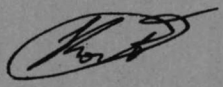
На правах рукописи



КОСТЕНЕЦКИЙ Павел Сергеевич

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИЕРАРХИЧЕСКИХ МНОГОПРОЦЕССОРНЫХ СИСТЕМ БАЗ ДАННЫХ

05.13.18 – математическое моделирование,
численные методы и комплексы программ

Автореферат
диссертации на соискание ученой степени
кандидата физико-математических наук

Челябинск – 2010

Работа выполнена на кафедре системного программирования Южно-Уральского государственного университета.

Научный руководитель: доктор физ.-мат. наук, профессор
СОКОЛИНСКИЙ Леонид Борисович.

Официальные оппоненты: член-корр. РАН, доктор физ.-мат. наук, профессор
ВОЕВОДИН Владимир Валентинович;

кандидат физ.-мат. наук, доцент
ДЕМЕНЕВ Алексей Геннадьевич.

Ведущая организация: Институт программных систем имени А.К. Айламазяна РАН

Защита состоится 3 марта 2010 г. в 12 часов на заседании диссертационного совета Д 212.298.14 при Южно-Уральском государственном университете по адресу: 454080, г. Челябинск, пр. Ленина, 76, ауд. 1001.

С диссертацией можно ознакомиться в библиотеке Южно-Уральского государственного университета.

Автореферат разослан 2 февраля 2

Ученый секретарь
диссертационного совета Л.Б. Соколинский

Общая характеристика работы

Актуальность темы. Современные многопроцессорные системы в большинстве случаев организуются по иерархическому принципу. Например, большая часть вычислительных кластеров сегодня имеют трехуровневую архитектуру. В рамках такой архитектуры многопроцессорная система строится как набор однородных вычислительных модулей, соединенных высокоскоростной сетью. Это – первый (верхний) уровень иерархии. Каждый вычислительный модуль является, в свою очередь, многопроцессорной системой с разделяемой памятью и образует второй уровень иерархии. Так как в современной кластерной системе, как правило, используются многоядерные процессоры, то мы получаем третий уровень иерархии. Еще одним источником многопроцессорных иерархий являются Grid-технологии, позволяющие объединять несколько различных кластеров в единую вычислительную систему. Подобная Grid-система будет иметь многоуровневую иерархическую структуру.

Важным классом приложений для иерархических многопроцессорных систем являются задачи, связанные с хранением и обработкой сверхбольших баз данных. В соответствие с этим актуальной является задача моделирования и анализа новых иерархических многопроцессорных архитектур для приложений баз данных.

Цель и задачи исследования. *Цель* данной работы состояла в построении математической модели иерархической многопроцессорной системы в контексте приложений баз данных, а также в разработке на ее основе методов и алгоритмов моделирования процессов параллельной обработки транзакций, которые могут быть применены для поиска и исследования перспективных аппаратных архитектур. Для достижения этой цели необходимо было решить следующие задачи:

1. Разработать математическую модель мультипроцессоров баз данных, включающую в себя модель аппаратной платформы, модель операционной среды, стоимостную модель и модель транзакций.
2. Разработать методы и алгоритмы, позволяющие реализовать предложенную модель на ЭВМ.
3. Спроектировать и реализовать эмулятор многопроцессорных иерархических машин баз данных.
4. Произвести проверку адекватности модели мультипроцессоров баз данных путем сравнения результатов, полученных на эмуляторе, с результатами, полученными на реальной параллельной СУБД.
5. При помощи эмулятора провести вычислительные эксперименты для поиска оптимальных аппаратных архитектур параллельных систем баз данных.

Методы исследования. Проведенные в работе исследования базируются на реляционной модели данных. Для построения моделей использовался математический аппарат, в основе которого лежат теория множеств, теория графов, теория алгоритмов и теория вероятности. При разработке программной системы применялись методы объектно-ориентированного проектирования и язык UML.

Научная новизна работы заключается в следующем:

1. Разработана математическая модель для описания иерархических многопроцессорных систем, ориентированная на перспективные суперкомпьютерные архитектуры и грид-системы, учитывающая специфику приложений баз данных.
2. Создана модель операционной среды, позволяющая моделировать работу приложения с интенсивным дисковым вводом-выводом на иерархической многопроцессорной системе.
3. Разработана стоимостная модель для оценки времени, расходуемого на обмены с дисками и передачу данных внутри иерархической многопроцессорной системы.
4. Предложена модель для описания выполнения смеси транзакций в многопроцессорной системе.

Теоретическая ценность работы состоит в том, что в ней дано формальное описание модели мультипроцессоров баз данных *DMM* (*Database Multiprocessor Model*), включающей в себя модель аппаратной платформы, модель операционной среды, стоимостную модель и модель транзакций. **Практическая ценность** работы заключается в том, что на базе предложенной модели *DMM* разработан эмулятор многопроцессорных иерархических машин баз данных *DMS* (*Database Multiprocessor Simulator*), позволяющий моделировать и исследовать эффективность различных иерархических многопроцессорных конфигураций в контексте задач баз данных класса OLTP.

