Вокруг проекта «Омега»

Текст Михаил Цимблер

Параллельная система управления базами данных – это аппаратно-программный комплекс, в котором СУБД, функционирующая на платформе многопроцессорной вычислительной системы, обеспечивает хранение сверхбольших баз данных и эффективную обработку параллельных транзакций в режиме «24 часа в сутки, 7 дней в неделю»

Примерами приложений, характеризующихся сверхбольшим объемом храни-мых данных, являются электронная коммерция, электронные библиотеки, геоинформационные системы, мультимедиа-архивы, социальные сети, научные базы данных и многое другое.

Одной из самых больших и быстро наполняемых научных баз данных является база данных проекта WLCG (Worldwide Large Hadron Collider Computing Grid). Главной целью проекта WLCG является использование грид-среды для обработки экспериментальных данных, получаемых с Большого адронного коллайдера (Large Hadron Collider, LHC) Европейского центра ядерных исследований (CERN). Поток экспериментальных данных, который необходимо обрабатывать, составляет около 15 петабайт (1 Пб = 210 Тб) в год.

Другим примером сверхбольшой базы данных является база данных проекта системы SkyServer проекта SDSS (Sloan Digital Sky Survey). Данный проект предполагает создание виртуальной обсерватории, доступной через Интернет. База данных проекта должна объединить в себе полную информацию о наблюдениях всех участков звездного неба различными обсерваториями мира. Суммарный объем данных, поступающих с телескопов, составляет около 5 петабайт в год.

В настоящее время в России начаты работы по двум масштабным космическим экспериментам «Лира» и «Свеча», целью которых является высокоточный многоцветный фотометрический обзор звезд всего неба вплоть до 16-17-ти звездной величины. В обзор войдут около 400 млн звезд. Измерения ведутся в 10 спектральных полосах от 0.2 до 1.0 мкм с борта Российского сегмента МКС. Для хранения базы данных проекта «Лира» потребуется дисковое пространство суммарным объемом до 1 петабайта, а для хранения данных проекта «Свеча» – размером несколько экзабайт (1 Эб = 103 Пб).

Еще одним примером сверхбольших баз данных являются базы данных проекта EOS/DIS (Earth Observation System / Data Information System), разрабатываемого агентством NASA в США. Система наблюдения земли EOS включает в себя множество спутников, которые собирают информацию, необходимую для изучения долгосрочных тенденций состояния атмосферы, океанов, земной поверхности. Начиная с 1998 года спутники поставляют информацию в объеме 0.3 петабайта в год. К 2010 году общий объем поддерживаемых в системе данных достиг 20 Петабайт.

Современные параллельные СУБД – это, как правило, дорогой коммерческий продукт, который по карману только крупной организации; свободно распространяемые продукты в этой области – скорее исключение.

Годовая лицензия на СУБД Oracle RAC (Real Application Cluster) для параллельной обработки запросов на кластерных системах стоит $25 000. СУБД Greenplum продается по бессрочной лицензии, цена которой $16 000 за процессорное ядро или $70 000 за 1 Тб обслуживаемой базы данных, а годовая поддержка продукта стоит 22% от суммы покупки. СУБД Teradata предлагается по ценам от $16 000 за 1 Тб обслу-

живаемой базы данных. Компания Amazon Web Services предоставляет услугу «Virtual HPC as a Service» для работы с базами данных. Данная услуга предполагает развертывание на виртуальной машине предварительно подготовленного образа, содержащего установленную и настроенную параллельную СУБД. Стоимость этой услуги складывается из аренды машинного времени, дискового пространства для данных и оплаты операций чтения-записи: соответственно $0.11-3.10 в час в зависимости от конфигурации машины, $0.10 за 1 Гб в месяц и $0.10 за каждый миллион I/O-операций.

Свободно распространяемая система Apache Hadoop представляет собой реализацию концепции MapReduce компании Google с открытыми исходными кодами на языке Java. Hadoop поддерживает выполнение распределенных приложений, работающих на больших кластерных системах, и позволяет этим приложениям легко масштабироваться до уровня тысяч узлов и петабайт данных. Однако Hadoop является инструментальным средством прикладного Java-программиста и не поддерживает напрямую технологии баз данных (реляционные таблицы, язык запросов SQL и др.).

