004.272.26

Захаров В.Н.[1], Мунерман В.И.[2]

[1]Федеральный исследовательский центр Информатика и управление, г. Москва, Россия
[2]Смоленский государственный университет (СмолГУ), г. Смоленск, Россия

АЛГЕБРАИЧЕСКИЙ ПОДХОД К ФОРМАЛИЗАЦИИ ПАРАЛЛЕЛИЗМА ДАННЫХ*

АННОТАЦИЯ

В статье рассматривается алгебраический подход к формализации параллелизма данных. Дается строгое определение этого понятия на основе теории алгебраических систем. Формулируется критерий параллелизма данных для конкретных типов данных. Приводится пример доказательства параллелизма данных для теоретико-множественной и реляционной моделей данных.

КЛЮЧЕВЫЕ СЛОВА

Параллелизм данных; алгебраические системы; массовая обработка данных.

Victor Zakharov[1], Victor Munerman[2]

[1]Федеральный исследовательский центр Информатика и управление, г. Москва, Россия
[2]Смоленский государственный университет (СмолГУ), г. Смоленск, Россия

ALGEBRAIC APPROACH TO THE FORMALIZATION OF DATA PARALLELISM

SUMMARY

Algebraic approach to the formalization of data parallelism is considered in the article. Strict definition of this concept on the basis of the theory of algebraic systems is given. Criterion data parallelism for specific types of data is formulated. Example of the proof of data parallelism for set-theoretical and relational data models is given.

KEYWORDS

Data parallelism; algebraic systems; mass data processing.

Параллелизм данных (или параллелизм объектов) обычно определяют как обработку по одной и той же или почти одной и той же программе некоторой совокупности данных, поступающих в систему одновременно. Обычно считается, что этот вид параллелизма присущ задачам, которые могут быть формализованы средствами векторной алгебры. Такой взгляд существенно сужает круг задач, сложность которых требует для их решения использование параллельной обработки данных.

Некоторые авторы указывают на сложность проблемы сопоставления параллельно решаемых задач (подсистем) процессорам (ядрам), которые осуществляют эту параллельную обработку. В качестве выхода предлагается решать проблему назначения элементов данных процессорам на уровне программирования на основе технологии мелкозернистого распараллеливания. Для этого в программе выделяются независимые внутренние циклы, которые могут выполняться параллельно [1].

Вместе с тем, в чисто технологических решениях рассматривается более широкое представление о параллелизме данных [2]. В этом случае предполагается, что параллелизм данных относится к сценариям, в которых одна и та же операция выполняется одновременно (то есть параллельно) для элементов в исходной коллекции или массиве. В параллельных операциях с данными исходная коллекция ***секционируются*** таким образом, чтобы несколько потоков могли одновременно работать в разных сегментах. Это представление позволяет перейти от интуитивного взгляда на параллелизм данных к строгому формальному определению.

В основу формального подхода к параллелизму данных положен аппарат алгебраических систем. Пусть обрабатываемые данные представляют собой некоторое множество *A*, на котором определены функции, реализующие унарные, бинарные и групповые (в том смысле, в котором этот

* **Труды I Международной научной конференции «Конвергентные когнитивно-информационные технологии» (Convergent'2016), Москва, 25-26 ноября, 2016**

термин используется в языке SQL) операции $\omega_1, \dots, \omega_p$. Для вычисления значений функций, реализующих групповые операции, необходимо разбиение множества A на непересекающиеся подмножества. Это возможно при условии, что на множестве A определены отношения эквивалентности $\rho_1, \dots, \rho_q$, которые порождают различные разбиения множества A на непересекающиеся классы эквивалентности, тем самым превращая его в различные фактор-множества $A_{\rho_1}, \dots, A_{\rho_q}$.

Таким образом, задание на множестве A операций $\omega_1, \dots, \omega_p$ и предикатов, задающих отношения эквивалентности $\rho_1, \dots, \rho_q$, определяет алгебраическую систему

$$\boldsymbol{A} = < A;\ \omega_1, \dots, \omega_p;\ \rho_1, \dots, \rho_q >.$$

В общем случае, если какая-то функция ω отображает множество A в другое множество, то это будет двухосновная или, в общем случае, многоосновная алгебраическая система.