Апробация работы. Основные положения диссертационной работы, разработанные модели, методы, алгоритмы и результаты вычислительных экспериментов докладывались автором на следующих международных и всероссийских научных конференциях:

- на Международной научной конференции «Высокопроизводительные вычисления, сети и коммуникационные системы (HPCNCS-07)» (9-12 июля 2007 г., Орландо, США);
- на Всероссийской научной конференции «Научный сервис в сети Интернет: масштабируемость, параллельность, эффективность» (21-26 сентября 2009 г., Новороссийск);
- на Всероссийской научной конференции «Научный сервис в сети Интернет: решение больших задач» (28 сентября 2008 г., Новороссийск);

- на Международной научной конференции «Параллельные вычислительные технологии» (29 января – 2 февраля 2007 г., Челябинск);
- на Втором весеннем коллоквиуме молодых исследователей в области баз данных и информационных систем (SYRCoDIS) (1–2 июня 2005 г., Санкт-Петербург);
- на Всероссийской научной конференции «Научный сервис в сети Интернет: технологии распределенных вычислений» (19-24 сентября 2005 г., Новороссийск);
- на Международной научной конференции «Суперкомпьютерные системы и их применение» (26–28 октября 2004 г., Минск).

Публикации. По теме диссертации опубликовано 6 печатных работ и получено одно свидетельство Роспатента об официальной регистрации программы для ЭВМ. Статья [1] опубликована в научном журнале «Автоматика и телемеханика», включенном ВАК в перечень журналов, в которых должны быть опубликованы основные результаты диссертаций на соискание ученой степени доктора и кандидата наук. В статье [1] П.С. Костенецкому принадлежит раздел 2 (стр. 113-117). В работах [4-6] Л.Б. Соколинскому принадлежит постановка задачи, П.С. Костенецкому принадлежат все полученные результаты.

Структура и объем работы. Диссертация состоит из введения, четырех глав, заключения и библиографии. Объем диссертации составляет 112 страниц, объем библиографии – 136 наименований.

Содержание работы

Во введении приводится обоснование актуальности темы, формулируются цели работы, ее новизна и практическая значимость; приводится обзор работ по тематике исследования и кратко излагается содержание диссертации.

Первая глава, «Обработка транзакций в многопроцессорных иерархиях», посвящена описанию и исследованию многопроцессорных вычислительных систем с иерархической архитектурой. *Иерархическая многопроцессорная система* – это многопроцессорная система, в которой процессоры объединяются в единую систему с помощью соединительной сети, имеющей иерархическую структуру и обладающую свойствами однородности по горизонтали и неоднородности по вертикали. *Однородность по горизонтали* означает, что в пределах одного уровня иерархии скорость обменов между двумя процессорами является постоянной величиной, независимо от того в каком поддереве иерархии эти процессоры находятся. *Неоднородность по вертикали означает*, что скорость обменов на разных уровнях иерархии существенно различается и этот факт должен учитываться в алгоритмах СУБД для многопроцессорных иерархий.

Основой параллельной обработки запросов в реляционных системах баз данных является *фрагментный параллелизм*. Данная форма параллелизма предполагает *фрагментацию* отношения, являющегося аргументом реляционной операции, по дискам многопроцессорной системы. Обработка запроса в параллельной СУБД на вычислительном кластере состоит из трех этапов. На первом этапе SQL-запрос передается пользователем на выделенную host-машину, где транслируется в некоторый последовательный физический план. На втором этапе последовательный физический план преобразуется в параллельный план, представляющий собой совокупность параллельных агентов. Это достигается путем вставки специального оператора обмена **exchange** в соответствующие места дерева запроса. На третьем этапе параллельные агенты пересылаются с host-машины на соответствующие вычислительные узлы, где интерпретируются исполнителем запросов. Результаты выполнения агентов объединяются корневым оператором **exchange** на нулевом узле, откуда передаются на host-машину. Роль host-машины может играть любой узел вычислительного кластера.

В конце главы 1 произведен обзор наиболее известных параллельных вычислительных моделей с общей памятью RAM, PRAM, YPRAM, HPRAM, параллельных вычислительных моделей с распределенной памятью BSP, LogP, LogGP, параллельных вычислительных моделей с иерархией памяти Memory logP,DRAM(h,k), H-BSP, а так же рассмотрено использование иерархической сети Петри для моделирования параллельных процессов. Проведенный обзор показывает, что указанные модели не учитывают иерархическую структуру соединительной сети. Кроме этого, все рассмотренные модели не учитывают специфику параллельных систем баз данных, использующих фрагментный параллелизм. В связи с этим, актуальной является задача разработки новых моделей и методов анализа иерархических многопроцессорных систем баз данных.

