СТРАТЕГИЯ РАЗМЕЩЕНИЯ ДАННЫХ В МНОГОПРОЦЕССОРНЫХ СИСТЕМАХ С СИММЕТРИЧНОЙ ИЕРАРХИЧЕСКОЙ АРХИТЕКТУРОЙ

А.В. Лепихов, Л.Б. Соколинский

Введение

Современные многопроцессорные системы в большинстве случаев организуются по иерархическому принципу. Основным источником многопроцессорных иерархий, позволяющим объединять несколько различных суперкомпьютеров в единую вычислительную систему, являются Grid-технологии [1]. Grid-система может иметь многоуровневую иерархическую структуру. В такой системе на нижних уровнях иерархии располагаются процессоры отдельных кластерных систем, соединенные высокоскоростной внутренней сетью. На верхних уровнях располагаются вычислительные системы, объединенные корпоративной сетью. Высший уровень иерархии может представлять сеть Интернет.

Проблеме распределения данных и связанной с ней проблеме балансировки загрузки в параллельных системах баз данных без совместного использования ресурсов посвящено большое количество работ (см., например, [2,3,4,5]), однако данная проблематика в контексте иерархических многопроцессорных систем до настоящего времени практически не исследовалась.

В работе предлагается новый подход к размещению данных и балансировке загрузки в многопроцессорных системах реляционных баз данных с иерархической архитектурой. В основе описываемой стратегии лежит модель симметричной многопроцессорной иерархической системы. Стратегия предполагает использование горизонтальной фрагментации отношений и частичного зеркалирования дисков по узлам и уровням многопроцессорной иерархии. При этом каждый фрагмент на логическом уровне разбивается на равномощные сегменты, являющиеся наименьшей единицей репликации данных и балансировки загрузки. Размер реплицируемой части фрагмента на каждом узле задается коэффициентом репликации. Предлагаемые методы размещения данных ориентированы на использование в кластерах и Grid-системах.

1. Симметричные иерархии

Симметричная модель многопроцессорной иерархической системы задает достаточно широкий класс реальных систем и является математическим фундаментом для описания стратегии распределения данных, предлагаемой в настоящей работе.

В основе симметричной модели лежит понятие *DM-дерева*, представляющего собой абстракцию иерархической многопроцессорной системы баз данных [6].

DM-дерево является ориентированным деревом, узлы которого относятся к одному из трех классов:

$\mathfrak{P}_T$ – класс «процессорные модули»;

$\mathfrak{D}_T$ – класс «дисковые модули»;

$\mathfrak{N}_T$ – класс «модули сетевых концентраторов».

Для произвольного *DM*–дерева *T* мы будем обозначать множество всех его узлов как $\mathfrak{M}_T$, множество всех дуг как $\mathfrak{E}_T$.

С каждым узлом $v \in \mathfrak{M}_T$ в *DM*-дереве *T* связывается *коэффициент трудоемкости* $\eta(v)$, являющийся вещественным числом, большим либо равным единицы. Коэффициент трудоемкости определяет время, необходимое узлу для обработки некоторой порции данных. В качестве такой порции данных может фигурировать, например, кортеж.

DM-деревья A и B называются *изоморфными*, если существуют взаимно однозначное отображение f множества $\mathfrak{M}_A$ на множество $\mathfrak{M}_B$ и взаимно однозначное отображение g множества $\mathfrak{E}_A$ на множество $\mathfrak{E}_B$ такие, что:

1) узел v является конечным узлом дуги e в дереве A тогда и только тогда, когда узел $\mathrm{f}(v)$ является конечным узлом дуги $\mathrm{g}(e)$ в дереве B;

2) узел w является начальным узлом дуги e в дереве A тогда и только тогда, когда узел $\mathrm{f}(w)$ является начальным узлом дуги $\mathrm{g}(e)$ в дереве B;

3) $p \in \mathfrak{P}_A \Leftrightarrow \mathrm{f}(p) \in \mathfrak{P}_B$;

4) $d \in \mathfrak{D}_A \Leftrightarrow \mathrm{f}(d) \in \mathfrak{D}_B$;

5) $n \in \mathfrak{N}_A \Leftrightarrow \mathrm{f}(n) \in \mathfrak{N}_B$;

6) $\eta(\mathrm{f}(v)) = \eta(v)$.