Важное качество алгебраических систем состоит в возможности задания отображения одной алгебраической системы в другую [3]. Если заданы две алгебраические системы $\boldsymbol{A}_1 = < A_1;\ \omega_{11}, \dots, \omega_{1p};\ \rho_{11}, \dots, \rho_{1q} >$ и $\boldsymbol{A}_2 = < A;\ \omega_{21}, \dots, \omega_{2p};\ \rho_{21}, \dots, \rho_{2q} >$, то отображением φ $\boldsymbol{A}_1$ в $\boldsymbol{A}_2$ называется отображение основного множества A_1 системы $\boldsymbol{A}_1$ в основное множество A_2 системы $\boldsymbol{A}_2$. Если отображение φ взаимно однозначно и сохраняет операции и предикаты, то есть

$$\omega_{1i}(a_1, \dots, a_{1m})\,\varphi = \omega_{2i}(a_1\varphi, \dots, a_{1m}\varphi) \text{ и } \rho_{1i}(a_1, \dots, a_{1m})\,\varphi \Leftrightarrow \rho_{2i}(a_1\varphi, \dots, a_{1m}\varphi),$$

то φ называется изоморфизмом $\boldsymbol{A}_1$ в $\boldsymbol{A}_2$. Если же отображение основного множества A_1 системы $\boldsymbol{A}_1$ в основное множество A_2 системы $\boldsymbol{A}_2$ однозначное и/или $\rho_{1i}(a_1, \dots, a_{1m})\ \varphi \Rightarrow \rho_{2i}(a_1\varphi, \dots, a_{1m}\varphi)$, то φ называется гомоморфизмом.

В интуитивном подходе утверждается, что параллелизм данных присущ, например, массивам. Но фактически, он связан не только с множеством массивов, но и с операциями, которые выполняются над этими массивами. Поэтому можно выделить класс алгебраических систем, которым исторически, с момента их создания присущ параллелизм данных. Одна из таких алгебраических систем – алгебра матриц. Действительно, когда речь идет об операции сложения матриц, предполагается, что все пары соответствующих друг другу элементов складываются одновременно, а при умножении – все строки одной матрицы одновременно умножаются на столбец другой.

Теорема 1. Если алгебраическая система $\boldsymbol{A}_1$ обладает параллелизмом данных, то любая изоморфная или гомоморфная ей алгебраическая система $\boldsymbol{A}_2$ также обладает параллелизмом данных.

Действительно, если алгебраическая система $\boldsymbol{A}_1$ изоморфна или гомоморфна алгебраической системе $\boldsymbol{A}_2$, то из теоремы о гомоморфизме [3, стр. 63] следует, что, во-первых, сохраняется отношение эквивалентности, и, во-вторых, каждому классу эквивалентности фактор-множества основного множества A_1 по отношению эквивалентности ρ_{1i} соответствует единственный класс эквивалентности фактор-множества основного множества A_2 по отношению эквивалентности ρ_{2i}. Поскольку также сохраняются и операции, то из наличия параллелизма данных в алгебраической системе $\boldsymbol{A}_1$ следует наличие параллелизма данных в алгебраической системе $\boldsymbol{A}_2$.

Это утверждение констатирует существование параллельных алгоритмов для операций алгебраической системы $\boldsymbol{A}_2$.

Далее в качестве примера рассматривается доказательство существования параллелизма данных для двух моделей данных (алгебраических систем): теоретико-множественной [4] и реляционной. В качестве алгебраической системы, которой естественно присущ параллелизм данных, используется алгебра логических многомерных матриц (ЛММ). Это объясняется тем, что рассматривается гомоморфное соответствие алгебраических систем структур данных: таблиц, файлов, многомерных матриц, а не типов их элементов. Такое упрощение не ограничивает общность, так как при выполнении операций важно знать образуется ли в файле (отношении) результате запись (строка) с заданным набором значений ключевых полей, а значения остальных полей не представляют интереса. Гомоморфизм алгебраических систем будет доказываться для трех операций, соответствующих свертке, сложению и умножению ЛММ.