Во второй главе, «Модель мультипроцессоров баз данных», предлагается DMM-модель для описания иерархических многопроцессорных систем баз данных. Модель DMM включает в себя модель аппаратной платформы, модель операционной среды, стоимостную модель и модель транзакций. ***Модель аппаратной платформы*** строится следующим образом. Множество модулей многопроцессорной системы разбивается на три непересекающихся подмножества:

$$\mathfrak{M} = \mathfrak{P} \cup \mathfrak{D} \cup \mathfrak{N},\ \mathfrak{P} \cap \mathfrak{D} = \varnothing,\ \mathfrak{P} \cap \mathfrak{N} = \varnothing,\ \mathfrak{D} \cap \mathfrak{N} = \varnothing .$$

$\mathfrak{P}$ обозначает множество *процессорных* модулей, $\mathfrak{D}$ – множество *дисковых* модулей, $\mathfrak{N}$ – множество модулей *сетевых концентраторов*.

DM-графом будем называть связный ациклический граф (свободное дерево) $\mathfrak{W}(\mathfrak{M},\mathfrak{E})$, в котором множество ребер $\mathfrak{E}$ удовлетворяет следующим ограничениям:

$$\mathfrak{P} \cup \mathfrak{D} \neq \varnothing, \mathfrak{N} \neq \varnothing; \quad (1)$$

$$\forall P \in \mathfrak{P}\left(\forall M \in \mathfrak{M}\left(\left(\exists E(A,A') \in \mathfrak{E}\left(\begin{array}{l}(\text{init}(A) = P \wedge \text{fin}(A) = M)\\ \vee(\text{init}(A') = P \wedge \text{fin}(A') = M)\end{array}\right)\right) \Rightarrow M \in \mathfrak{N}\right)\right); \quad (2)$$

$$\forall D \in \mathfrak{D}\left(\forall M \in \mathfrak{M}\left(\left(\exists E(A,A') \in \mathfrak{E}\left(\begin{array}{l}(\text{init}(A) = D \wedge \text{fin}(A) = M)\\ \vee(\text{init}(A') = D \wedge \text{fin}(A') = M)\end{array}\right)\right) \Rightarrow M \in \mathfrak{N}\right)\right); \quad (3)$$

$$\forall M \in \mathfrak{M}\left(\left(\forall E(A,A'), \tilde{E}(\tilde{A},\tilde{A}')\left(\left(\begin{array}{l}(\text{init}(A) = M \vee \text{init}(A') = M)\\ \wedge(\text{init}(\tilde{A}) = M \vee \text{init}(\tilde{A}') = M)\end{array}\right) \Rightarrow (E = \tilde{E})\right)\right) \Rightarrow (M \in \mathfrak{D} \cup \mathfrak{P})\right). \quad (4)$$

Здесь каждое ребро E представляется в виде двух противоположно направленных дуг A и A'; $\text{init}(A)$ – узел, являющийся начальным для дуги A; $\text{fin}(A)$ – узел, являющийся конечным для дуги A.

DM-деревом будем называть *DM-граф*, в котором имеется выделенная вершина $\bar{N} \in \mathfrak{N}$, называемая *корневым* сетевым концентратором.

Уровень узла дерева – это длина пути от корня до узла. Уровень корня всегда равен нулю. *Высота* $h(T)$ дерева T – это максимальный уровень его узлов. *m-й ярус* дерева – множество узлов дерева, на уровне m от корня дерева. Узел M' является *сыном* узла M, тогда, и только тогда, когда узлы M' и M являются смежными и уровень M' на единицу больше уровня M.

DM-дерево является *регулярным*, если выполняются следующие условия:

- процессорные и дисковые модули не могут иметь сыновей, то есть они всегда являются *листьями* (концевыми узлами);
- степень любого узла, не являющегося *листом*, больше, либо равна двум.

Пусть задано регулярное *DM*-дерево T с корнем $\bar{N}$. Пусть $\{M_1,\ldots,M_k\}$ – множество узлов 1-го яруса дерева T. Определим *поддерево* T_{M_i} с корнем в узле M_i $(1 \le i \le k)$ конструктивно следующим образом:

1) для всех j таких, что $1 \le j \le k$ и $j \neq i$, удалить все листья, и инцидентные им ребра, от которых есть путь до корня $\bar{N}$, имеющий длину больше единицы и проходящий через узел M_j;
2) если на шаге 1 было удалено не менее одного листа, снова перейти на шаг 1, в противном случае перейти на шаг 3;
3) для всех j таких, что $1 \le j \le k$ и $j \neq i$, удалить узел M_j и инцидентное ему ребро;
4) удалить узел $\bar{N}$ и инцидентное ему ребро.

