Исследование эффективности использования фрагментированных колоночных индексов при выполнении операции естественного соединения с использованием многоядерных ускорителей*

Е.В. Иванова

ФГБОУ ВПО «ЮУрГУ» (НИУ)

В работе исследуется подход к обработке сверхбольших баз данных, основанный на использовании колоночных индексов с доменно-интервальной фрагментацией. Данный подход позволяет организовать эффективное параллельное выполнение запросов на современных кластерных вычислительных системах, оснащенных многоядерными ускорителями. В работе дается определение колоночного индекса и доменной фрагментации, рассматривается формальное описание операции естественного соединения над фрагментированными колоночными индексами, приводятся результаты вычислительных экспериментов, выполненные на разработанном прототипе системы.

1. Введение

В настоящее время фактически единственным эффективным решением проблемы хранения и обработки сверхбольших баз данных является использование параллельных систем баз данных, обеспечивающих распределенную обработку запросов на многопроцессорных вычислительных системах с распределенной памятью [1-4].

В последние годы основным способом наращивания производительности процессоров является увеличение количества ядер, а не тактовой частоты, и эта тенденция, вероятно, сохранится [5]. Сегодня GPU (Graphic Processing Units) и Intel MIC (Many Integrated Cores) значительно опережают традиционные процессоры в производительности по арифметическим операциям и пропускной способности памяти, позволяя использовать сотни процессорных ядер для выполнения десятков тысяч потоков. Последние исследования показывают, что многоядерные ускорители могут эффективно использоваться для обработки запросов к базам данных в оперативной памяти [6-8].

Для обработки сверхбольших баз данных в работах [9, 10] были предложены индексные структуры специального вида, которые называются *колоночными индексами*, и *доменно-интервальная фрагментация* для колоночных индексов. Все фрагменты колоночного индекса хранятся в оперативной памяти в сжатом виде. При параллельном выполнении реляционной операции над колоночными индексами упакованные фрагменты индексов входных отношений загружаются на различные процессорные ядра, где происходят их распаковка, выполнение реляционной операции над фрагментами и упаковка частичного результата, представляющего собой наборы ключей. Затем частичные результаты объединяются в результирующий набор ключей, с использованием которого СУБД собирает результирующее отношение. Указанный подход позволяет организовать параллельное выполнение ресурсоемких реляционных операций без обменов данными между процессорными ядрами, а также использовать для этого современные кластерные вычислительные системы, оснащенные многоядерными ускорителями.

Для обозначения реляционных операций в статье используется нотация, заимствованная из монографии [11]. Символом ∘ обозначается конкатенация кортежей.

* Работа выполнена при финансовой поддержке Минобрнауки РФ в рамках ФЦП «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2014—2020 годы» (Госконтракт № 14.574.21.0035).

2. Колоночный индекс и доменно-интервальная фрагментация

Под $R\left(A^*, B_1, \ldots, B_u\right)$ будем понимать отношение R с первичным ключом A и атрибутами $B_1, \ldots, B_u$, представляющее собой множество кортежей длины $u+1$ вида $(a, b_1, \ldots, b_u)$, где $a \in \mathbb{Z}_{\geq 0}$ и $\forall j \in \{1, \ldots, u\}\left(b_j \in \mathfrak{D}_{B_j}\right)$. Здесь $\mathfrak{D}_{B_j}$ - домен атрибута B_j. Через $r.B_j$ будем обозначать значение атрибута B_j, через $r.A$ – значение *первичного ключа* в кортеже r: $r = (r.A, r.B_1, \ldots, r.B_u)$. *Первичный ключ* отношения R обладает свойством $\forall r', r'' \in R\left(r' \neq r'' \Leftrightarrow r'.A \neq r''.A\right)$. Под *адресом кортежа* r мы будем понимать значение первичного ключа этого кортежа. Для получения кортежа отношения R по его адресу будем использовать *функцию разыменования* $\&_R$: $\forall r \in R\left(\&_R(r.A) = r\right)$.

