением.

Элементы $q(C_i, C_j)$ представляют собой недиагональные элементы инфинитезимальной образующей матрицы Q марковского процесса. Диагональные элементы вычисляются как сумма недиагональных элементов каждой строки, взятая с обратным знаком. Решением является решение матричного уравнения $\Pi \cdot Q=0$, где П – матрица стационарных вероятностей состояний. Дополнительным нормирующим условием является $\Sigma\Pi(C_i)=1$.

Базовый метод анализа производительности в алгебре процессов “PEPA” основан на методе стоимостных структур (reward structures). Однако данный метод не обладает достаточной семантикой, поскольку стоимость назначается для определенных состояний, определяемыми неформализованными критериями. В методе стоимостных структур каждому переходу между состояниями назначают определенный вес. Каждый переход ассоциирован с конкретным действием из исходной модели. Суммарная стоимость $R_i = \sum_i \rho_i \Pi(C_i)$, где ρ_i есть стоимость, соответствующая компоненту C_i , а $\Pi(C_i)$ есть стационарная вероятность нахождения системы в состоянии, соответствующего компоненту C_i . Следует заметить, что вектор $\vec{\rho}$ в общем случае может содержать произвольные числа, в соответствие с семантикой оцениваемой марковской цепи. Расширение информации исходной модели в промежуточной марковской цепи позволило определить следующие характеристики производительности, использование которых в “PEPA” затруднено из-за невозможности разделения характеристик различных процессов:

Производительность системы по типу действия. Производительность системы по типу действия представляет собой число заявок, обрабатываемых данным типом действия. Производительность процесса, таким образом, определяется как суммарный поток вероятности по некоторому типу действия α. $Thr_\alpha^k = \sum_i p_i \lambda_{ij}^k$, где *k* – индекс процесса. Очевидно, общая производительность (форма “PEPA”) по типу α есть $Thr_\alpha = \sum_k Thr_\alpha^k$.

Время отклика. Актуально для определения в рамках одного процесса уравнения системы. Определяется как $R^k = \frac{1}{Thr_\alpha^k} - \sum_y \frac{1}{r_a(*C_y)}$, где Thr_α^k - производительность по характеризующей деятельности *k*-го процесса, время вне обработки заявки $Z = \sum_y \frac{1}{r_a(*C_y)}, *C_y \in \{*C_{client}\}$, где компонент $*C_y$ обозначает динамический компонент конкретного процесса клиента.

Абсолютная утилизация ресурса. Физический смысл данной величины – процент времени выполнения системой данного множества типов действий. В процессе оптимизации системы позволяет явно обнаруживать узкие места. Допустим, существует множество компонентов $Comp_a$, которые имеют возможные переходы с действиями $a_i \in Act(C_i), C_i \in Comp_a$. Тогда абсолютной утилизацией ресурса, соответствующего типу действия a_i является $U = \sum_i \Pi(C_i), C_i \in Comp_a$.

Относительная утилизация ресурса. Под относительной утилизацией ресурса подразумевается эффективность использования конкретного типа действий в системе со статически комбинируемыми процессами, в частности, параллельной кооперации действий. Т.е. данная величина представляет собой меру эффективности параллельного использования типов действий (дисбаланс параллелизма). Для определения относительной утилизации определим понятие предельной производительности циклического процесса как $Thr^k_{proc} = \frac{1}{\sum \frac{1}{\lambda_i}}$, где λ_i есть интенсивности переходов между состояниями процесса с индексом *k* в уравнении системы при рассмотрении этого процесса как самостоятельную марковскую цепь. При этом пассивные действия игнорируются.

Относительная утилизация по типу действия α определяется как $\widetilde{U}_\alpha = Thr^k_{proc} \frac{N_\alpha}{Thr_\alpha}$, где N_α - число параллельных процессов в уравнении системы, выполняющих данный тип действия, а Thr_α - общая производительность по типу действия α.

