РАЗРАБОТКА АЛГОРИТМА СОЕДИНЕНИЯ ОТНОШЕНИЙ ДЛЯ ПАРАЛЛЕЛЬНОЙ СУБД В ОПЕРАТИВНОЙ ПАМЯТИ

А.А. Гусев

В настоящее время одним из наиболее эффективных средств обработки сверхбольших баз данных являются параллельные СУБД, реализованные на аппаратной платформе кластерных вычислительных систем. Параллельная СУБД использует принцип фрагментного параллелизма, который заключается в следующем: каждое отношение (таблица) базы данных разбивается на горизонтальные фрагменты, распределяемые по процессорным узлам кластера. Запрос к базе данных выполняется в виде нескольких параллельных процессов (агентов), каждый из которых обрабатывает свой фрагмент отношения. Полученные фрагменты сливаются в результирующее отношение [1, 2].

Базы данных в оперативной памяти работают намного быстрее, чем те, которые рассчитаны на использование дисковой памяти, поскольку их внутренние алгоритмы проще и заключаются в выполнении меньшего числа машинных команд. При разработке СУБД в оперативной памяти основной задачей является оптимизации подсистем, требующих чтения и обработки большого количества данных [3].

В настоящее время в Южно-Уральском Государственном Университете ведется разработка параллельной СУБД в оперативной памяти в рамках проекта «Омега» [4]. В данной работе рассматривается разработка алгоритма соединения отношений для параллельной СУБД в оперативной памяти, который позволит получить наиболее эффективную реализацию этой операции в параллельной системе управления базами данных в оперативной памяти.

Существует несколько принципов, руководствуясь которыми можно повысить скорость выполнения программы за счет более эффективного использования кэш-памяти. Способ, который лучшим образом подходит для решения стоящей перед нами задачи, разбиение данных, участвующих в операции соединения, на блоки, размер блока данных подбирается таким образом, чтобы он мог разместиться в кэш-памяти процессора. После этого полученные блоки последовательно обрабатываются. Благодаря этому при выполнении алгоритма данные считываются преимущественно из быстродействующей кэш-памяти [5].

Применив к алгоритму соединения с использованием вложенных циклов этот метод оптимизации, был разработан и реализован алгоритм соединения, который может быть представлен с использованием псевдокода следующим образом:

```
Для каждого блока b из отношения S
      для каждого кортежа r из отношения R
            для каждого кортежа s из блока S
                  если Выполняется_условие_соединения(r,s) то
                        записать результирующий кортеж
```

Здесь R, S – отношения базы данных; r, s – кортежи из этих отношений; b – блок, который используется для временного хранения кортежей.

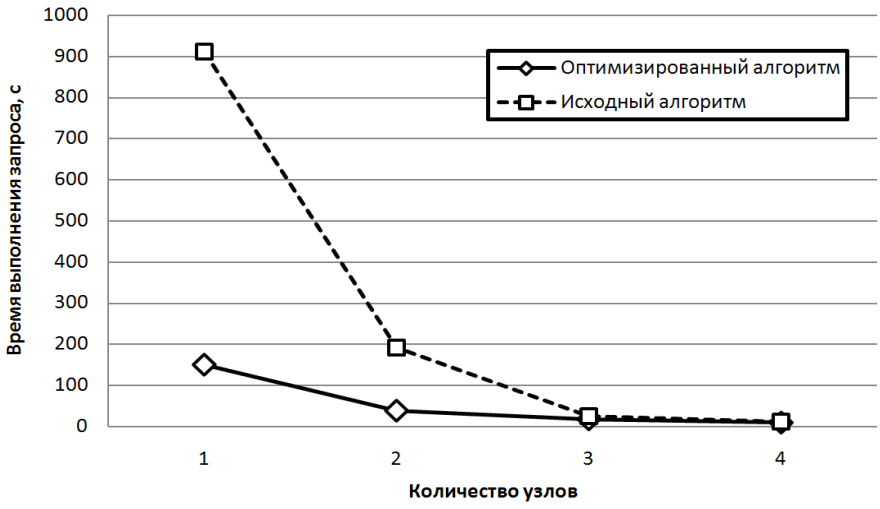

Рис. 1. График зависимости скорости выполнения запроса от количества узлов.

Данный алгоритм был реализован и встроен в параллельную СУБД Омега. Для проверки эффективности реализации была проведена серия экспериментов с использованием вычислительного кластера «СКИФ Урал» Суперкомпьютерного центра Южно-Уральского государственного университета. Каждый вычислительный узел кластера собран на основе процессора Intel Xeon E5472, с объемом кэш-памяти второго уровня 12 MB, и обладает 8 Gb оперативной памяти.

На Рис. 1 приведены результаты эксперимента, в котором за переменную величину было взято количество узлов кластера, на которых размещается параллельная СУБД. Количество кортежей в каждом отношении было выбрано равным 200000. Для сравнения на диаграмме приведены также полученные в аналогичных условиях результаты для неоптимизированного алгоритма.