Определение 1. Пусть задано отношение $R\left(A^*, B, \ldots\right)$, $T\left(R\right) = n$. Пусть на множестве $\mathfrak{D}_B$ задано отношение линейного порядка. *Колоночным индексом* $I_{R.B}$ атрибута B отношения R называется упорядоченное отношение $I_{R.B}\left(A^*, B\right)$, удовлетворяющее следующим требованиям:

$$T(I_{R.B}) = n \text{ и } \pi_A\left(I_{R.B}\right) = \pi_A\left(R\right); \tag{1}$$

$$\forall x_1, x_2 \in I_{R.B}\left(x_1 \leq x_2 \Leftrightarrow x_1.B \leq x_2.B\right); \tag{2}$$

$$\forall r \in R\left(\forall x \in I_{R.B}\left(r.A = x.A \Rightarrow r.B = x.B\right)\right). \tag{3}$$

Условие (1) означает, что множества значений первичных ключей (адресов) индекса и индексируемого отношения совпадают. Условие (2) означает, что элементы индекса упорядочены в порядке возрастания значений атрибута B. Условие (3) означает, что атрибут A элемента индекса содержит адрес кортежа отношения R, имеющего такое же значение атрибута B, как и у данного элемента колоночного индекса.

С содержательной точки зрения колоночный индекс $I_{R.B}$ представляет собой таблицу из двух колонок с именами и (см. рис. 1). Количество строк в колоночном индексе совпадает с количеством строк в индексируемой таблице. Колонка B индекса $I_{R.B}$ включает в себя все значения колонки B таблицы R (с учетом повторяющихся значений), отсортированных в порядке возрастания. Каждая строка x индекса $I_{R.B}$ содержит в колонке A первичный ключ (адрес) строки r в таблице R, имеющей такое же значение в колонке B, что и строка x.

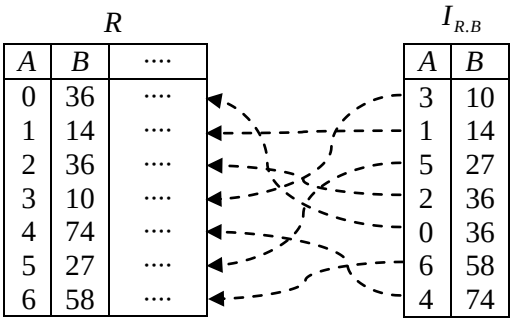

Рис. 1. Колоночный индекс

Определение 2. Пусть на множестве значений домена $\mathfrak{D}_B$ задано отношение линейного порядка. Пусть также задано разбиение множества $\mathfrak{D}_B$ на $k>0$ непересекающихся интервалов:

$$\left.\begin{array}{l} V_0=[v_0;v_1],V_1=(v_1;v_2],\ldots,V_{k-1}=(v_{k-1};v_k]; \\ v_0<v_1<\ldots<v_k; \\ \mathfrak{D}_B=\bigcup_{i=0}^{k-1}V_i. \end{array}\right\}. \quad (4)$$

Функция $\varphi_{\mathfrak{D}_B}:\mathfrak{D}_B\to\{0,\ldots,k-1\}$ называется *интервальной функцией фрагментации* для домена $\mathfrak{D}_B$, если она удовлетворяет следующему условию:

$$\forall i\in\{0,\ldots,k-1\}\left(\forall b\in\mathfrak{D}_B\left(\varphi_{\mathfrak{D}_B}(b)=i\Leftrightarrow b\in V_i\right)\right). \quad (5)$$

Определение 3. Пусть задан колоночный индекс $I_{R.B}$ для отношения $R\left(A^*,B,\ldots\right)$ с атрибутом B над доменом $\mathfrak{D}_B$ и интервальная функция фрагментации $\varphi_{\mathfrak{D}_B}$. Функция

$$\varphi_{I_{R.B}}:I_{R.B}\to\{0,\ldots,k-1\} \quad (6)$$

называется *доменно-интервальной функцией фрагментации* для индекса $I_{R.B}$, если она удовлетворяет следующему условию:

$$\forall x\in I_{R.B}\left(\varphi_{I_{R.B}}(x)=\varphi_{\mathfrak{D}_B}(x.B)\right). \quad (7)$$

Определим *i*-тый фрагмент $\left(i=0,\ldots,k-1\right)$ индекса $I_{R.B}$ следующим образом:

$$I_{R.B}^i=\left\{x\mid x\in I_{R.B};\ \varphi_{I_{R.B}}(x)=i\right\}. \quad (8)$$

Это означает, что в *i*-тый фрагмент попадают кортежи, у которых значение атрибута B принадлежит *i*-тому доменному интервалу. Будем называть фрагментацию, построенную таким образом, *доменно-интервальной*. Количество фрагментов k будем называть *степенью фрагментации*.