Традиционный способ наложения ограничений в моделях “PEPA” заключается во введении в модель дополнительных контролирующих процессов, объединяемых посредством комбинатора кооперации по некоторому общему ограничивающему набору типов действий. При этом уже на этапе генерации марковской цепи отбрасываются состояния, в которые система не может перейти после наложения ограничения. Однако данный метод требует изменение модели каждый раз, когда возникает необходимость изменения наложенных ограничений, что требует выполнения длительной операции генерации новой марковской цепи. Кроме того, введение дополнительных процессов имеет существенный недостаток – усложнение модели, а также возможное увеличение количества состояний в виду возможного введения фиктивных промежуточных действий для кооперации с контролирующим процессом. В ряде источников зафиксировано расширение алгебр процессов посредством добавления в дескриптор действия дополнительных параметров. Однако, такой подход усложняет восприятие модели и управление параметрами. Поэтому, в данной работе расширение производится не для конкретных действий, а для типа действия в целом.

Естественным следствием анализа модели “PEPA” по каждому процессу является ассоциация каждого перехода марковской модели с набором действий, которые становятся активными в этот момент. Поскольку в рамках марковской цепи, генерируемой по модели “PEPA”, один переход между соседними состояниями может быть ассоциирован лишь с одним действием, другие процессы, комбинируемые в данном состоянии кооперацией, также имеют возможные переходы, ассоциированные со своими действиями.

Множество возможных комбинаций активных действий в любом состоянии системы $Sys_{\{C_{i_1 j_1}, C_{i_2 j_2} \ldots C_{ij}\}}$ есть декартово произведение условных множеств активных действий по компонентам всех процессов системы:

$$CAct(\{C_{i_1 j_1}, C_{i_2 j_2} \ldots C_{ij}\}) = Act(C_{i_1 k_1} \mid f_{\triangleright\triangleleft}(\{C_{i_1 j_1}, C_{i_2 j_2} \ldots C_{ij}\}) \times \ldots$$

$$\times Act(C_{i_2 k_2} \mid f_{\triangleright\triangleleft}(\{C_{i_1 j_1}, C_{i_2 j_2} \ldots C_{ij}\}) \times \ldots \times Act(C_{ij} \mid f_{\triangleright\triangleleft}(\{C_{i_1 j_1}, C_{i_2 j_2} \ldots C_{ij}\})$$

В общем случае $CAct(\{C_{i_1 j_1}, C_{i_2 j_2} \dots C_{ij}\}) = \{RAct_1, RAct_2 \dots RAct_n\}$, где $RAct$ есть элементарное множество действий по различным процессам уравнения системы.

Кроме того, система описывается конечной матрицей доступных ресурсов и матрицами реакции на превышение ресурсов, соответственно $SYS_{RS} = \begin{bmatrix} RS_{1,1} & RS_{2,1} & \dots & RS_{m,1} \\ RS_{1,2} & RS_{2,2} & \dots & RS_{m,2} \\ \dots & \dots & \dots & \dots \\ RS_{1,n} & RS_{2,n} & \dots & RS_{m,n} \end{bmatrix}$ и $SYS^i_{ex} = \begin{bmatrix} k^i_{1,1} & k^i_{2,1} & \dots & k^i_{m,1} \\ k^i_{1,2} & k^i_{2,2} & \dots & k^i_{m,2} \\ \dots & \dots & \dots & \dots \\ k^i_{1,n} & k^i_{2,n} & \dots & k^i_{m,n} \end{bmatrix}$, где n – количество типов ресурсов, m – количество виртуальных вычислительных систем, на которых происходит выполнение процессов, $i = \{1,2,3\}$.

Каждый столбец матриц определяет предельный условный объем некоторых ресурсов виртуальной вычислительной системы VM_k, на которой выполняется конечное множество процессов $\{Proc_{k1}, Proc_{k2} \dots Proc_{k1}\}$.