Многомерная матрица $A = \left\| a_{i_1 \dots i_p} \right\|$ называется логической многомерной матрицей, если ее элементы принадлежат множеству {0, 1} и над ними определены аддитивная операция дизъюнкции и мультипликативная операция конъюнкции.

1. Пусть дано разбиение совокупности индексов матрицы $A=\|a_{i_1\dots i_p}\|$ на совокупности $l=(l_1,\dots,l_\kappa)$ и $c=(c_1,\dots,c_\mu)$, $\kappa+\mu=p$. Матрица ${}^{\mu}A=\|a_l\|$, элементы которой связаны с элементами матрицы $A=\|a_{lc}\|$ соотношением $b_l=\bigvee\limits_{(c)}a_{lc}$, называется μ-свернутой матрицей и обозначается $B={}^{\mu}A$. Индексы разбиения $l=(l_1,\dots,l_\kappa)$ называются свободными индексами, а индексы разбиения $c=(c_1,\dots,c_\mu)$ – кэлиевыми индексами;

2.Сложение: если $C=A+B$, то $c_{i_1\dots i_p}=a_{i_1\dots i_p}\vee b_{i_1\dots i_p}$;

3.Пусть матрицы $A=\|a_{i_1\dots i_p}\|$ и $B=\|b_{i_1\dots i_q}\|$, p и q-мерные соответственно. Совокупности индексов этих матриц $i_1,\dots,i_p$ и $i_1,\dots,i_q$ разбиваются на четыре группы, содержащие соответственно κ, λ, μ и ν индексов (κ, λ, μ, $\nu\geq 0$). Причем $\kappa+\lambda+\mu=p$, а $\lambda+\mu+\nu=q$. Для полученных групп индексов используются обозначения: $l=(l_1,\dots,l_\kappa)$, $s=(s_1,\dots,s_\lambda)$, $c=(c_1,\dots,c_\mu)$ и $m=(m_1,\dots,m_\nu)$. Тогда матрицы A и Bможно представить в виде в виде $A=\|a_{lsc}\|$ и $B=\|b_{scm}\|$. Индексы групп s и c в матрицах A и B полностью совпадают. Так же как в операции свертки, индексы разбиения c называются кэлиевыми. Индексы разбиения s называются скоттовыми, а индексы разбиения m, также как и индексы разбиения l, – свободными. Матрица $C=\|c_{lsm}\|$, элементы которой вычисляются по формуле $c_{lsm}=\bigvee\limits_{(c)}a_{lsc}\wedge b_{scm}$., называется ($\lambda$, μ)-свернутым произведением матриц A и B.

Пусть $\boldsymbol{X}$ множество однотипных записей. $K=\{K_1,\dots,K_m\},(m<p)$ – конечное множество полей записи R, на множествах значений которых заданы отношения порядка. Множество K называется *множеством ключей*, а его элементы *ключами*.

Кортеж $K^*=\{K_1^*,\dots,K_m^*\}$, элементы которого значения соответствующих ключей, называется экземпляром множества ключей (K_i^* называется экземпляром ключа).

Две однотипные записи называются *эквивалентными*, если они содержат одинаковые экземпляры множества ключей.

Задание множества ключей K разбивает множество однотипных записей $\boldsymbol{X}$ на классы эквивалентности, содержащие записи с одинаковыми значениями ключей – эквивалентные записи. Совокупность всех классов эквивалентности по отношению, заданному множеством ключей, образует фактор-множество множества однотипных записей $\boldsymbol{X}$. Такое фактор-множество обозначается X_K, составляющие его классы эквивалентности – X_{K^*}, или $X_{K^*_{(1)}},X_{K^*_{(2)}}\dots$. Если некоторому экземпляру множества ключей во множестве $\boldsymbol{X}$ не соответствует ни одной записи, считается, что ему соответствует универсальная неопределенная запись Θ. Класс эквивалентности, соответствующий экземпляру множества ключей K^* и состоящий из единственной записи Θ, будет обозначаться Θ_{K^*}

Файлом X_K называется фактор-множество множества однотипных записей X по отношению эквивалентности порожденному множеством K. При таком подходе файл не может быть неупорядоченным. Если каждый класс эквивалентности файла X_K содержит единственную запись, то файл X_K называется строго упорядоченным, если же в каждом классе эквивалентности может быть более одной записи – нестрого упорядоченным. Далее рассматриваются три операции над файлами.