Теорема 1. Пусть задано регулярное *DM*-дерево T с высотой больше единицы. Пусть $\{M_1,\ldots,M_k\}$ – множество узлов 1-го яруса дерева T. Тогда для любого i, $1 \le i \le k$, поддерево T_{M_i} является регулярным *DM*-деревом ли-

бо тривиальным деревом, состоящим из одного узла M, такого, что $M \in \mathfrak{P} \cup \mathfrak{D}$.

Теорема 2. Пусть $\mathcal{T}$ – регулярное *DM*-дерево. Тогда $h(\mathcal{T}) < |\mathfrak{P} \cup \mathfrak{D}|$, где $\mathfrak{P}$ – множество процессорных модулей, $\mathfrak{D}$ – множество дисковых модулей дерева $\mathcal{T}$.

С каждым узлом $M \in \mathfrak{M}(\mathcal{T})$ в *DM*-дереве $\mathcal{T}$ связывается *весовая функция* $\eta(M)$, которая определяет время, необходимое узлу для обработки некоторой порции данных.

DM-деревья $\mathcal{T}_1$ и $\mathcal{T}_2$ называются *изоморфными*, если существуют взаимно однозначное отображение f множества узлов $\mathfrak{M}(\mathcal{T}_1)$ дерева $\mathcal{T}_1$ на множество узлов $\mathfrak{M}(\mathcal{T}_2)$ дерева $\mathcal{T}_2$ и взаимно однозначное отображение g множества ребер $\mathfrak{E}(\mathcal{T}_1)$ дерева $\mathcal{T}_1$ на множество ребер $\mathfrak{E}(\mathcal{T}_2)$ дерева $\mathcal{T}_2$ такие, что:

(1) ребро E инцидентно узлам M и M' в дереве $\mathcal{T}_1$ тогда и только тогда, когда ребро $g(E)$ инцидентно узлам $f(M)$ и $f(M')$ в дереве $\mathcal{T}_2$;

(2) $P \in \mathfrak{P}(\mathcal{T}_1) \Leftrightarrow f(P) \in \mathfrak{P}(\mathcal{T}_2)$;

(3) $D \in \mathfrak{D}(\mathcal{T}_1) \Leftrightarrow f(D) \in \mathfrak{D}(\mathcal{T}_2)$;

(4) $N \in \mathfrak{N}(\mathcal{T}_1) \Leftrightarrow f(N) \in \mathfrak{N}(\mathcal{T}_2)$;

(5) $\eta(f(M)) = \eta(M)$.

Упорядоченную пару отображений $q = (f,g)$ будем называть *изоморфизмом* *DM*-дерева $\mathcal{T}_1$ на *DM*-дерево $\mathcal{T}_2$. Под *уровнем поддерева* дерева $\mathcal{T}$ мы будем понимать уровень корня этого поддерева в дереве $\mathcal{T}$. Два поддерева одного уровня называются *смежными*, если их корни являются братьями, то есть в дереве существует узел, являющийся общим родителем по отношению к корневым узлам данных поддеревьев.

Будем называть регулярное *DM*-дерево $\mathcal{T}$ высоты $h \geq 2$ *симметричным*, если любые два смежных поддерева уровня l ($0 < l < h(\mathcal{T})$) имеют одинаковую высоту и являются изоморфными.

Теорема 3. Пусть $\mathcal{T}$ – симметричное *DM*-дерево с корнем $\bar{N}$. Тогда $h(\mathcal{T}) \leq \log_2(|\mathfrak{P} \cup \mathfrak{D}|)$, где $\mathfrak{P}$ – множество процессорных модулей, $\mathfrak{D}$ – множество дисковых модулей дерева $\mathcal{T}$.

Модель аппаратной платформы параллельной системы баз данных представляется в виде регулярного *DM*-дерева с точностью до изоморфизма.

```
if r(P) < s_r then
 Поместить пакет E с адресом P в
очередь D;
 r(P)++;
else
 wait;
end if
```

Рис. 1.

```
if w(P) < s_w then
 Поместить пакет E с адресом D в
очередь родительского сетевого
концентратора;
 w(P)++;
else
 wait;
end if
```

Рис. 2.

```
Извлечь пакет E из очереди N;
if α(E) ∉ T(N) then
  Поместить E в очередь F(N);
else
  Найти максимальное поддерево U
    дерева T(N), содержащее α(E);
  if T(α(E))≠U then
    if α(E)∈P then
      r(α(E))-- ;
    else
      Поместить E в очередь α(E);
    end if
  else
    Поместить E в очередь R(U);
  end if
end if
```

Рис. 3.

```
Извлечь пакет E из очереди D;
if α(E)∈D then
 w(β(E))-- ;
else
 Поместить E в очередь родитель-
ского узла;
end if
```

Рис. 4.