Упорядоченную пару отображений $\mathbf{q} = \mathbf{(f, g)}$ будем называть *изоморфизмом DM*-дерева A на *DM*-дерево B.

Мы будем называть *DM*-дерево *T* высоты *H симметричным*, если выполняются следующие условия

1) любые два смежных поддерева уровня $l < H$ являются изоморфными;
2) любое поддерево уровня $H-1$ содержит в точности один диск и один процессор.

2. Фрагментация и сегментация данных

Размещение базы данных на узлах многопроцессорной иерархической системы задается следующим образом. Каждое отношение разбивается на непересекающиеся фрагменты, которые размещаются на различных дисковых модулях. При этом мы используем горизонтальную фрагментацию [5] и предполагаем, что кортежи фрагмента некоторым образом упорядочены, что этот порядок фиксирован для каждого запроса и определяет последовательность считывания кортежей в операции сканирования фрагмента. Мы будем называть этот порядок *естественным*. На практике естественный порядок может определяться физическим порядком следования кортежей или индексом.

Каждый фрагмент на логическом уровне разбивается на последовательность *сегментов* фиксированной длины. Длина сегмента измеряется в кортежах и является атрибутом фрагмента. Разбиение на сегменты выполняется в соответствии с естественным порядком и всегда начинается с первого кортежа. В соответствии с этим последний сегмент фрагмента может оказаться неполным.

Количество сегментов фрагмента *F* обозначается как $\mathrm{S}(F)$ и может быть вычислено по формуле

$$\mathrm{S}(F) = \left\lceil \frac{\mathrm{T}(F)}{\mathrm{L}(F)} \right\rceil. \qquad (1)$$

Здесь $\mathrm{T}(F)$ обозначает количество кортежей во фрагменте *F*, $\mathrm{L}(F)$ – длину сегмента для фрагмента *F*.

3. Алгоритм построения реплики

Пусть фрагмент F_0 располагается на дисковом модуле $d_0 \in \mathfrak{D}_T$ многопроцессорной иерархической системы *T*. Мы полагаем, что на каждом дисковом модуле $d_i \in \mathfrak{D}_T \quad (i > 0)$ располагается *частичная реплика* F_i, включающая в себя некоторое подмножество (возможно пустое) кортежей фрагмента *F*.

Наименьшей единицей репликации данных является сегмент. Длина сегмента реплики всегда совпадает с длиной сегмента реплицируемого фрагмента:

$$\mathrm{L}(F_i) = \mathrm{L}(F_0), \quad \forall d_i \in \mathfrak{D}_T.$$

Размер реплики F_i задается *коэффициентом репликации*

$$\rho(F_i) \in \mathbb{R},\ 0 \le \rho(F_i) \le 1,$$

являющимся атрибутом реплики F_i, и вычисляется по следующей формуле

$$\mathrm{T}(F_i) = \mathrm{T}(F_0) - \left\lceil \left(1 - \rho(F_i)\right) \cdot \mathrm{S}(F_0) \right\rceil \cdot \mathrm{L}(F_0). \qquad (2)$$

Естественный порядок кортежей реплики F_i определяется естественным порядком кортежей фрагмента F_0. При этом *номер первого кортежа* реплики F_i вычисляется по формуле

$$\mathrm{N}(F_i) = \mathrm{T}(F) - \mathrm{T}(F_i) + 1.$$

Для пустой реплики F_i будем иметь $\mathrm{N}(F_i) = \mathrm{T}(F_0) + 1$, что соответствует признаку «конец файла».

4. Метод частичного зеркалирования

Пусть задано симметричное *DM*-дерево *T* высоты $H = h(T) > 1$. Пусть задана *функция репликации* $r(l)$, сопоставляющую каждому уровню $l < H$ дерева *T* коэффициент репликации $\rho_l = r(l)$.