Действие характеризуется определенным типом и является неделимым, переводящим систему в другое состояние. Различают активные и пассивные действия. Каждый тип действия α_i характеризуется параметром вектором потребления ресурсов $act_\alpha = \begin{bmatrix} rs_{i,1}\ t_{i,1} \\ rs_{i,2}\ t_{i,2} \\ \dots\ \dots \\ rs_{i,j}\ t_{i,j} \end{bmatrix}$, где $rs_{i,j}$ есть величина потребления j-го ресурса, $t_{i,j}$ есть среднее время, за которое происходит потребление ресурса j. Нулевое значение $t_{i,j}$ является признаком статического потребления ресурса, независимо от времени, т.е. функция потребления $Use(act^k_\alpha) = \begin{cases} t_{i,j} > 0, \left(rs_{i,j} \cdot \lambda_\alpha \cdot t_{i,j}\right) \\ t_{i,j} = 0, \left(rs_{i,j}\right) \end{cases}$. Активные действия характеризуются коэффициентом активности r_a. Пассивное действие может быть реализовано только в рамках кооперации с компонентом, описывающим активное поведение системы. Пассивное действие не потребляет ресурсы.

Таким образом, для любого перехода, переводящего систему $Sys_{\{C_{i_1 j_1}, C_{i_2 j_2} \dots C_{ij}\}}$ в соседнее состояние через действие *a*, множество потреблений ресурсов определяется как матричная сумма:

$$Rs^m_v = \sum_k \sum_w Use(act^k_w), act^k_w \in RAct, RAct \in CAct(\{C_{i_1 j_1}, C_{i_2 j_2} \dots C_{ij}\}), P_k \in VM_m,$$ где *k*-индекс процесса в уравнении системы, *v* - индекс множества Rs в множестве комбинаций переходов для состояния $Sys_{\{C_{i_1 j_1}, C_{i_2 j_2} \dots C_{ij}\}}$, *w*-индекс действия во множестве активных действий, *m*-индекс виртуальной машины.

Критерий не превышения ресурса *j*: $Rs^m_j \le SYS^m_j \ \ \forall v$.

1) Интенсивность активных действий в состоянии, вызвавшем максимальное превышение ресурса *j* есть функция, $\lambda'(\Delta) = \lambda \cdot f(\Delta)$, где $\Delta = \dfrac{Rs^m_j}{RS_{i,j}}$.

Для функции $f(\Delta)$ вида $f(\Delta)=f_1(\Delta)\cdot f_2(\Delta)$. Приняты $f_2(\Delta)=\left(\frac{1}{\Delta}\right)^{k^3_{i,j}}=\left(\frac{RS_{i,j}}{Rs^m_j}\right)^{k^3_{i,j}}$,

где $k^3_{i,j}=1$ и $f_1(\Delta)=1-F_{Weibull(\alpha,\beta)}(\Delta-1)=\exp\left(-\left(k^2_{i,j}\cdot(\Delta-1)\right)^{k^1_{i,j}}\right)=\exp\left(-\left(k^2_{i,j}\cdot\left(\frac{Rs^m_j}{RS_{i,j}}-1\right)\right)^{k^1_{i,j}}\right)$

Итоговая функция коррекции интенсивности имеет следующий вид:

$$\lambda'=\lambda\cdot\left[\exp\left(-\left(k^2_{i,j}\cdot\left(\frac{Rs^m_j}{RS_{i,j}}-1\right)\right)^{k^1_{i,j}}\right)\cdot\left(\frac{RS_{i,j}}{Rs^m_j}\right)^{k^3_{i,j}}\right]$$