Модель операционной среды строится следующим образом. Наименьшей неделимой единицей обработки данных является *пакет*. С каждым дисковым модулем и модулем сетевого концентратора в модели *DMM* ассоциируется *очередь*, в которую помещаются пересылаемые пакеты. Модель *DMM* допускает асинхронный обмен пакетами в том смысле, что процессорный модуль может инициализировать новый обмен, не дожидаясь завершения предыдущего. Процессорный модуль может иметь в каждый момент не более s_r незавершенных операций чтения и s_w незавершенных операций записи. Время работы системы в модели *DMM* делится на дискретные промежутки, называемые *тактами*. Для произвольного $M \in \mathfrak{M}$ введем следующие обозначения: $F(M)$ – родительский модуль узла M, $T(M)$ – поддерево с корнем в вершине M.

Псевдокод *операции чтения* представлен на рис. 1. Здесь $r(P)$ – количество незавершенных операций чтения процессора P, s_r – максимальное допустимое число незавершенных операций чтения.

На рис. 2 представлен псевдокод алгоритма инициирующего *запись*. Здесь $w(P)$ – количество незавершенных операций записи процессора P, s_W – максимальное допустимое число незавершенных операций записи.

Модуль сетевого концентратора $N \in \mathfrak{N}$ осуществляет перманентную передачу пакетов по соединительной сети, выполняя алгоритм, изображенный на рис. 3. Здесь E – пакет, $\alpha(E)$ – адресат пакета E, $T(N)$ – поддерево с корнем N, $F(N)$ – родительский модуль узла N, $\mathfrak{P}$ – множество процессорных модулей, $r(P)$ – количество незавершенных операций чтения процессора P, $R(U)$ – корень поддерева U.

Дисковый модуль $D \in \mathfrak{D}$ осуществляет перманентное чтение и запись пакетов, выполняя алгоритм, изображенный на рис. 4. Здесь $\beta(E)$ – отправитель пакета, $w(\beta(E))$ – количество незавершенных операций записи отправителя.

Такт определяется как следующая последовательность действий:

1) каждый модуль сетевого концентратора обрабатывает все пакеты, ожидающие передачи;
2) каждый процессорный модуль выполняет одну операцию обмена с дисками, если только он не находится в состоянии ожидания завершения предыдущей операции;
3) каждый дисковый модуль обрабатывает один пакет из своей очереди.

Стоимостная модель определяется следующим образом. С каждым модулем $M \in \mathfrak{M}$ связывается *коэффициент трудоемкости* $h_M \in \mathbb{R}$, $1 \le h_M < +\infty$. Полагаем $h_P = 1, \forall P \in \mathfrak{P}$. Для модуля сетевого концентратора $N \in \mathfrak{N}$ вводим функцию помех $f_N(m_i^N) = \left\lceil \frac{m_i^N}{\tau_N} \right\rceil \delta_N$. Здесь m_i^N обозначает число пакетов, проходящих через N на i-том такте; $0 \le \delta_N \le 1$ – масштабирующий коэффициент; $\tau_N > 1$ – пороговое значение (максимальное число одновременно передаваемых пакетов, не вызывающее замедление работы модуля сетевого концентратора). Время, требуемое модулю сетевого концентратора N для выполнения i-того такта, вычисляется по формуле $t_i^N = h_N f_N(m_i^N)$, $\forall N \in \mathfrak{N}$. Общее время работы системы, затраченное на обработку смеси транзакций в течении k тактов, вычисляется по формуле

$$t = \sum_{i=1}^{k} \left(\max_{N \in \mathfrak{N}} (t_i^N) + \max_{N \in \mathfrak{D}} (h_D) \right).$$

Модель транзакций строится следующим образом. Последовательная транзакция Z моделируется путем задания двух групп процессов ρ и ω: $Z = \{\rho, \omega\}$, $\rho \cup \omega \neq \varnothing$. Группа ρ включает в себя *читающие* процессы, группа ω – *пишущие* процессы. Для каждого читающего и пишущего процесса задается *количество обращений к диску*, которое он должен произвести. После того, как процесс выполнил все обращения, он удаляется из множества ρ или ω соответственно. Транзакция $Z = \{\rho, \omega\}$ считается завершенной, когда $\rho \cup \omega = \varnothing$. Каждая транзакция $Z = \{\rho, \omega\}$ разбивается на конечную последовательность шагов: $Z_1, \ldots, Z_s$. Здесь s обозначает количество шагов транзакции. В соответствие с этим каждый процесс $x \in \rho \cup \omega$ транзакции Z разбивается на s шагов: $x_1, \ldots, x_s$. Каждый шаг x_i $(i = 1, \ldots, s)$ описывается тройкой чисел (n_i, p_i, d_i), где d_i – номер диска, с которым процесс x обменивается данными на i-том шаге; n_i – количество обращений к диску;

p_i – вероятность обращения процесса x к диску D_{d_i} на каждом такте работы эмулятора в течении i-го шага.