1. Сжатие (*quant*). Пусть даны файлы X_K, нестрого упорядоченный по множеству ключей K, и Y_K, строго упорядоченный по множеству ключей K. Классы эквивалентности этих файлов связаны соотношением $Y_{K^*}=f(X_{K^*})$, где f – функция, реализующая групповую операцию (операцию квантификации). Тогда считается, что файл Y_K получен из файла X_K в результате применения операции сжатия;
2. Слияние строго упорядоченных файлов (*ms*). Пусть даны два файла X_K и Y_K строго упорядоченные по одному и тому же множеству ключей K. В результате слияния этих строго упорядоченных файлов образуется файл Z_K, классы эквивалентности задаются соотношением

$Z_{K^*}=f(X_{K^*},Y_{K^*})$. Функция $f(X_{K^*},Y_{K^*})$, определенная на классах эквивалентности исходных файлов, задает характер операции. В качестве примера задания функция *f* рассматриваются теоретико-множественные операции над файлами:

объединение: $f(X_{K^*},Y_{K^*})=\begin{cases} Y_{K^*},\ если\ X_{K^*}=\Theta_{K^*}, \\ X_{K^*},\ в\ противном случае; \end{cases}$

пересечение: $f(X_{K^*},Y_{K^*})=\begin{cases} Y_{K^*},\ если\ X_{K^*}\neq\Theta_{K^*}\ и\ Y_{K^*}\neq\Theta_{K^*}, \\ \Theta_{K^*},\ в\ противном\ случае; \end{cases}$

разность: $f(X_{K^*},Y_{K^*})=\begin{cases} X_{K^*},\ если\ X_{K^*}\neq Y_{K^*}, \\ \Theta_{K^*},\ в\ противном случае. \end{cases}$

3. Слияние нестрого упорядоченных файлов (*mns*). Пусть X_L и Y_M – файлы, упорядоченные (возможно строго) по множествам ключей *L* и *M*, причем выполняется условие $L\cap M\neq\varnothing$, и пусть *K* – множество ключей, связанное с множествами *L* и *M* соотношениями: $K\subseteq L\cup M$, $L\cap K\neq\varnothing$ и $M\cap K\neq\varnothing$. Это означает, что множество ключей *K* состоит из ключей, входящих в множества *L* и *M*, причем в *K* содержится, по крайней мере, по одному ключу из каждого множества. Тогда, по крайней мере, один файл $X_{K\cap L}$ или $Y_{K\cap M}$ нестрого упорядочен по множеству ключей. Если $M\not\subset L$ и $L\not\subset M$, то файлы $X_{K\cap L}$ и $Y_{K\cap M}$ нестрого упорядочены. Слияние файлов производится по множеству ключей *K*. Пусть *K** – фиксированный экземпляр множества ключей *K*, а $(K\cap L)^*$ и $(K\cap M)^*$ – такие фиксированные экземпляры множеств ключей $(K\cap L)$ и $(K\cap M)$, что значения одноименных ключей в них совпадают. Тогда можно задать вычисление класса эквивалентности файла Z_K по следующему правилу:

$$Z_{K^*}=\begin{cases} \Theta_{K^*}, \quad \text{если}\ X_{(K\cap L)^*}=\Theta_{(K\cap L)^*},\ \text{или}\ Y_{(K\cap M)^*}=\Theta_{(K\cap M)^*}, \\ \quad f(X_{(K\cap L)^*},Y_{(K\cap M)^*}),\ \text{в противном случае.} \end{cases}$$

Функция $f(X_{(K\cap L)^*},Y_{(K\cap M)^*})$ определена на классах эквивалентности $X_{(K\cap L)^*}$ и $Y_{(K\cap M)^*}$, а ее значение – класс эквивалентности Z_{K^*}, состоящий из элементов, каждый из которых вычисляется из пары элементов, принадлежащей декартову произведению $X_{(K\cap L)^*}\times Y_{(K\cap M)^*}$. Если функция $f(X_{(K\cap L)^*},Y_{(K\cap M)^*})$ реализует групповую операцию, то операция слияния нестрого упорядоченных файлов включает в себя и операцию сжатия файла Z_K, в результате которой получается файл $Z_{K'}$ (множество ключей *K'* есть подмножество множества ключей *K*, то есть $K'\subset K$).