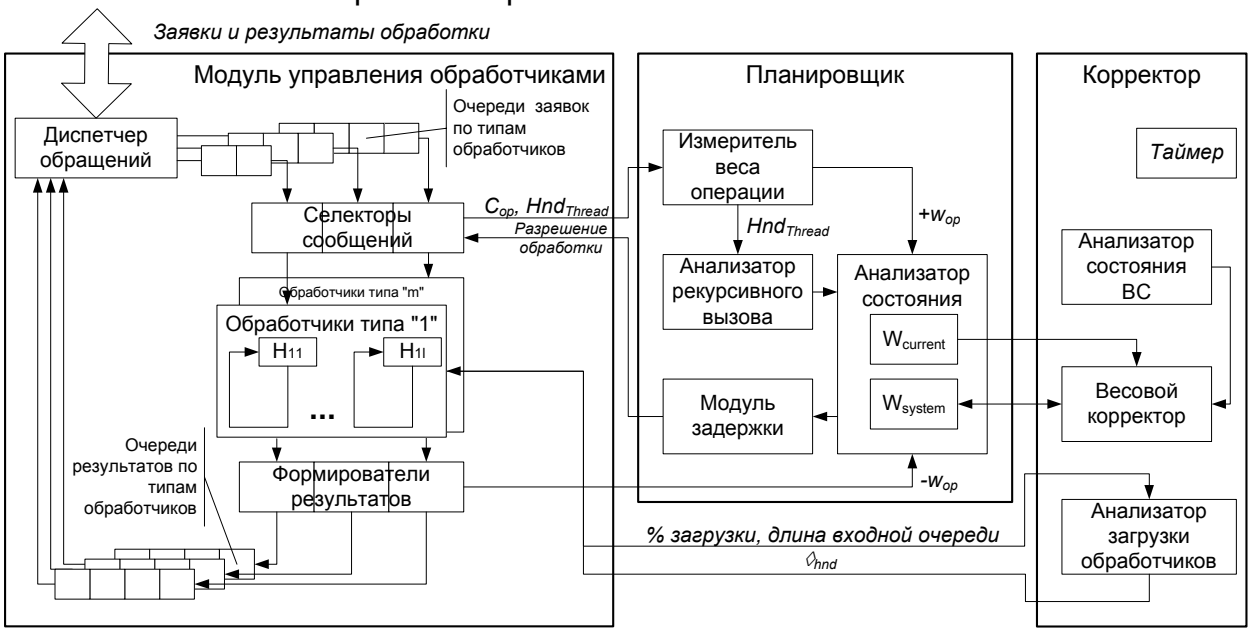
В главе рассмотрено применение конструкций алгебры “PEPA” для моделирования ИС. Рассмотрены особенности применения “PEPA” как для элементарных структур типа очередей обработки, так и более сложные подсистемы, включая иерархический внутренний параллелизм, обеспечиваемый в реальных ИС разработанным в данной работе параллелизатором.

Применительно к объекту исследования ОСУБД и ИПС на её основе, разработаны модели различного уровня детализации и отдельных подсистем.

В главе «Характеристики производительности СУБД с использованием адаптивного планировщика параллельных запросов» проводится разработка метода и его реализация в виде программного модуля, обеспечивающего асинхронное параллельное выполнение операций. В отличие от других работ по организации параллельной обработки данных (Volcano – G. Graefe; DBS3 - Mikal Ziane и др.; Gamma - D. DeWitt; Омега – Соколинский Л.Б., Цимблер М.Л.), разработанный метод позволяет его использовать не только в рамках монопольного использования ресурсов ВС для решения задач СУБД, но и в составе комплекса программ, функционирующего на тех же ВС с учетом текущих ресурсов ВС. Отличием разработанного метода организации параллелизма от традиционных подходов типа MPI, OpenMP, mpC является наличие централизованного управления степенью параллелизма и возможность реакции на динамическое состояние ВС, посредством обратной связи через корректор параллелизатора.