В рамках научного проекта ParGRES, выполняемого под руководством Патрика Валдурица в Национальном институте информатики и автоматизации Франции (INRIA, Institut National de Recherche en Informatique et en Automatique), разрабатывается параллельная СУБД, предназначенная для обработки OLAP-запросов. СУБД ParGRES представляет собой надстройку (middleware), которая управляет экземплярами свободной СУБД PostgreSQL, запускаемыми на узлах кластерной системы.

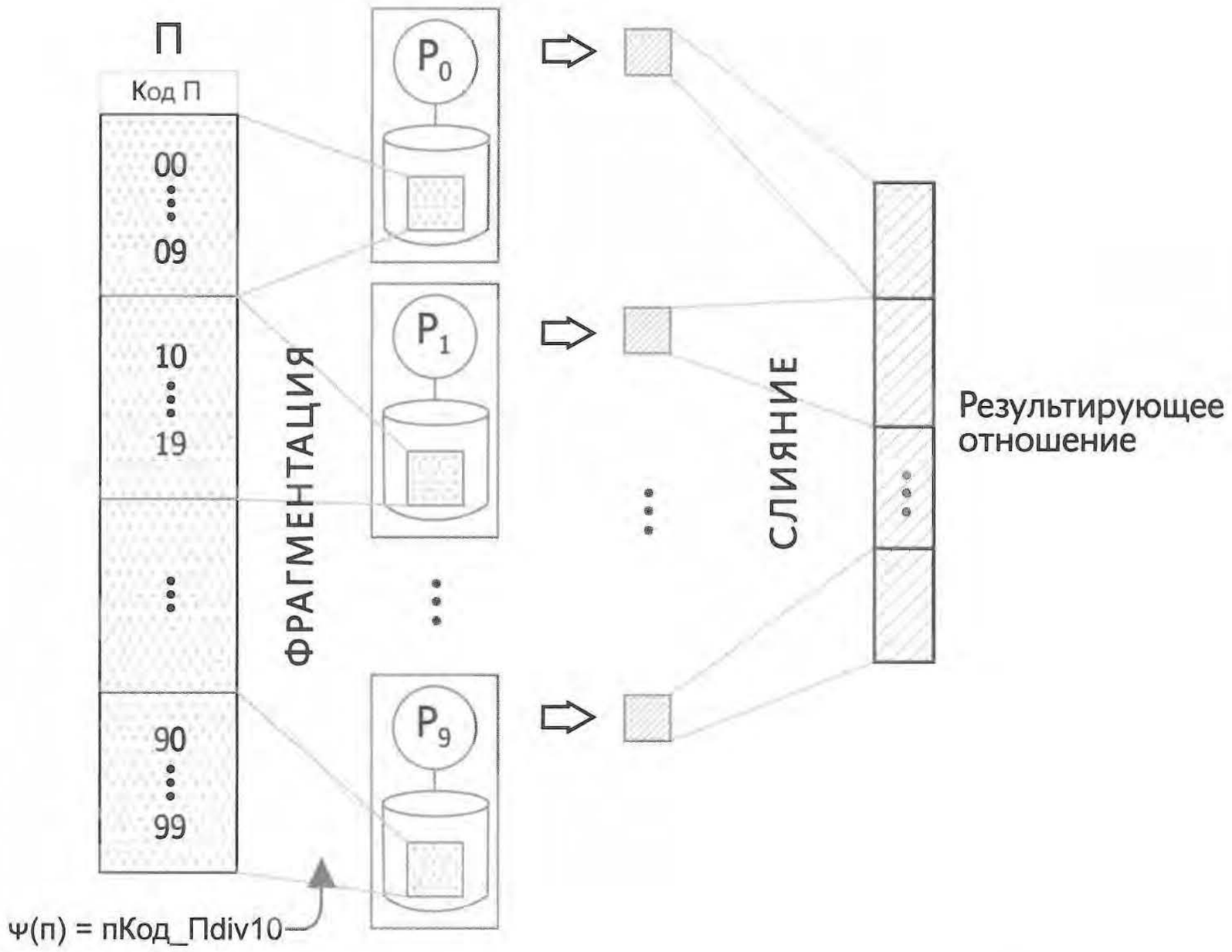
На факультете Вычислительной математики и информатики Южно-Уральского государственного университета (Челябинск) выполняется ряд проектов, посвященных параллельным и распределенным системам баз данных. Идейным вдохновителем и научным руководителем работ стал декан факультета Леонид Борисович Соколинский. В проектах участвуют студенты, магистранты, аспиранты и преподаватели кафедры системного программирования факультета ВМИ, а также сотрудники Лаборатории суперкомпьютерного моделирования ЮУрГУ.