Теорема 2. Каждому строго упорядоченному файлу соответствует единственная ЛММ.

Пусть X_K – файл, строго упорядоченный по множеству ключей $K=\{K_1,\dots,K_p\}$. Поскольку множества значений ключей конечные, то можно пронумеровать все значения каждого ключа, и те самым поставить в соответствие каждому ключу K_α индекс $i_\alpha=(1,\dots,n_\alpha)$. Тогда каждому экземпляру множества ключей $K^*=\{K_1^*,\dots,K_p^*\}$ соответствует один и только один набор значений индексов $(i_1^0,\dots,i_p^0)$. Это означает, что между совокупностью всех экземпляров множества ключей *K* файла X_K и совокупностью всех наборов значений индексов $(i_1,\dots,i_p)$ установлено взаимно однозначное соответствие. Пусть экземпляру множества ключей $K^*=\{K_1^*,\dots,K_p^*\}$ соответствует набор значений индексов $(i_1^0,\dots,i_p^0)$. Тогда классу эквивалентности X_{K^*} можно поставить в соответствие элемент ЛММ $X=\left\|x_{i_1\dots i_p}\right\|$, значение которого определяется по формуле:

$$x_{i_1^0 \dots i_p^0} = \begin{cases} 0, \text{если } X_{K^*} = \Theta, \\ 1, \text{если } X_{K^*} \neq \Theta. \end{cases}$$

Так как файл X_K строго упорядочен, каждый его класс эквивалентности содержит либо единственную определенную запись, либо универсальную неопределенную запись. Следовательно, при таком методе построения ЛММ каждому строго упорядоченному файлу соответствует единственная ЛММ. Однако, возможна ситуация, при которой нескольким различным файлам, содержащим данные из различных предметных областей, будет соответствовать одна и та же ЛММ. То есть построенное отображение множества строго упорядоченных файлов на множество ЛММ – однозначное.

Далее рассматривается соответствие операций.

Лемма 1. Пусть X_K – файл, строго упорядоченный по множеству ключей $K = \{K_1, \dots, K_p\}$, $A = \left\| a_{i_1 \dots i_p} \right\|$ – соответствующая ему ЛММ $(\varphi(X_K) = A)$, $M = \{K_1, \dots, K_m\}, (m < p)$ – подмножество множества ключей *M*, по которому файл X_M нестрого упорядочен. Тогда файлу $Y_M = quant(X_M)$ соответствует единственная свертка ЛММ *A*, то есть $\varphi(quant(X_M)) = {}^{\mu}A$.

В соответствии с определением операции сжатия классы эквивалентности строго упорядоченного файла $quant(X_M)$ получаются по формуле $Y_{M^*} = f(X_{M^*})$, где $f(X_{M^*})$ – функция, реализующая групповую операцию на каждом классе эквивалентности, соответствующем экземпляру множества ключей M^*. То есть, совокупность записей файла X_K, в которых ключи $K_1, \dots, K_m$ имеют одни и те же значения, преобразуется в единственную запись файла Y_M. Каждой записи из этой совокупности соответствует элемент ЛММ *A* $a_{i_1^* \dots i_m^* i_{m+1}^0 \dots i_p^0} = 1$. Тогда $a_{i_1^* \dots i_m^*} = \bigvee_{(i_{m+1}, \dots, i_p)} a_{i_1^* \dots i_m^* i_{m+1} \dots i_p} = 1$. Следовательно, $\varphi(quant(X_M)) = {}^{\mu}A$.

Лемма 2. Пусть X_K и Y_K – файлы, строго упорядоченные по множеству ключей $K = \{K_1, \dots, K_p\}$, $A = \left\| a_{i_1 \dots i_p} \right\|$ и $B = \left\| b_{i_1 \dots i_p} \right\|$ – соответствующие им ЛММ $(\varphi(X_K) = A \text{ и } \varphi(Y_K) = B)$. Тогда результату операции слияния строго упорядоченных файлов X_K и Y_K соответствует единственная ЛММ $C = \left\| c_{i_1 \dots i_p} \right\|$ такая, что *C=A+B*.