Предлагаемый метод параллельного выполнения запросов предполагает, что в рассматриваемой СУБД, поступающий запрос поступает через диспетчер запросов в функциональный модуль, обеспечивающий корректность выполнения запроса. Микрооперации, из которых состоит физический план выполнения запроса, рассматриваются как самостоятельные операции, процессом выполнения которых необходимо управлять для достижения равномерной загрузки ВС. Для организации параллельного выполнения запроса строится параллельный план с максимально возможной степенью параллелизма. В данном методе предполагается соответствие типа выполняемой микрооперации группе потоков обработки, однако их активное количество может меняться в зависимости от текущего состояния ВС. Кроме того, число входных и выходных параметров обработчиков зависит от типа операторов. Используются операции постановки заявки на выполнение и ожидания результата выполнения конкретной микрооперации. Это позволяет, имея линейный физический план выполнения запроса, построить параллельный план выполнения, выделив точки постановки на обработку и ожидания обработанных данных, который будет выполнен максимально параллельно с учетом текущей загрузки ВС и СУБД. Отказ от конвейерной схемы обработки данных

обусловлен тем, что разработанный метод параллельного выполнения, прежде всего, предназначен для внедрения в существующих последовательных системах обработки данных, поскольку переход на конвейерную схему обработки требует полное изменение алгоритмов обработки.



Функциональная схема асинхронного адаптивного параллелизатора

Программный модуль «параллелизатор» обеспечивает максимально возможное параллельное выполнение всех поступивших операций не конвейерного типа. Решение о постановке операции (соответствующей ей заявки) принимается планировщиком. Корректор обеспечивает контроль состояния ВС и внесения поправок в планировщик. Выполнение обеспечивается обработчиками.

Формально определим параллелизатор.

Параллелизатор: $Par = <Hnd, W_{SYS}, W_{cur}>$, где W_{SYS}, W_{cur} – векторы доступных ресурсов и текущего состояния системы.

Множество обработчиков: $Hnd=\{hnd\}$, где $hnd_j=<\{op\}, st_j>$, где $op \in A$, A – множество типов операций (число каналов обработки для различных операций различно).

Текущее состояние j-го обработчика $st_i = < nc, Nc, l_{in}, l_{out}, L_{in}, L_{out} >$, где nc, Nc- число активных и доступных каналов обработки, l_{in}, L_{in} - текущая и допустимая длины входной очереди, l_{out}, L_{out} текущая и допустимая длины выходной очереди.

Операция $op_i=< \{t_{in}\}, \{ t_{out}\}, W_{op,i} >$, где t_{in}, $t_{out} \in T$ - типы входных и выходных параметров, T – множество типов данных.

Заявка $q=<№, pr , params:op_i, W_q>$, где № - последовательный номер операции в системе, pr – приоритет заявки, $params:op_i =< \{paramin:t_{in}\}, \{ paramout:t_{out}\} >$ – множества значений входных и выходных параметров соответствующих типов.

$$W_{cur} = \begin{bmatrix} rs_1 \\ rs_2 \\ \dots \\ rs_N \end{bmatrix}, \quad W_{op,i} = \begin{bmatrix} w_1 \\ w_2 \\ \dots \\ w_N \end{bmatrix}, \quad W_{SYS} = \begin{bmatrix} RS_1 \\ RS_2 \\ \dots \\ RS_N \end{bmatrix},$$ где W_{cur} - вектор текущего состояния, $W_{op,i}$ - вес операции, W_{SYS} - вектор доступных ресурсов, rs_j - величина потребления j-го ресурса, N – число типов ресурсов, w_j – величина потребления j-го ресурса, по-

лученная на основе априорных данных или в результате накопленных статистических данных во время работы системы, *i* – тип операции, RS_j – Допустимая величина потребления j-го ресурса, полученная на основе априорных данных или выполненного тестирования. Частным случаем условия постановки заявки операции на выполнение является неравенство $W_{SYS}^{j} \geq W_{cur}^{j} + W_{op,i}^{j}$ для всех ресурсов. Рассмотрены и другие варианты планирования, однако до стадии реализации доведена именно эта схема.

В работе проводится анализ доступных для измерения ресурсов ОС MS Windows счетчиков производительности, необходимых для работы корректора.

В главе «Методики проектирования параллельных СУБД» приводятся разработанные методики проектирования параллельных СУБД в контексте реорганизации существующей последовательной СУБД.