Доменно-интервальная фрагментация обладает следующими фундаментальными свойствами, вытекающими непосредственно из ее определения:

$$I_{R.B}=\bigcup_{i=0}^{k-1}I_{R.B}^i\ ; \quad (9)$$

$$\forall i,j\in\{0,\ldots,\mathrm{k}-1\}\left(i\neq j\Rightarrow I_{R.B}^i\cap I_{R.B}^j=\varnothing\right). \quad (10)$$

3. Декомпозиция операции естественного соединения

Пусть заданы два отношения

$$R\left(A^*,B_1,\ldots,B_u,C_1,\ldots,C_v\right)$$

и

$$S\left(A^*,B_1,\ldots,B_u,D_1,\ldots,D_w\right).$$

Пусть имеется два набора колоночных индексов по атрибутам $B_1,\ldots,B_u$:

$$I_{R.B_1},\ldots,I_{R.B_u}\ ;$$

$$I_{S.B_1},\ldots,I_{S.B_u}\ .$$

Пусть для всех этих индексов задана доменно-интервальная фрагментация степени k:

$$I_{R.B_j}=\bigcup_{i=0}^{k-1}I_{R.B_j}^i\ ; \quad (11)$$

$$I_{S.B_j} = \bigcup_{i=0}^{k-1} I^i_{S.B_j} \ . \tag{12}$$

Положим

$$P^i_j = \pi_{I^i_{R.B_j}.A \to A_R,\ I^i_{S.B_j}.A \to A_S} \left(I^i_{R.B_j} \underset{I^i_{R.B_j}.B_j = I^i_{S.B_j}.B_j}{\bowtie} I^i_{S.B_j} \right) \tag{13}$$

для всех $i = 0,\ldots,k-1$ и $j = 1,\ldots,u$. Определим

$$P_j = \bigcup_{i=0}^{k-1} P^i_j \ . \tag{14}$$

Положим

$$P = \bigcap_{j=1}^{u} P_j \ . \tag{15}$$

Определим

$$Q = \left\{ r \circ (s.D_1,\ldots,s.D_w) \mid r \in R \wedge s \in S \wedge (r.A, s.A) \in P \right\}. \tag{16}$$

Тогда $\pi_{*\backslash A}(R) \bowtie \pi_{*\backslash A}(S) = \pi_{*\backslash A}(Q)$. Доказательство этого факта можно найти в [12]. Заметим, что вычисления P^i_j по формуле (13) могут выполняться независимо на *k* различных процессорах без обменов данными. Это обеспечивает ускорение близкое к линейному.

4. Вычислительные эксперименты

Для подтверждения эффективности выполнения операции естественного соединения с использованием фрагментированных колоночных индексов был создан прототип колоночного сопроцессора баз данных, с помощью которого были проведены вычислительные эксперименты. В ходе экспериментов выполнялось естественное соединение двух распределенных колоночных индексов $I_{R.B}$ и $I_{S.B}$ по атрибуту B. Атрибут B колоночного индекса $I_{R.B}$ является первичным ключом, атрибут B колоночного индекса $I_{S.B}$ – внешним ключом. Операция соединения производилась с использованием алгоритма соединения слиянием (Merge Join, MJ). Особенностью алгоритма MJ является то, что соединение осуществляется за одно сканирование каждой из входных таблиц, что существенно уменьшает количество обменов с дисками по сравнению с алгоритмом вложенных циклов.

В экспериментах рассматривалось следующее количество кортежей колоночных индексов $I_{R.B}$ и $I_{S.B}$: $T(I_{R.B})$ = 600 000 и $T(I_{S.B})$ = 60 000 000. Индексы $I_{R.B}$ и $I_{S.B}$ были фрагментированы на основе доменно-интервальной фрагментации на сегменты. Под сегментом будем понимать часть фрагмента, полученную в результате применения к фрагменту доменно-интервальной фрагментации. Каждый сегмент колоночных индексов был сжат алгоритмом Deflate. В проведенных экспериментах атрибут B колоночного индекса $I_{S.B}$ имел одно из следующих распределений значений:

1) равномерное распределение (uniform), при котором вероятность появления значения одинакова для всех значений;
2) распределение по правилу «80-20», при котором 80% кортежей приходятся на 20% значений. В реальных коммерческих приложениях баз данных часто встречается распределение, близкое к правилу «80-20»;
3) распределение по правилу «65-20»;
4) распределение по правилу «45-20».