В зависимости от решаемой задачи, для формирования классов эквивалентности (записей) файла – результата слияния строго упорядоченных файлов X_K и Y_K строятся функции, порождающие либо класс эквивалентности, содержащий либо реальную, либо универсальную неопределенную запись. Аналогично, в качестве аддитивной операции над элементами ЛММ *A* и *B* выбирается одна из шестнадцати логических операций, которая формирует элемент ЛММ результата на основе класса эквивалентности файла-результата и по правилам формирования ЛММ – образа файла при отображении φ. Таким образом, файлу-результату операции слияния строго упорядоченных файлов X_K и Y_K соответствует единственная ЛММ, равная сумме ЛММ *A* и *B* с аддитивной операцией $\circ$ над элементами матриц. То есть, $\varphi(ms(X_K, Y_K)) = A +_{(\circ)} B$. Знак операции $+_{(\circ)}$ читается как операция сложения ЛММ с аддитивной операцией $\circ$ над их элементами.

Лемма 3. Пусть X_L и Y_M – файлы, нестрого упорядоченные по множествам ключей $L = \{L_1, \dots, L_p\}$, и $M = \{M_1, \dots, M_q\}$ причем выполняется условие $L \cap M \neq \varnothing$, и пусть $K = \{K_1, \dots, K_r\}, (r < p + q)$ – множество ключей, связанное с множествами *L* и *M* соотношениями: $K \subseteq L \cup M$, $K \cap L \neq \varnothing$, $K \cap M \neq \varnothing$ (файлы $X_{K \cap L}$ и $Y_{K \cap M}$ нестрого упорядочены по своим множествам ключей). $A = \left\| a_{i_1 \dots i_r} \right\|$ и $B = \left\| b_{i_1 \dots i_r} \right\|$ – соответствующие файлам $X_{K \cap L}$ и $Y_{K \cap M}$ ЛММ $(\varphi(X_{K \cap L}) = A \text{ и } \varphi(Y_{K \cap M}) = B)$. Тогда файлу Z_K – результату операции слияния нестрого упорядоченных по множеству ключей *K* файлов X_K и Y_K, соответствует единственная многомерная матрица $C = \left\| c_{i_1 \dots i_r} \right\| = {}^{\lambda,\mu}(A \times B)$.

В соответствии с определением операции слияния нестрого упорядоченных файлов, множество ключей K состоит из трех подмножеств:

1. L' – ключи, принадлежащие только множеству ключей L;
2. M' – ключи, принадлежащие только множеству ключей M;
3. T – ключи, принадлежащие обоим множествам ключей L и M.

При построении операции ${}^{\lambda,0}(A \times B)$ можно считать, что ключам подмножеств L' и M' соответствуют свободные индексы ЛММ A и B (индексы разбиений l и m), а ключам подмножества T – скоттовы индексы ЛММ A и B (индексы разбиения s). Класс эквивалентности $Z_{K^*} \neq \Theta_{K^*}$ только в том случае, когда соответствующие ему классы эквивалентности $X_{(L' \cup T)^*} \neq \Theta_{(L' \cup T)^*}$ и $Y_{(M' \cup T)^*} \neq \Theta_{(M' \cup T)^*}$. Тогда элементы ЛММ A и B, соответствующие этим классам эквивалентности имеют значение 1. А значит, результат конъюнкции этих элементов также будет иметь значение 1. То есть соответствующий этому элементу класс эквивалентности $Z_{K^*} \neq \Theta_{K^*}$. Если функция $f(X_{(K \cap L)^*}, Y_{(K \cap M)^*})$ реализует групповую операцию, то операция слияния нестрого упорядоченных файлов включает в себя и операцию сжатия файла Z_K, в результате которой получается файл $Z_{K'}$ (множество ключей K' есть подмножество множества ключей K, то есть $K' \subset K$). В этом случае, части ключей подмножества T соответствуют кэлиевы индексы ЛММ A и B (индексы разбиения c), и операции слияния нестрого упорядоченных файлов X_L и Y_M соответствует операция (λ, μ)-свернутого произведения соответствующих им матриц. То есть, $\varphi(mns(X_{K \cap L}, Y_{K \cap M})) = {}^{\lambda,\mu}(A \times B)$.