Параллельная транзакция Z, выполняемая на l процессорных модулях, моделируется путем задания l последовательных транзакций $Z^1,\ldots,Z^l$, каждая из которых выполняется на отдельном процессорном модуле. Алгоритм, моделирующий выполнение отдельной транзакции на процессорном модуле, определяется следующим образом. Пусть задана транзакция $Z=\{\rho,\omega\}$, состоящая из s шагов: $Z_1,\ldots,Z_s$. Обозначим через X множество всех читающих и пишущих процессов, составляющих транзакцию Z: $X=\rho\cup\omega$. Пусть $X=\{x^1,\ldots,x^r\}$. Каждый процесс x^j $(j=1,\ldots,r)$ делится на s шагов: $x_1^j,\ldots,x_s^j$. Пусть выполнение транзакции Z находится на i-том шаге. Предположим, что процесс x^j *получил управление*. Пусть i-тый шаг процесса x^j имеет вид $x_i^j=(n_i^j,p_i^j,d_i^j)$. Предполагается, что процесс x^j может получить управление на i-том шаге только в случае $n_i^j>0$. Тогда на текущем такте выполняется следующая последовательность действий.

1. Значение n_i^j уменьшается на единицу.
2. Процесс x^j инициирует операцию обмена с диском, имеющим номер d_i^j.
3. Если $\sum_{j=1}^{r} n_i^j=0$ и $i<s$, то увеличить i (номер текущего шага транзакции) на единицу.

Модель *DMM* допускает выполнение на одном процессоре *смеси последовательных транзакций* $\left\{Z^i \middle| i=1,\ldots,k\right\}$ в режиме разделения времени. При этом каждая транзакция Z^i $(i=1,\ldots,k)$ представляется своей собственной парой групп читающих и пишущих процессов: $Z^i=\{\rho^i,\omega^i\}$. Все множество процессов моделирующих выполнение смеси транзакций на некотором процессоре $P\in\mathfrak{P}$ определяется следующим образом: $\Phi_P=\bigcup_{i=1}^{k}(\rho^i\cup\omega^i)$. При запуске новой транзакции на процессоре P, в множество Φ_P динамически добавляются читающие и пишущие процессы, представляющие данную транзакцию. Если какой-либо процесс завершается, то он удаляется из множества Φ_P.

Выполнение смеси транзакций Φ_P на некотором процессоре P организуется следующим образом. На каждом такте работы эмулятора процессор P должен инициализировать одну операцию обмена с дисками. В соответствие с этим, процессор должен выбрать некоторый процесс $x\in\Phi$ и произвести

операцию чтения с диска или записи на диск, ассоциированный с x на текущем шаге. Такой процесс называется *активным*. Все процессы из множества Ф организуются в список. Вводится указатель на текущий элемент списка (его начальная позиция может быть произвольной). Для определения активного процесса используется алгоритм, изображенный на рис. 5.

```
for (P=P.begin();P!=P.end();P++){
  prob = 0;
  for(x=P.Ф.begin();x!=P.Ф.end(); x++){
    if (n(x,i(x)) == 0) continue;
    prob += p(x,i(x));
    if (g(prob)) return x;
  };
};
```

Рис. 5.

Здесь $\mathfrak{P}$ – множество всех процессорных модулей *DM*-дерева; *s(x)* – общее количество шагов процесса *x*; *n(x,i)* – количество обращений к диску, которые осталось выполнить процессу x на шаге i; *p(x,i)* – вероятность обращения процесса x к диску при выполнении i-го шага; g – функция срабатывания, вычисляемая следующим образом. Для каждого шага i процесса $x \in \Phi$ известна вероятность p_i^x обращения процесса x к диску, ассоциированному с этим процессом на i-том шаге. Функция срабатывания $g(p_i^x) = G$ определяется как функция дискретной случайной величины G, закон распределения которой задается следующим рядом распределения:

G	1	0
P	p_i^x	$1-p_i^x$

В третьей главе, «Эмулятор многопроцессорных иерархических машин баз данных», описывается процесс проектирования и реализации программной системы *DMS* (*Database Multiprocessor Simulator*), представляющей собой эмулятор многопроцессорных иерархических машин баз данных. Проектирование эмулятора производилось при помощи унифицированного языка моделирования UML 2.0. Требования к эмулятору *DMS* зафиксированы при помощи модели вариантов использования. Приводятся диаграммы классов модели аппаратной платформы и модели операционной среды. Построены диаграммы последовательности для операций чтения и записи, диаграмма последовательности работы модуля сетевого концентратора и диаграмма последовательности работы дискового модуля. Для представления информации о мультипроцессорных иерархиях разработан *язык описания многопроцессорных иерархических конфигураций HMML*, базирующийся на синтаксисе расширяемого языка разметки *XML*. Для автоматизации процесса создания входных файлов разработан *HMML*-генератор. Эмулятор *DMS* был реализован на языке C++. Исходные тексты эмулятора свободно доступны в сети Интернет по адресу http://kps.susu.ru/science/dms/. На эмулятор *DMS* получено свидетельство Роспатента об официальной регистрации программы для ЭВМ.