Эксперименты выполнялись на узле с одним многоядерным ускорителем Intel Xeon Phi кластерной вычислительной системы «Торнадо ЮУрГУ», установленной в Южно-Уральском государственном университете. Многоядерный ускоритель Intel Xeon Phi имеет 61 вычислительное ядро. Операция соединения выполнялась на 1,2 и 4 нитях, запущенных на каждом ядре (рис. 2). Результаты этого эксперимента показывают, что наибольшее ускорение достигается при запуске на каждом процессорном ядре только одной нити.

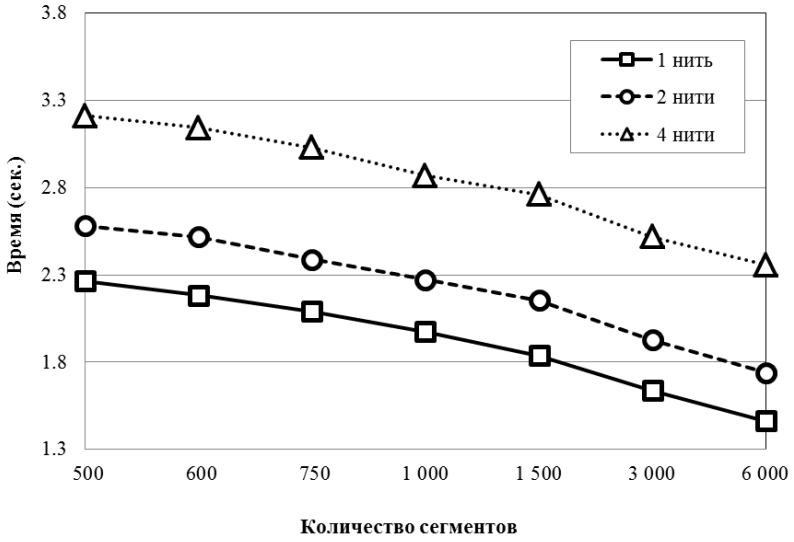

Рис. 2. Зависимость времени выполнения соединения от количества сегментов

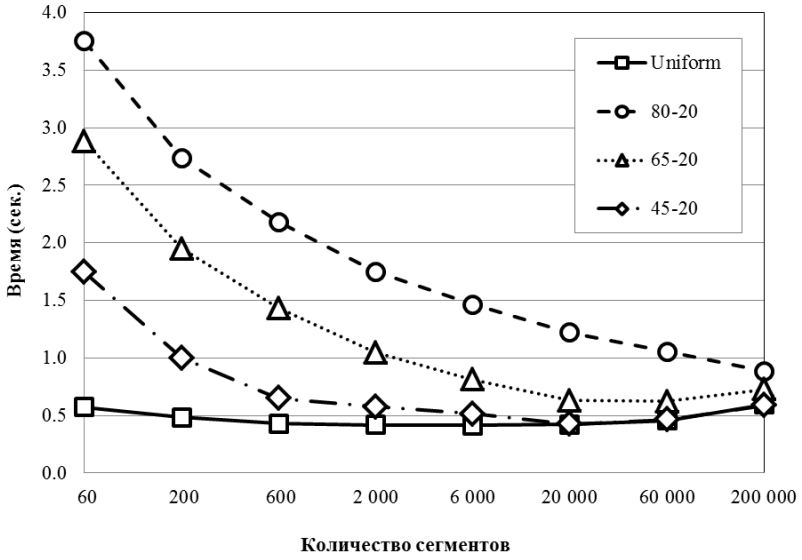

Рис. 3. Влияние количества сегментов на баланс загрузки процессорных ядер Xeon Phi