В таблице 1 приведено соответствие операций над файлами операциям над ЛММ.

Таблица 1. Соответствие алгебраических операций в теоретико-множественной и многомерно- матричной моделях данных

Теоретико-множественная модель (алгебра файлов)	Многомерно-матричная модель (алгебра многомерных матриц)	Тип операции
сжатие	свертка	унарная
слияние строго упорядоченных файлов	сложение	бинарная
слияние нестрого упорядоченных файлов	(λ, μ)-свернутое произведение	бинарная

Таким образом, верно утверждение о том, что множества операций файловой и многомерно матричной моделей данных находятся во взаимно однозначном соответствии.

Теорема 3. Файловая алгебраическая система гомоморфна алгебре ЛММ и также обладает параллелизмом данных.

Эта теорема есть непосредственное следствие теорем 1, 2 и лемм 1-3.

При рассмотрении моделей данных уровня проектирования баз данных, выбор был сделан в пользу SQL реляционной модели, поскольку присущий ей язык манипулирования данными – непроцедурный язык SQL ориентирован на операции с данными, представленными в виде логически взаимосвязанных совокупностей таблиц. То есть предложения этого языка ориентированы в большей степени на конечный результат обработки данных, чем на процедуру этой обработки. Таким образом, их суть в наибольшей степени соответствуют сути алгебраических выражений в алгебрах файлов и многомерных матриц. Кроме того, набор операций SQL реляционной модели достаточно полон для решения задач во многих предметных областях. Остальные современные модели данных, либо имеют не такие полные наборы операций, что приводит к необходимости разработки процедур, реализующих недостающие операции, либо, как объектно-ориентированные модели, наборы операций, дополненные средствами разработки пользовательских АТД.

Выбор SQL реляционной модели обусловлен также тем, что промежуточные файловая и многомерно-матричная модели имеют технологическую направленность. Без этой направленности трудно решить проблемы, связанные с распараллеливанием запросов в целом и составляющих эти запросы операций. Одно из требований к данным в этих двух моделях состоит в возможности их упорядочивания по ключам (индексам многомерных матриц). В отличие от классической реляционной модели SQL реляционная модель допускает возможность упорядочивания данных благодаря наличию в операторе SELECT команды ORDER BY. Хотя при записи выражения запроса упорядоченность входных данных не требуется и не учитывается, в силу непроцедурности языка SQL, в технологических целях, можно считать, что операция сортировки в реляционной модели SQL определена. Несмотря на то, что это предположение ослабляет модель, главное ее свойство – ориентированность на конечный результат сохраняется. А в промежуточных моделях, которые являются абстрактными алгебраическими машинами и в которых операции над структурой отделены от операций над ее элементами, это свойство усиливается. Сделанное допущение позволит упростить доказательство соответствия моделей данных.

SQL реляционная модель рассматривается, в предположении, что все таблицы (отношения), находятся в третьей нормальной форме. Считается, что третья нормальная форма схем отношений достаточна в большинстве случаев, и приведением к третьей нормальной форме процесс проектирования реляционной базы данных обычно заканчивается. Таблица в третьей нормальной форме, как и строго упорядоченный файл, для каждого определенного значения ключа содержит единственную однозначно определенную строку, неключевые атрибуты которой никаким образом не могут быть определены посредством других атрибутов этой таблицы. Следовательно, можно утверждать, что каждой таблице, находящейся в третьей нормальной форме, соответствует (с точностью до порядка следования записей) единственный строго упорядоченный файл.

