РАСПРЕДЕЛЕННЫЕ И ПАРАЛЛЕЛЬНЫЕ СИСТЕМЫ БАЗ ДАННЫХ

Владислав КУШНИР

Технический Университет Молдовы, Департамент Программной Инженерии и Автоматики

Аннотация: *В статье представлен обзор технологий распределенных и параллельных СУБД, выделены их отличительные черты, отмечены схожие признаки. Цель статьи – помочь в осмыслении уникальной роли систем каждого из этих двух типов и их взаимодополняемости в решении задач управления данными.*

Ключевые слова: *архитектура, клиент-сервер, без разделяемых ресурсов, обработка, запрос, оптимизация, декомпозиция, локализация данных, пространство поиска, глобальная оптимизация.*

Введение

Становление систем управления базами данных (СУБД) совпало по времени со значительными успехами в развитии технологий распределенных вычислений и параллельной обработки. В результате возникли *распределенные системы управления базами данных и параллельные системы управления базами данных*. Именно эти системы становятся доминирующими инструментами для создания приложений интенсивной обработки данных. Благодаря интеграции рабочих станций в распределенную среду становится возможным более эффективное распределение функций в ней, когда прикладные программы выполняются на рабочих станциях, называемых *серверами приложений,* а базы данных обслуживаются выделенными компьютерами, называемыми *серверами баз данных*. Это служит источником развития таких распределенных архитектур, где в роли узлов выступают не просто компьютеры общего назначения, а специализированные серверы.

1. Архитектурные проблемы

Существует множество альтернатив распределенной обработки. Наиболее популярна в настоящее время *архитектура клиент-сервер*, когда множество машин-клиентов осуществляют доступ к одному серверу баз данных. В таких системах, которые можно определить как системы типа *много-клиентов/один-сервер*, проблемы управления базой данных решаются относительно просто, поскольку вся она хранится на одном сервере. Задачи, с которыми приходится здесь сталкиваться, – это управление буферами клиентов, кэширование данных и, возможно, блокировки. Управление данными реализуется *централизованно* на одном сервере. Более распределенной и более гибкой является архитектура типа *много-клиентов/много-серверов*, когда база данных размещена на нескольких серверах, которым, для того чтобы вычислить результат пользовательского запроса или выполнить транзакцию, необходимо взаимодействовать друг с другом. Каждая клиентская машина имеет свой "домашний" сервер; ему она направляет пользовательские запросы. Взаимодействие серверов друг с другом прозрачно для пользователей. В большинстве существующих СУБД реализован один из этих двух типов архитектуры клиент-сервер. В истинно распределенной СУБД клиентские и серверные машины не различаются. В идеале каждый узел может выступать и как клиент, и как сервер. Такие архитектуры, тип которых определяют как *равный-к-равному*, требуют сложных протоколов управления данными, распределенными по нескольким узлам. Предложение продуктов такого вида задерживается из-за сложности необходимого для реализации их программного обеспечения. Архитектуры параллельных систем варьируются между двумя крайними точками, называемыми *архитектура без разделяемых ресурсов* и *архитектура с разделяемой памятью***.** Промежуточную позицию занимает *архитектура с разделяемыми дисками.*

2. Обработка и оптимизация запросов

Обработка запроса – это процесс трансляции декларативного определения запроса в операции манипулирования данными низкого уровня. *Оптимизация запроса* – это процедура выбора "наилучшей" стратегии выполнения запроса из множества альтернатив.

Для централизованной СУБД весь процесс состоит обычно из двух шагов: *декомпозиции запроса* и *оптимизации запроса*. Декомпозиция запроса – это трансляция его с языка SQL в выражение реляционной алгебры. В ходе декомпозиции запрос подвергается семантическому анализу; при этом некорректные запросы отвергаются, а корректные упрощаются. Для заданного SQL-запроса существует более чем одно алгебраическое представление, причем некоторые из них могут быть "лучше" других. "Качество" алгебраического выражения определяется исходя из объема

затрат, необходимых для его вычисления. В распределенной СУБД между шагами декомпозиции и оптимизации запроса включаются еще два действия: *локализация данных* и *глобальная оптимизация запроса*. Исходной информацией для локализации данных служит исходное алгебраическое выражение, полученное на шаге декомпозиции запроса. Основная роль локализации данных заключается в том, чтобы локализовать участвующие в запросе данные, используя информацию об их распределении. На этом шаге выявляются фрагменты, реально участвующие в запросе, и запрос преобразуется к форме, где операции применяются уже не к глобальным отношениям, а к фрагментам. Распределенные отношения реконструируются путем применения инверсии правил фрагментации. Это называется *программой локализации*. Программа локализации для горизонтально (вертикально) фрагментированного отношения представляет собой объединение (union) (соединение (join)) соответствующих фрагментов. Таким образом, на шаге локализации данных каждое глобальное отношение запрос заменяется его программой локализации, а затем результирующий фрагментный запрос упрощается и реструктурируется с целью получения другого "хорошего" запроса. Цель глобальной оптимизации – найти стратегию выполнения запроса, близкую к оптимальной. Стратегию выполнения распределенного запроса можно выразить в терминах *операций реляционной алгебры* и *коммуникационных примитивов*, используемых для пересылки данных между узлами. На предыдущих шагах запрос уже был в определенной мере оптимизирован, в частности, за счет удаления избыточных выражений. Однако проведенная оптимизация не зависела от характеристик фрагментов, например, их мощности. Оптимизация запроса заключается в нахождении "наилучшего" плана из множества возможных планов, исследуемых оптимизатором.

Оптимизатор запросов обычно представляется в виде трех компонентов: пространство поиска, модель стоимости и стратегия поиска. *Пространство поиска* – это множество альтернативных планов выполнения исходного запроса. Эти планы эквивалентны в том смысле, что они дают один и тот же результат, но различаются порядком и способами выполнения отдельных операций. *Модель стоимости* – это способ оценить стоимость данного плана выполнения запроса. Для достижения точности модель стоимости должна основываться на точных знаниях о среде параллельных вычислений. *Стратегия поиска* – это способ обхода пространства поиска и выбора наилучшего плана.

Выводы

За последние несколько лет распределенные и параллельные СУБД стали реальностью. Они предоставляют функциональность централизованных СУБД, но в такой среде, где данные распределены между компьютерами, связанными сетью, или между узлами многопроцессорной системы. Распределенные СУБД допускают естественный рост и расширение баз данных путем простого добавления в сеть дополнительных машин. Подобные системы обладают более привлекательными характеристиками "цена/производительность", благодаря современным прогрессивным сетевым технологиям. Параллельные СУБД – это, пожалуй, единственный реалистичный подход для удовлетворения потребностей многих важных прикладных областей, которым необходима исключительно высокая пропускная способность баз данных. Поэтому при проектировании параллельных и распределенных СУБД следует предусмотреть в них соответствующие протоколы и стратегии обработки, направленные на достижение высокой производительности.

Библиография

1. Распределенные и параллельные системы управления базами данных. – [Электронный ресурс]. – Режим доступа: http://unesco.kemsu.ru/study_work/method/bd/umk/book/gl_1.html
2. Параллельные базы данных. – [Электронный ресурс]. – Режим доступа: https://knowledge.allbest.ru/programming/2c0a65635b3ac68a4d53b89421306c36_0.html
3. М.Тамер Оззу, Патрик Валдуриз. Распределенные и параллельные системы баз данных. Журнал системы управления базами данных № 4/1996, издательский дом “Открытые системы”. – [Электронный ресурс]. – Режим доступа: - http://citforum.ru/database/classics/distr_and_paral_sdb/