Вторая группа экспериментов (рис. 3) показала, что деление колоночных индексов на достаточно большое количество фрагментов позволяет эффективно сбалансировать загрузку процессорных ядер даже при наличии большого перекоса в распределении значений в колоночных индексах. Проведенные исследования показывают, что предложенный подход на основе распределенных колоночных индексов позволяет выполнять ресурсоемкую операцию естественного соединения двух таблиц, содержащих 600 000 и 60 000 000 кортежей, менее чем за одну секунду на одном многоядерном ускорителе Intel Xeon Phi. Поскольку предложенная технология позволяет выполнять соединение при отсутствии обменов данных между процессорами, имеются все основания ожидать ускорение близкое к линейному на кластерных вычислительных системах с миллионами процессорных узлов, оснащенных многоядерными ускорителями.

5. Заключение

В статье было представлено формальное описание алгоритма выполнения операции естественного соединения на основе колоночных индексов с доменно-интервальной фрагментаци-

ей. В рамках проведенных экспериментов было исследовано время выполнения операции естественного соединения двух колоночных индексов $I_{R.B}$ и $I_{S.B}$ по атрибуту B. Исследовались зависимость времени выполнения естественного соединения от количества сегментов при запуске 1, 2 и 4 нитей на каждом ядре Intel Xeon Phi и влияние количества сегментов на баланс загрузки процессорных ядер Xeon Phi при равномерном и неравномерном распределении значений атрибута B в индексе $I_{S.B}$. Проведенные исследования показывают, что предложенный подход на основе использования колоночных индексов позволяет эффективно выполнять ресурсоемкую операцию естественного соединения двух таблиц на вычислительных кластерах с многоядерными ускорителями.

Литература

1. Соколинский Л.Б. Параллельные машины баз данных // Природа. 2001. № 8. С. 10–17.
2. Sokolinsky L.B. Design and Evaluation of Database Multiprocessor Architecture with High Data Availability // Proceedings of the 12th International workshop on database and expert systems applications. IEEE Computer Society, 2001. P. 115–120.
3. Pan C.S., Zymbler M.L. Taming Elephants, or How to Embed Parallelism into PostgreSQL // Lecture Notes in Computer Science. 2013. Vol. 8055, Part 1. P. 153–164.
4. Костенецкий П.С., Соколинский Л.Б. Моделирование иерархических многопроцессорных систем баз данных // Программирование. 2013. Т. 39, № 1. С. 3–22.
5. Fang J., Varbanescu A.L., Sips H. Sesame: A User-Transparent Optimizing Framework for Many-Core Processors // Proceedings of the 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid2013), May 13–16, 2013, Delft, Netherlands. IEEE, 2013. P. 70–73.
6. Scherger M. Design of an In-Memory Database Engine Using Intel Xeon Phi Coprocessors // Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'14), July 21–24, 2014, Las Vegas, USA. CSREA Press, 2014. P. 21–27.
7. Breß S., Beier F., Rauhe H., et al. Efficient Co-Processor Utilization in Database Query Processing // Information Systems. 2013. Vol. 38, No. 8. P. 1084–1096.
8. Беседин К.Ю., Костенецкий П.С. Моделирование обработки запросов на гибридных вычислительных системах с многоядерными сопроцессорами и графическими ускорителями // Программные системы: теория и приложения. 2014. Т. 5, № 1-1 (19). С. 91–110.
9. Иванова Е.В., Соколинский Л.Б. Использование распределенных колоночных индексов для выполнения запросов к сверхбольшим базам данных // Параллельные вычислительные технологии (ПАВТ'2014). Труды международной научной конференции. Челябинск: Издательский центр ЮУрГУ, 2014. С. 270–275.
10. Иванова Е.В. Использование распределенных колоночных хеш-индексов для обработки запросов к сверхбольшим базам данных // Научный сервис в сети Интернет: многообразие суперкомпьютерных миров. Труды Международной суперкомпьютерной конференции. М.: Изд-во МГУ, 2014. С. 102–104.
11. Гарсиа-Молина Г., Ульман Дж., Уидом Дж. Системы баз данных. Полный курс. М.: Издательский дом «Вильямс». 2004. 1088 с.
12. Иванова Е.В., Соколинский Л.Б. Декомпозиция операций пересечения и соединения на основе доменно-интервальной фрагментации колоночных индексов // Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. 2015. Т. 4. № 1. С. 44-56.