Под операциями в SQL реляционной модели будут пониматься запросы, схемы которых представлены в первом столбце таблицы 2. Определения операций над таблицами в реляционной модели SQL аналогичны определениям операций в теоретико-множественной модели. Поэтому можно не прибегая к сложным формальным доказательствам считать установленным следующий факт. Если для соответствующих друг другу операций SQL реляционной модели и алгебры файлов выбраны соответствующие друг другу таблицы и файлы операнды, то и результаты этих операций также будут соответствовать друг другу. Например, теоретико-множественным операциям над строго упорядоченными файлами соответствуют следующие запросы:

объединение: SELECT $R.A_1$, …, $R.A_n$ FROM R UNION SELECT $S.A_1$, …, $S.A_n$ FROM S;

пересечение: SELECT $R.A_1$, …, $R.A_n$ FROM R, S WHERE $R.A_1$=$S.A_1$ AND … AND $R.A_n$=$S.A_n$;

разность:

SELECT $R.A_1$, …, $R.A_n$ FROM R WHERE NOT EXIST

(SELECT $S.A_1$, …, $S.A_n$ FROM S WHERE $R.A_1$=$S.A_1$ AND … AND $R.A_n$=$S.A_n$).

Тогда соответствие операций реляционной модели SQL и теоретико-множественной модели может быть установлено так, как это показано в таблице 2.

Таблица 2.3. Соответствие алгебраических операций в реляционной SQL и теоретико-множественной моделях данных

Реляционная модель SQL	Теоретико-множественная модель (алгебра файлов)	Тип операции
SELECT … GROUP BY	сжатие	унарная
Теоретико-множественные операции	слияние строго упорядоченных файлов	бинарная
SELECT … FROM JOIN(…)	слияние нестрого упорядоченных файлов	бинарная

Таким образом, верно утверждение о том, что множества операций SQL реляционной модели, файловой и многомерно-матричной моделей данных находятся во взаимно-однозначном соответствии.

При таком построении моделей, с учетом соответствия операций, можно считать, что между SQL реляционной моделью и теоретико-множественной моделью можно установить даже изоморфное соответствие. Из этого следует и гомоморфизм алгебры многомерных матриц и SQL реляционной модели. Таким образом SQL реляционная модель, рассматриваемая как алгебраическая система также обладает параллелизмом данных и, следовательно, существуют параллельные алгоритмы реализации ее операций. Примеры таких алгоритмов приведены в [5].

Из сказанного следует, что метод, основанный на построении алгебраических систем, формализующих решение задач в предметных областях, позволяет применять строгие методы для определения возможности повышения производительности вычислительных средств за счет использования параллелизма данных.

Литература

1. Boyd C. Data-parallel computing. – ACM QUEUE AC March/April 2008 www.acmqueue.com. – p. 32-39.
2. Data Parallelism (Task Parallel Library). – https://msdn.microsoft.com/ru-ru/library/dd537608(v=vs.110).aspx.
3. Мальцев А.И. Алгебраические системы. – М.: "Наука", 1970 г. – 392 с.
4. Мунерман В.И. Построение архитектур программно-аппаратных комплексов для повышения эффективности массовой обработки данных. – Системы высокой доступности. 2014. Т. 10. № 4. – с. 3-16.
5. Мунерман В.И. Опыт массовой обработки данных в облачных системах (на примере WINDOWS AZURE). – Системы высокой доступности. 2014. Т. 10. № 2. С. 8-13.

References

1. Boyd C. Data-parallel computing. – ACM QUEUE AC March/April 2008 www.acmqueue.com. – p. 32-39.
2. Data Parallelism (Task Parallel Library). – https://msdn.microsoft.com/ru-ru/library/dd537608(v=vs.110).aspx.
3. Мальцев А.И. Алгебраические системы. – М.: "Наука", 1970 г. – 392 с.
4. Мунерман В.И. Построение архитектур программно-аппаратных комплексов для повышения эффективности массовой обработки данных. – Системы высокой доступности. 2014. Т. 10. № 4. – с. 3-16.
5. Мунерман В.И. Опыт массовой обработки данных в облачных системах (на примере WINDOWS AZURE). – Системы высокой доступности. 2014. Т. 10. № 2. С. 8-13.

Поступила: 10.10.2016

Об авторах

Захаров В.Н., ученый секретарь Федерального исследовательского центра "Информатика и управление" РАН, доктор технических наук, vzakharov@ipiran.ru;

Мунерман Виктор Иосифович, доцент кафедры информатики Смоленского государственного университета, кандидат технических наук, vimoon@gmail.com.