Мы полагаем, что $r(H-1) = 1$. Это мотивируется тем, что уровень иерархии $H-1$ включает в себя поддеревья высоты 1, которым соответствуют SMP-системы. В SMP-системе все диски в равной мере доступны любому процессору, поэтому нет необходимости в физической репликации данных. На логическом уровне балансировка загрузки осуществляется путем сегментирования исходного фрагмента, то есть сам фрагмент играет роль своей реплики.

Определим функцию репликации $r(l)$ для значений $l \le H-2$

Пусть фрагмент F_0 располагается на диске $d_0 \in \mathfrak{D}_T$. Мы будем использовать следующий метод для построения реплики F_i на диске $d_i \in \mathfrak{D}_T \quad (i>0)$, называемый *методом частичного зеркалирования*. Построим последовательность поддеревьев дерева *T*

$$\{M_0, M_1, \ldots, M_{H-2}\}, \tag{3}$$

обладающую следующими свойствами:

$$\begin{cases} l(M_j) = j \\ d_0 \in \mathfrak{D}_{M_j} \end{cases} \tag{4}$$

для всех $0 \le j \le H-2$. Здесь $l(M_j)$ обозначает уровень поддерева M_j. Для любого симметричного дерева *T* существует только одна такая последовательность.

Найдем наибольший индекс $j \ge 1$ такой, что

$$\{d_0, d_i\} \subset \mathfrak{D}_{M_j}.$$

Мы полагаем

$$\rho(F_i) = r(j). \tag{5}$$

Для формирования реплики F_i на диске d_i мы используем алгоритм, описанный в пункте 3 с коэффициентом репликации, определяемым по формуле (5).

5. Выбор функции репликации

При выборе функции репликации $r(l)$ целесообразно учитывать коэффициенты трудоемкости узлов *DM*-дерева. Очевидно, что в симметричном *DM*-дереве все вершины уровня *l* имеют одинаковую трудоемкость $\eta(l)$, которую мы будем называть *трудоемкостью уровня l.*

Назовем симметричное *DM*-дерево *T регулярным*, если для любых двух уровней l и l' дерева *T* справедливо

$$l < l' \quad \Rightarrow \quad \eta(l) \ge \eta(l'),$$

то есть, чем выше уровень в иерархии, тем больше его трудоемкость.

Определим рекурсивно *нормальную* функцию репликации $r(l)$ следующим образом:

1. для $l = H-2$: $r(H-2) = \dfrac{1}{\eta(H-2)(\delta_{H-2}-1)}$
2. для $0 \le l < H-2$: $r(l) = \dfrac{r(l+1)\eta(l+1)(\delta_{l+1}-1)}{\eta(l)(\delta_l - 1)\delta_{l+1}}$

Описанный механизм репликации данных позволяет использовать в многопроцессорных иерархиях простой и эффективный метод балансировки загрузки. Описание данного метода можно найти в работе [7].

Литература

1. *Foster I.T., Grossman R.L.* Blueprint for the future of high-performance networking: Data integration in a bandwidth-rich world // Communications of the ACM. -2003. -Vol. 46, No. 11. -P. 50-57.
2. *Bitton D., Gray J.* Disk Shadowing // Fourteenth International Conference on Very Large Data Bases, August 29 – September 1, 1988, Los Angeles, California, USA, Proceedings. Morgan Kaufmann. -1988. -P. 331-338.
3. *Chen S., Towsley D.F.* Performance of a Mirrored Disk in a Real-Time Transaction System // 1991 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, San Diego, California, USA, May 21-24, 1991, Proceedings. Performance Evaluation Review. -May 1991. -Vol. 19, No. 1. -P. 198-207.
4. *Mehta M., DeWitt D.J.* Data Placement in Shared-Nothing Parallel Database Systems // The VLDB Journal. -January 1997. -Vol. 6, No. 1. -P. 53-72.
5. *Williams M.H., Zhou S.* Data Placement in Parallel Database Systems // Parallel database techniques / IEEE Computer society. -1998. -P. 203-218.
6. *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.
7. *Sokolinsky L.B.* Organization of Parallel Query Processing in Multiprocessor Database Machines with Hierarchical Architecture // Programming and Computer Software. -2001. -Vol. 27, No. 6. -P. 297-308.