Разработана обобщенная методика реорганизации СУБД, включающая методику оценки характеристик производительности ИС на основе СУБД, в рамках которой проводится создание и параметризация моделей ИС и сервера СУБД, методику нагрузочного тестирования сервера ОСУБД в составе ИС, необходимую для получения параметров моделей и оценки характеристик производительности реальной ИС. Также, методика учитывает создание и модификацию алгоритмов сервера БД для обеспечения параллельного обслуживания запросов.

Методика нагрузочного тестирования сервера ОСУБД в составе ИС предполагает выбор метода измерения интервалов времени выполнения внутренних функций программы, создание замкнутого контура тестирования программа-клиент ИС – сервер СУБД – модуль нагрузочного тестирования, обеспечивающего нагрузку сервера СУБД в соответствии с реальными запросами клиентов ИС, выбор способа и измерение характеристик сервера СУБД, а также обработку полученных данных.

Для решения задачи анализа производительности реальной системы и определения априорных данных для выполнения математического моделирования разработан соответствующий методика нагрузочного тестирования. В рамках данной методики выделяется схема определения малых величин времени выполнения, т.е. величин порядка 10 мкс на многопроцессорных системах, основанная на особенностях современных процессоров, заключающихся в возможности получения значения счетчика тактов для каждого процессора и методов построения операционных систем, позволяющих управлять режимами работы процессоров и построить процесс измерения таким образом, чтобы получить достоверное время выполнения заданного фрагмента кода программы. Интегрированным в сервер СУБД модулем измерения, реализующего данный алгоритм, были измерены времена выполнения операций исходной СУБД и проверены результаты выполнения модифицированной СУБД.

Для решения задачи измерения производительности ИС в целом, разработан комплекс программ, позволяющих сформировать контур выполнения клиент ИС – сервер БД – контрольно управляющий модуль. Особенностью данного комплекса тестирования является то, что на сервер БД поступает полноценный запрос, сформированный реальным приложением ИС. Контрольно-управляющий модуль обеспечивает съем в реальном времени внутренних характеристик выполнения запросов с сервера БД и обеспечивает выдачу команд и анализ результатов полученных приложением ИС.

Методика оценки характеристик производительности ИС на основе СУБД предполагает выявление требований к производительности ИС, выявление струк-

туры СУБД, модели данных, схемы данных, множеств последовательностей выполняемых операций. В рамках данной методики производится создание необходимых моделей, их параметризация с использованием значений, полученных при применении предыдущей методики на реальной ИС, проводится модификация моделей в соответствие с будущими алгоритмами работы, проводится вычислительный эксперимент и оценка полученных значений.

Для обеспечения трансляции моделей на различные архитектуры ВС при решении задачи предсказания характеристик ИС, разработана методика подбора коэффициентов функции коррекции интенсивностей марковской цепи, применительно к задаче моделирования систем с общей памятью. Выполнено моделирование деградации пропускной способности подсистемы ОЗУ в зависимости от числа одновременно работающих процессоров и получены нормированные коэффициенты деградации производительности ВС при совместном доступе.

Для учета взаимного влияния процессоров через подсистему памяти, на модель подсистемы полнотекстового поиска наложено параметрическое ограничение. Характер взаимного влияния общего ресурса в системе известен и может быть аппроксимирован обратной степенной функцией, входящей в состав ранее указанной формулой коррекции интенсивностей марковской цепи. Коэффициент вычислен как $k^3_i = \frac{\log\left[1 + \%_{MEM}\left(1/k_{\text{ОЗУ}}(n) - 1\right)\right]}{\log(\text{n})}$, где время выполнения кода программы распределяется между временем обмена с ОЗУ и временем работы ЦП. Причем, доли выполнения связаны соотношением $\%_{CPU} = 1 - \%_{MEM}$, при условии пренебрежения потерей времени на остальных аппаратных подсистемах. Учитывая, что коэффициент k^3_i в начальный момент может быть подобран эмпирически, это позволяет оценить распределение долей времени работы ЦП и обмены ОЗУ на конкретной ВС. Из приведенного выражения можно определить, что $\%_{MEM} = \frac{n^{k^3_i} - 1}{1/k_{\text{ОЗУ}}(n) - 1}$. Пересчет долей времени выполнения выполняем по формуле