Базовый проект – научный проект «Омега» (http://omega.susu.ru), целью которого является разработка прототипа параллельной СУБД для кластерных вычислительных и грид-систем. В рамках этого проекта многочисленные идеи, методы и алгоритмы параллельной обработки баз данных проходят отладку и тестирование в прототипе, чтобы далее быть внедренными в полноценные коммерциализуемые программные продукты.

В основе СУБД «Омега» лежит идея использования фрагментного параллелизма. Каждая реляционная таблица делится на горизонтальные фрагменты. Фрагменты таблицы распределяются по различным процессорным узлам многопроцессорной системы. При этом на каждом узле многопроцессорной системы запускается параллельный агент (ядро СУБД), обрабатывающий запросы пользователей. Один и тот же запрос параллельно выполняется на всех процессорных узлах, и затем полученные фрагменты результата сливаются в результирующую таблицу.

Рисунок показывает простейший пример применения фрагментного параллелизма: запрос на выборку строк из одной таблицы. В примере задействована база данных Поставки из классического учебника К. Дж. Дейта по базам данных. Таблица «Поставщики» фрагментируется по колонке Код_П на основе функции фрагментации Ψ(Поставщики) = Код_П div 10, где div – операция деления нацело. Функция фрагментации для каждой записи таблицы вычисляет номер процессорного узла, на котором должна быть размещена эта запись.

Однако естественному желанию



получить от параллельной СУБД ускорение, равное количеству задействованных узлов, мешает ряд препятствий. Во-первых, при выполнении запроса, в котором данные выбираются из двух и более таблиц, могут потребоваться межпроцессорные обмены. Такой пример приведен на следующем рисунке.

Помимо текста запроса на языке SQL на рисунке показано внутрен-

П (поставщики)

Код_П*	Имя_П	Город
23	Иванов И.И.	Москва
14	Петров П.П.	Самара

Д (детали)

Код_Д*	Имя_Д	Цвет
3	Гайка	Красный
7	Болт	Синий

ПД (поставки)