В четвертой главе, «Вычислительные эксперименты», отражены результаты вычислительных экспериментов, выполненных с использованием

эмулятора многопроцессорных иерархических машин баз данных *DMS*, разработанного на базе модели *DMM*. В ходе вычислительных экспериментов использовалась транзакция, в которой выполнялось естественное соединение двух отношений R и S, имеющих один общий атрибут. Предполагалось, что оба отношения R и S фрагментированы не по общему атрибуту. Процентное содержание «своих» кортежей во фрагментах отношений R и S определялось коэффициентом перекоса μ. Для вычисления соединения использовался алгоритм *MHJ*.

Для *подтверждения адекватности* модели *DMM* была использована параллельная СУБД «Омега», установленная на вычислительном кластере «СКИФ Урал». Размеры отношений R и S составляли соответственно 6 000 000 и 600 000 000 записей. Результаты выполнения транзакций приведены на СУБД «Омега» приведены на рис. 6. С помощью эмулятора *DMS* было произведено моделирование выполнения таких же транзакций на *DM*-дереве, описывающем вычислительный кластер «СКИФ Урал». Результаты этих экспериментов приведены на 7. Сравнение показывает, что эмулятор *DMS* адекватно моделирует выполнение транзакций на вычислительном кластере для любых значений коэффициента перекоса μ. Это подтверждает адекватность модели *DMM*.

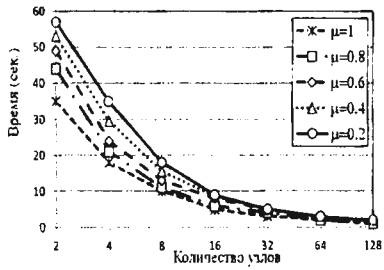


Рис. 6.

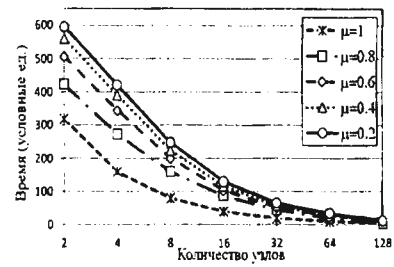


Рис. 7.

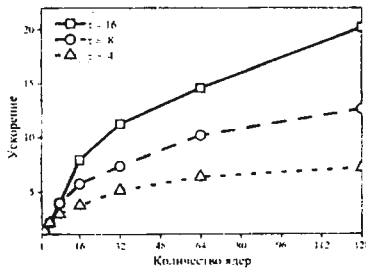


Рис. 8.

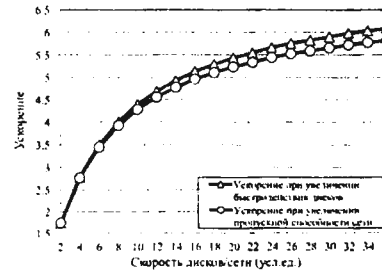


Рис 9.

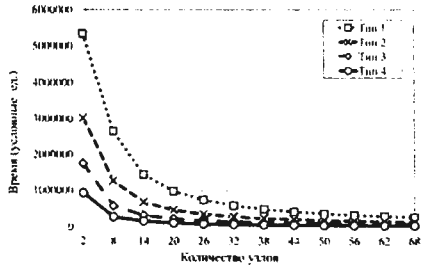


Рис. 10.

На рис. 8 показано влияние пропускной способности системной шины на общую производительность SMP-узла на многоядерных процессорах. Эти результаты были получены с помощью эмулятора *DMS*. Данный эксперимент показывает, что при увеличении числа процессорных ядер и дисков, системная шина становится узким местом SMP архитектуры. Когда количество процессоров и дисков становится больше 32, для обеспечения требуемой производительности, требуются использование дорогостоящих кросс-коммутаторов.

На рис. 9 показано влияние интерконнекта и дисков на масштабирование системы. Эти результаты также были получены с помощью эмулятора *DMS*. Эксперимент показывает, что в вычислительных узлах иерархических многопроцессорных систем баз данных целесообразно устанавливать диски и коммуникационное оборудование приблизительно одинаковой производительности.

Табл. 1.