$$\%_{MEM\_новая} = \frac{\%_{MEM} \cdot \frac{Thr_{MEM}}{Thr_{MEM\_новая}}}{\%_{MEM} \cdot \frac{Thr_{MEM}}{Thr_{MEM\_новая}} + \%_{CPU} \cdot \frac{Thr_{CPU}}{Thr_{CPU\_новая}}},$$ где Thr_{MEM} и $Thr_{MEM\_новая}$ пропускные способности подсистем памяти существующей и новой системы, а Thr_{CPU} и $Thr_{CPU\_новая}$ индексы производительности процессоров в соизмеримых величинах существующей и новой системы (Для процессоров одной серии для пересчета производительности применимы значения тактовых частот).

В работе приводится метод использования разработанного параллелизатора в тексте программ, а также пример его применения. Модуль параллелизатора предназначен для использования в программах на C++ с использованием специальных классов-переходников, что позволяет использовать большинство компиляторов C++, в то время как использование специальных директив в методах типа OpenMP требует специальные компиляторы. Метод предполагает, что программист создает класс-потомок унифицированного класса сообщения, где добавляет необходимые обработчику входные и выходные параметры. Классы обработчики реализуются на основе унифицированного класса посредством перекрытия метода выполнения. Предусмотрена возможность присоединение модуля асинхронно-

го корректора, универсальная реализация которого нецелесообразна, т.к. размерность весового вектора и алгоритм коррекции зависит от требований программы.

Рассматривается применение методики нагрузочного тестирования сервера ОСУБД в составе ИС применительно к ИПС «Обзор СМИ».

Разработанные методики обеспечивают выполнение задачи реорганизации СУБД с последовательной схемой обработки запросов в СУБД с параллельной схемой обработки запросов, обеспечивают оценку характеристик производительности ИС в целом и отдельных подсистем сервера СУБД, а также моделирование, анализ и предсказание характеристик ИС на ВС, для которых не проводился эксперимент на ИС.

В главе «Выполнение модельных и натурных экспериментов, анализ результатов» представлены результаты моделирования и результаты экспериментальных данных, полученных в процессе выполнения тестовых последовательностей в ИПС «Обзор СМИ».

В работе проведено моделирование ИС по экспериментальным данным, полученным на ВС на основе ЦП AMD Athlon 64 x2 4400+, AMD Opteron 880 (4 шт.), Intel Xeon 2.6 (4 шт.). Анализ адекватности моделей проверен на подсистеме полнотекстового поиска. Эксперимент проведен с целью оценить возможности внутренней параллельной обработки, исключив влияние медленных подсистем ВС, типа внешних накопителей данных. Для этого, тестовые запросы были подобраны таким образом, чтобы все промежуточные данные, необходимые для их обработки, были помещены в ОЗУ. Следует заметить, что такой режим в значительной степени соответствует установившемуся режиму работы ИПС «Обзор СМИ», поскольку при имеющемся наборе данных, наиболее вероятна работа пользователей системы с общим ограниченным массивом данных.

Результаты экспериментальных данных показывают снижение времени отклика в ИС с параллельным обслуживанием запросов в режимах одиночной загрузки серверов от 37 для 2-процессорной ВС до 65 процентов для 8-процессорной ВС. Кроме того, улучшено использование ресурсов процессоров. Таким образом, можно заключить об эффективной работе параллелизатора, обеспечивающего диспетчеризацию заявок на выполнение и его минимальном влиянии на ИС в целом.

В разделе «Общие выводы» сформулированы основные выводы и результаты данной работы, а также намечены дальнейшие пути развития метода моделирования ИПС и улучшения характеристик параллелизатора.