Код_ПД*	Имя_П*	Код_Д*
1	14	7
2	23	3

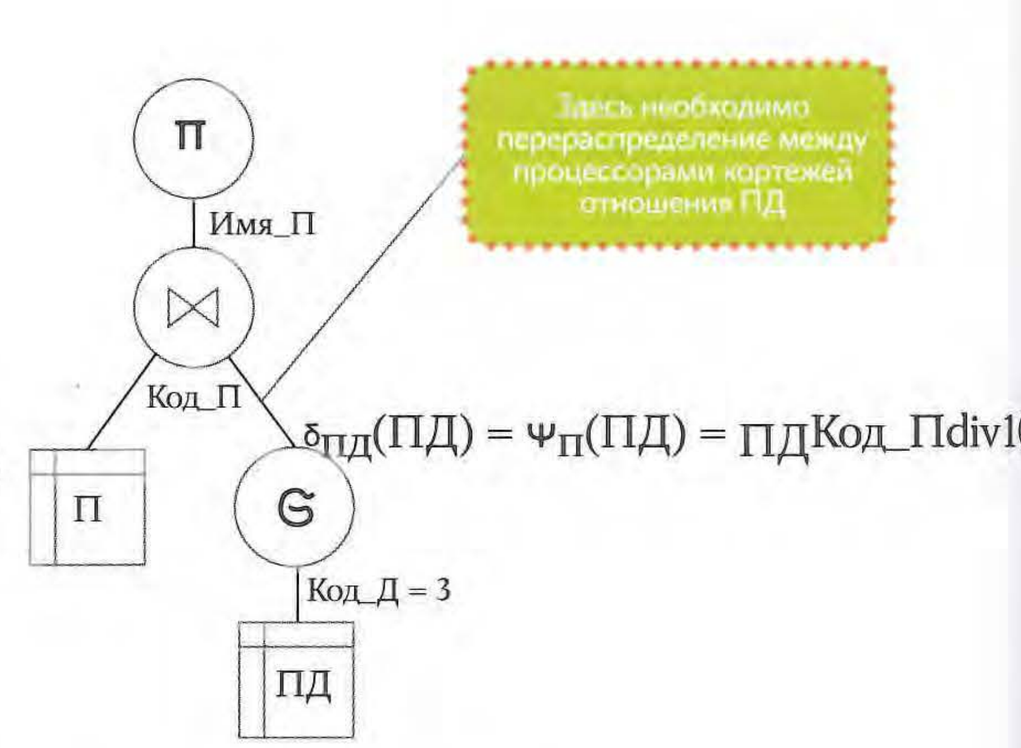
нее представление запроса в СУБД на языке реляционной алгебры. Здесь число «пи» означает операцию проекции, выполняющую отбор указанных столбцов таблицы, сигма – операцию ограничения, осуществляющую отбор строк таблицы по указанному условию, ΔΔ – операцию естественного соединения двух таблиц по общему столбцу. Последняя операция является самой затратной: записи результирующей таблицы получаются путем сцепления каждой записи первой таблицы с каждой записью второй таблицы, за исключением записей, имеющих различные значения в общей колонке. Если столбец, по которому фрагментирована таблица, не совпадает с общим столбцом соединяемых таблиц, то требуется перераспределение записей этой таблицы между узлами.

В СУБД «Омега» для решения данной проблемы вводится специальная операция обмена EXCHANGE. Операция EXCHANGE несет большую нагрузку, выполняя автоматическое распараллеливание запроса: она определяет, какая запись поступила к нему из нижележащей операции – своя (которую надо обработать на текущем узле) или чужая (которая должна быть обработана на другом узле); отправляет чужие записи соответствующим узлам; получает свои записи от других узлов; выдает результирующие записи в вышележащую операцию.

Другим препятствием на пути к получению ускорения являются перекосы в распределении данных по узлам. Существенное различие размеров фрагментов таблицы, вовлеченной в запрос, может привести к дисбалансу загрузки параллельных агентов на узлах и в конечном итоге – к деградации производительности.

В СУБД «Омега» предусмотрено использование техники репликации данных. Фрагмент может иметь несколько реплик (зеркальных копий), которые располагаются на других узлах. На каждом узле может находиться не более одной реплики данного фрагмента. При этом на логическом уровне каждый фрагмент разбивается на сегменты одинакового размера, которые являются наименьшей единицей балансировки загрузки. Схема балансировки загрузки выглядит следующим образом: агенты параллельной СУБД разделяются на два класса – лидеры и аутсайдеры. Агент-лидер – это агент, закончивший обработку назначенного ему фрагмента данных и находящийся в состоянии ожидания нового задания. Агент-аутсайдер – это агент, который в данный момент времени не закончил обработку назначенного ему фрагмента данных. Агент-лидер получает часть работы агента-аутсайдера в виде соответствующей реплики фрагмента данных, еще не обработанного агентом-аутсайдером. Каждому агенту в процессе балансировки загрузки присваивается рейтинг, задаваемый вещественным числом. В качестве аутсайдера всегда выбирается агент, имеющий максимальный положительный рейтинг.

```
/* Имена поставщиков, поставляющих деталь с кодом 3 */
SELECT Имя_П
FROM П, ПД
WHERE П.Код_П = ПД.Код_П
AND ПД.Код_Д = 3
```



П – фрагментированно по Код_П:Ψп(П) = пКод_Пdiv10
ПД – фрагментированно по Код_Д:Ψпд(ПД) = пдКод_Ддiv10