Тип узла	**Основные характеристики**	**Цена с инфраструктурой** (тыс. руб.)
1	Одноядерный процессор Intel Xeon DP EM64T и один диск SAS 73.4 Гб	266
2	Двуядерный процессор Intel Xeon E3110 и два диска SAS 73.4 Гб	298
3	Два двуядериых процессора Intel Xeon E3110 и четыре диска SAS 73.4 Гб	325
4	Два четырехъядерных процессора Intel Xeon E5472 и восемь дисков SAS 73.4 Гб	395

На рис. 10 показаны зависимости, полученные с помощью эмулятора *DMS*, которые можно использовать для оценки эффективности решений по приобретению и модернизации вычислительных кластеров для приложений баз данных. Указанные зависимости были получены при моделировании четырех типов узлов, приведенных в табл. 1. Предположим, например, что в бюджете некоторой организации в 2010 г. имеется 10 миллионов рублей на приобретение вычислительного кластера для приложений баз данных. Необходимо подобрать на эту сумму наиболее производительную аппаратную архитектуру, варьируя конфигурацию и количество узлов системы. На основе зависимостей на рис. 10 можно определить, что при заданной цене, наибольшую производительность будет обеспечивать конфигурация, состоящая из 25 узлов типа 4.

В заключении суммируются основные результаты диссертационной работы, выносимые на защиту, приводятся данные о публикациях и апробациях автора по теме диссертации, и рассматриваются направления дальнейших исследований.

Основные результаты диссертационной работы

На защиту выносятся следующие новые научные результаты.

1. Разработана новая математическая модель мультипроцессоров баз данных *DMM* (*Database Multiprocessor Model*), включающая в себя модель аппаратной платформы, модель операционной среды, стоимостную модель и модель транзакций.
2. Разработаны методы и алгоритмы, позволяющие реализовать модель *DMM* на ЭВМ.
3. Разработан эмулятор многопроцессорных иерархических машин баз данных *DMS* (*Database Multiprocessor Simulator*), реализующий модель *DMM*. Произведена проверка адекватности модели мультипроцессоров баз данных путем сравнения результатов, полученных на эмуляторе *DMS*, с результатами, полученными на реальной параллельной СУБД.
4. С использованием эмулятора *DMS* проведены вычислительные эксперименты, позволяющие находить оптимальные аппаратные архитектуры параллельных систем баз данных.

Публикации по теме диссертации

Статьи, опубликованные журналах из списка ВАК

1. *Костенецкий П.С., Лепихов А.В., Соколинский Л.Б.* Технологии параллельных систем баз данных для иерархических многопроцессорных сред // Автоматика и телемеханика. 2007. No. 5. С. 112-125.

Другие публикации

2. *Костенецкий П.С.* Моделирование параллельных систем баз данных для вычислительных кластеров // Научный сервис в сети Интернет: масштабируемость, параллельность, эффективность: Труды Всероссийск. науч. конф. (21-26 сентября 2009 г., г. Новороссийск). М.: Изд-во МГУ. 2009. С. 300-304.
3. *Костенецкий П.С.* Разработка эмулятора виртуальных мультипроцессоров баз данных // Параллельные вычислительные технологии: Труды международной науч. конф. (29 января - 2 февраля 2007 г., г. Челябинск). Челябинск.: Изд-во ЮУрГУ. 2007. С. 285.
4. *Kostenetskiy P.S., Sokolinsky L.B.* Analysis of Hierarchical Multiprocessor Database Systems // Proceedings of the 2007 International Conference on High Performance Computing, Networking and Communication Systems (HPCNCS-07), July 9-12 2007, Orlando, FL, USA. ISRST. 2007. P. 245–251.

5. *Костенецкий П.С., Соколинский Л.Б.* Моделирование иерархических архитектур параллельных систем баз данных // Научный сервис в сети Интернет: технологии распределенных вычислений: Труды всероссийск. науч. конф. (19–24 сентября 2005 г., г. Новороссийск). М.: Изд-во МГУ. 2005. С. 21–24.

6. *Костенецкий П.С., Соколинский Л.Б.* Моделирование и анализ иерархических архитектур параллельных систем баз данных // Суперкомпьютерные системы и их применение (SSA'2004) (26–28 октября 2004 г., г. Минск, Республика Беларусь), тез. докл. междунар. науч. конф., Минск: ОИПИ НАН Беларуси. 2004. С. 116–120.

Свидетельство о регистрации программы

7. *Костенецкий П.С. Соколинский Л.Б.* Свидетельство Роспатента об официальной регистрации программы для ЭВМ «Эмулятор параллельных систем баз данных» №2009616225 от 11.11.2009.

Работа выполнялась при поддержке *Российского фонда фундаментальных исследований* (проекты 06-07-89148, 09-07-00241-А).

Подписано в печать 28.01.2010
Формат 60x84 1/16. Бумага офсетная.
Печать офсетная. Усл. печ. л. 1,0. Уч.-изд. л. 1,2.
Тираж 100 экз.

Типография «Фотохудожник»
454111, г. Челябинск, ул. Свободы, 155/1