Основные результаты работы

1. Разработан алгебраический метод моделирования производительности, позволяющий оценивать параллельные СУБД с внутренним расщеплением параллельных процессов, обладающий свойством композиционности и позволяющий получать численные значения характеристик производительности, времени отклика, процент использования ресурсов по типу действия.
2. Предложен и реализован метод параллельного выполнения операций, который предполагает контроль допустимого уровня параллелизма в реальном времени и позволяет предотвратить состояние деградации производительности ВС в целом. Использование данного метода позволило увеличить производительность и уменьшить время отклика СУБД при работе на малых плотностях запросов в зависимости от архитектуры ВС от 37 до 65 процентов и не ухудшить их на режимах полной нагрузки.

3. Разработан способ измерения временных характеристик подсистем ОСУБД и ОСУБД в целом, позволяющий измерять малые величины времени и времена взаимодействия параллельных процессов, существенно повышающий точность моделирования и проектирования.
4. Разработаны инженерные методики применения предложенных моделей, позволяющие повысить производительность проектируемой ИС на базе ОСУБД.
5. Создана модель ОСУБД ODB-Jupiter, позволившая определить возможность функционирования ИС и выполнить ее преобразование в систему с внутренним параллельным выполнением запросов.
6. Предложенные в данной работе методы, модели и алгоритмы реализованы в программных комплексах ИПС «Обзор СМИ», ИПС «Архив председателя Совета Федерации Федерального Собрания РФ» в Аппарате Совета Федерации Федерального Собрания Российской Федерации, комплекс резервного копирования и восстановления информации в ЗАО «Всесоюзный институт волоконно-оптических систем связи и обработки информации», доведены до состояния промышленного программного продукта, внедрены на нескольких предприятиях.

Работы по теме диссертации

1. Разработка многопоточного сервера СУБД для Windows NT / А.М. Андреев, Д.В. Березкин, Ю.А. Кантонистов, Р.С. Самарев //Компьютерная хроника. – 1999. – N4. – С. 73-84.
2. Андреев А.М., Березкин Д.В., Самарев Р.С. Внутренний мир объектно-ориентированных СУБД // Открытые системы. – 2001. – № 3. – С. 47-57.
3. Настраиваемый интерфейс WEB-сервера ИПС, удаленное администрирование сервера ИПС / А.М. Андреев, Д.В. Березкин, Р.С. Самарев, А.В. Челмодеев //Современные информационные технологии: сборник докладов и сообщений. – М.:МГТУ, 2001, – С. 110-117.
4. Объектная распределенная ОСУБД / А.М. Андреев, Д.В. Березкин, Р.С. Самарев, А.В. Челмодеев // Современные информационные технологии в управлении и образовании – новые возможности и перспективы использования: Сборник научных трудов. – М.: ФГУП НИИ «Восход», МИРЭА., 2001. – С. 65-68.
5. Березкин Д.В., Самарев Р.С. Система построения архивов электронных документов // Информатика и системы управления в XXI веке: сборник трудов – М.: МГТУ, 2003. – №1. – С. 376-379.
6. Березкин Д.В., Морозов В.В. Самарев Р.С. Подсистема сбора сообщений с сайтов новостной сети Интернет // Информатика и системы управления в XXI веке: сборник трудов – М.: МГТУ, 2003. – №1. – С. 409-410.
7. Андреев А.М., Березкин Д.В., Самарев Р.С. Моделирование информационных систем. // Информатика и системы управления в XXI веке: Сборник трудов – М.: МГТУ, 2003. – №1. – С. 385-400.
8. Березкин Д.В., Каплин К.В., Самарев Р.С. Компилятор OQL для объектной СУБД //Информатика и системы управления в XXI веке: сборник трудов – М.: МГТУ, 2003. – №1. – С. 380-384.
9. Анализ производительности разрабатываемых СУБД и ИС на их основе с использованием алгебраических моделей / А.М. Андреев, Д.В. Березкин, Р.С. Самарев, В.В.Сюзев //Вестник МГТУ. Приборостроение.–2007. – №3.– С.94-115