ТИП УЗЛА	ОСНОВНЫЕ ХАРАКТЕРИСТИКИ	ЦЕНА ЗА УЗЕЛ (ТЫС. РУБ.)	КОЛИЧЕСТВО УЗЛОВ (НА 10 МЛН РУБ.)
1	1-ядерный процессор Intel Xeon DP EM64T и один диск SAS 73.4 Гб	266	37
2	2-ядерный процессор Intel Xeon E3110 и два диска SAS 73.4 Гб	298	33
3	Два 2-ядерных процессора Intel Xeon E3110 и четыре диска SAS 73.4 Гб	325	30
4	Два 4-ядерных процессора Intel Xeon E5472 и восемь дисков SAS 73.4 Гб	395	25

Одним из ответвлений проекта «Омега» является разработка прототипа параллельной СУБД в оперативной памяти (СУБД-ОП). Идея сделать основным местом хранения данных оперативную память вместо жесткого диска известна и реализована (например, коммерческие СУБД Oracle TimesTen, IBM solidDB и свободные CSQL, MonetDB), однако примеров совмещения этой идеи с идеей параллельной обработки данных пока крайне мало (по-видимому, только свободная СУБД VoltDB). В рамках этого проекта, с одной стороны, в параллельную СУБД-ОП внедряются механизмы, проверенные ранее в дисковой параллельной СУБД. Например, использование дисковых реплик для случаев, когда чтение данных с локального жесткого диска будет быстрее, чем ожидание сообщения от удаленного узла. С другой стороны, в проекте разрабатываются методы и алгоритмы, специфичные для СУБД-ОП: новые методы компактного хранения таблиц в оперативной памяти, параллельная обработка данных на уровне процессорных ядер, оптимизация производительности за счет использования процессорного кэша. Из научного проекта «Омега» вырос имеющий практическую направленность проект PargreSQL. Проект направлен на разработку параллельной СУБД на основе свободно распространяемой последовательной СУБД PostgreSQL.

Идея проста: есть отлично зарекомендовавшая себя последовательная СУБД, доступная в хорошо документированных исходных кодах. Почему бы не изменить ее исходные тексты таким образом, чтобы она могла работать так же, как прототип «Омега»?

В основе архитектуры PostgreSQL лежит модель «клиент-сервер». Взаимодействуют три вида процессов: приложение-клиент (frontend), серверный процесс (backend) и демон (daemon). Демон осуществляет прием соединений, устанавливаемых клиентами, и создает отдельный серверный процесс для обработки запросов каждого отдельного клиента. В СУБД PargreSQL клиент взаимодействует с двумя или более серверами одновременно: клиентское приложение подключается сразу ко всем экземплярам СУБД, отправляя им одинаковый запрос. При параллельной обработке запроса сначала выполняются те же шаги, что и в случае последовательной СУБД: разбор запроса на языке SQL (parse), замена синонимов таблиц в тексте запроса (rewrite), составление плана запроса и его оптимизация (plan/optimize). Затем на основе последовательного плана формируется параллельный план запроса путем вставки операций EXCHANGE. После запуска исполнителя запроса в случае необходимости осуществляется балансировка загрузки серверных процессов. Остается добавить, что использование PargreSQL будет прозрачно для пользовательских приложений, которые до этого использовали PostgreSQL. В пользовательское приложение необходимо подключить заголовочный файл, содержащий объявления макросов, которые подменяют вызовы функций PostgreSQL вызовами функций PargreSQL.

Другим важным направлением проекта «Омега» является моделирование и анализ иерархических многопроцессорных систем баз данных. Современные многопроцессорные системы в большинстве случаев организуются по иерархическому принципу. Например, большая часть вычислительных кластеров сегодня имеет 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. Проект направлен на разработку математической модели мультипроцессоров баз данных. Разрабатываются методы и алгоритмы, позволяющие реализовать эту модель на компьютере в виде эмулятора многопроцессорных иерархических машин баз данных. При помощи эмулятора можно выполнить вычислительные эксперименты по поиску оптимальных аппаратных архитектур параллельных систем баз данных.