Также был проведен эксперимент, в котором за переменную величину было выбрано количество кортежей в отношениях. Его целью было выявление зависимости скорости работы рассматриваемых алгоритмов от количества кортежей в отношениях. Из результатов, представленных на Рис. 2, можно сделать вывод, о том, что наибольшее увеличение эффективности происходит при обработке большого количества данных. Это объясняется тем, что достаточно малые отношения при обработке полностью или почти полностью помещаются в кэш-памяти процессора, и как для оптимизированного алгоритма, так и для неоптимизированного в этих случаях данные для обработки будут получаться преимущественно из процессорного кэша.

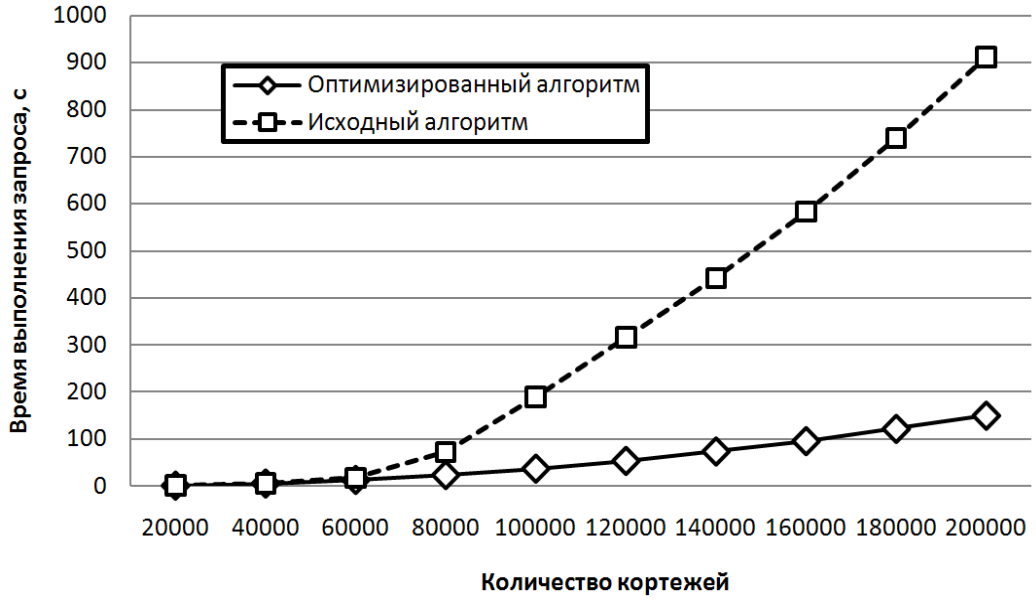

Рис. 2.График зависимости скорости выполнения запроса от количества кортежей.

Пересечение графиков (то есть худший, чем у неоптимизированного алгоритма результат для 40000 и 20000 кортежей) можно объяснить тем, что дополнительные операции, связанные с обработкой данных по блокам, требуют некоторого дополнительного времени на выполнение, но не увеличивают скорости обработки для такого малого количества кортежей.

На основе полученных результатов можно сделать вывод о том, что разработанный алгоритм позволяет получить, в зависимости от условий выполнения и доли вспомогательных операций, ускорить работу алгоритма до 6 раз по сравнению с обычным алгоритмом.

Таким образом, в результате проделанной работы был создан и реализован алгоритм соединения отношений, рассчитанный на использование в параллельной СУБД в оперативной памяти. За счет более эффективного использования процессорной кэш-памяти он обеспечивает выполнение операции за меньшее количество времени. Проведенные эксперименты показывают, что применение этого алгоритма позволяет снизить время выполнения запроса к СУБД на соединение отношений до 6 раз. Дальнейшие исследования по данной теме могут быть направлены на увеличение эффективности других компонентов СУБД, обеспечивающих выполнение основных функций.

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проект 09-07-00241-а).

ЛИТЕРАТУРА:

1. Аксенова Е.В. Разработка параллельной СУБД в оперативной памяти для кластерных систем // Параллельные вычислительные технологии (ПаВТ'2010): Труды международной научной конференции (Уфа, 29 марта -2 апреля 2010 г.). Челябинск: Издательский центр ЮУрГУ. -2010. -С. 652-652.

2. Соколинский Л.Б. Организация параллельного выполнения запросов в многопроцессорной машине баз данных с иерархической архитектурой // Программирование. -2001.№6. -С. 13-29.
3. Peter M. G. Apers, Care1 A. van den Berg, Jan Flokstra, Paul W. P. J. Grefen, Martin L. Kersten, Annita N. Wilschut, "PRISMA/DB: A Parallel, Main Memory Relational DBMS" // IEEE Transactions on Knowledge and Data Engineering -1992 -Vol. 4. -P. 541-554.
4. Параллельная СУБД «Омега» для многопроцессорных иерархий: [Сайт проекта]. URL: http://fireforge.net/projects/omega/
5. Ambuj Shatdal, Chander Kant, Jeffrey F. Naughton, "Cache Conscious Algorithms for Relational Query Processing". In proceedings of 20th International Conference on Very Large Data Bases, September 12--15, 1994, Santiago de Chile, Chile. Pages 510--521.