УДК 004.657

© 2011 г. **Ю.А. Григорьев,** д-р техн. наук,
В.Л. Плужников
(Московский государственный технический университет им. Н.Э. Баумана)

ОЦЕНКА ВРЕМЕНИ СОЕДИНЕНИЯ ТАБЛИЦ В ПАРАЛЛЕЛЬНОЙ СИСТЕМЕ БАЗ ДАННЫХ

В статье анализируются существующие архитектуры параллельных систем баз данных. Выводятся преобразования Лапласа-Стилтьеса (ПЛС) времени соединения двух таблиц методами NLJ и HJ, а также рассматриваются варианты этого преобразования и соответствующие математические ожидания для архитектур SE, SD, SN параллельных систем баз данных.
Ключевые слова: параллельная система баз данных, преобразование Лапласа-Стилтьеса, математическое ожидание времени выполнения соединения таблиц.

Введение

В настоящее время наблюдается бурное развитие технологий распределенных вычислений и параллельной обработки больших объемов данных [1]. Запросы к реляционной СУБД как нельзя лучше подходят для параллельного выполнения. Их реализация состоит из однородных операций над однородным потоком данных. Каждая операция образует новое отношение (таблицу), так что из операций могут быть составлены высокопараллельные графы потоков данных. Параллельные системы баз данных (ПСБД) становятся доминирующим инструментом для создания приложений, обеспечивающих анализ хранилищ данных.

В настоящее время не существует научно обоснованного метода оценки и выбора архитектуры параллельной системы баз данных [2]. В предыдущей статье [4] рассмотрен математический метод анализа времени выполнения запроса к одной таблице ПСБД. В данной работе демонстрируется применение математического аппарата производящих функций и преобразования Лапласа-Стилтьеса для оценки времени выполнения соединения таблиц в ПСБД.

Получение такой оценки важно, так как операция соединения (join) часто используется в аналитических запросах к хранилищам данных, построенных на основе реляционных баз данных (ROLAP). Прогнозирование индексов производительности параллельных систем баз данных необходимо – соответствующие аппаратно-программные комплексы очень дороги и ошибки в их выборе приводят к большим финансовым потерям.

Варианты реализация параллельных систем баз данных

Наиболее распространенной системой классификации параллельных систем баз данных является система, предложенная Майклом Стоунбрейкером [2]. В соответствии с этой классификацией архитектуры подразделяются на следующие типы: SE (Shared-Everything) – архитектура с разделяемыми памятью и дисками; SD (Shared-Disks) – архитектура с разделяемыми дисками и соединительной сетью; SN (Shared-Nothing) – архитектура с разделяемой сетью.

Копеланд (Copeland) и Келлер (Keller) предложили расширение классификации Стоунбрейкера путем введения двух дополнительных классов архитектур параллельных машин баз данных: CE (Clustered-Everything) – архитектура с SE кластерами, объединенными по принципу SN; CD (Clustered-Disk) – архитектура с SD кластерами, объединенными по принципу SN.

На текущий момент наиболее часто применяемыми типами архитектур при построении высокопроизводительных параллельных систем является архитектура SE. Реже используются архитектуры SD, SN и CE. Реализаций архитектуры CD не известно. Приведем более подробное описание средств реализации архитектурных решений.

SE-архитектура представляет систему баз данных, в которой все диски напрямую доступны всем процессорам и все процессоры разделяют общую оперативную память. Доступ к дискам в системах SE обычно осуществляется через общий буферный пул. При этом следует отметить, что каждый процессор в системе SE имеет собственную кэш-память. Чаще всего эта архитектура реализуется на промышленных SMP-серверах, построенных на процессорах RISC. Примером реализации архитектуры SE может служить комплекс, состоящий из сервера Sun Enterprise M9000, системы хранения данных EMC Symmetrix DMX-4 и инфраструктуры сети хранения данных SAN [5].

SD-архитектура представляет систему баз данных, в которой любой процессор имеет доступ к любому диску, однако каждый процессор имеет свою приватную оперативную память. Процессоры в таких системах соединены посредством некоторой высокоскоростной сети. Системы SD чаще всего реализуется на промышленных серверах с процессорной архитектурой z/Architecture. Хотя по своему строению эти системы близки к архитектуре SMP-сервера, общая архитектура системы может изменяться с помощью программно-аппаратных средств Parallel Sysplex, позволяющих множеству узлов работать с единой высокопроизводительной внешней системой хранения данных. Примером реализации архитектуры SD может служить комплекс, состоящий из двух серверов IBM z10, системы хранения данных IBM DS8800 и инфраструктуры FICON [6].

SN-архитектура характеризуется наличием у каждого процессора собственной оперативной памяти и собственной дисковой системы. Как и в системах SD , процессорные узлы соединены некоторой высокоскоростной сетью. К настоящему моменту создано большое количество исследовательских прототипов и несколько коммерческих систем с архитектурой SN , использующих фрагментный параллелизм на MPP-системах. Примером реализации архитектуры SN может служить комплекс Teradata Database [7, 8], который представляет собой со-

вокупность SMP-систем (или простых однопроцессорных систем), построенных на базе процессоров Intel x86_64 и связанных между собой высокоскоростной внутренней сетью BYNET. Каждый узел имеет свою собственную оперативную память и дисковую систему. Так как сеть BYNET реализует межузловой обмен данными по технологии точка-точка, то в данной конфигурации нет разделяемых ресурсов. В этом комплексе в качестве СУБД может функционировать только СУБД Teradata.

В *архитектуре CD* каждый узел представляет собой SD-систему. Используется дополнительное программное обеспечение, позволяющее осуществлять обмен метаданными между системами, и арбитраж глобальных блокировок в СУБД и преобразующее весь комплекс к архитектуре CD. Примером реализации CD-архитектуры может служить комплекс из 4-серверов IBM SystemX 3950M2 и системы хранения данных Hitachi Universal Storage Platform V [9].

Выполнение запросов в параллельной системе баз данных

Процесс выполнения SQL-запроса в параллельной системе баз данных можно представить в виде следующих шагов [2]:

генерация последовательного плана выполнения запроса;

тиражирование плана выполнения запроса на все узлы системы;

обработка запроса над фрагментированными таблицами (распределение фрагментов таблиц БД по узлам системы выполняется заранее и один раз);

слияние полученных данных.

Рассмотрим процесс параллельной обработки запроса, где выполняется соединение таблиц R и S базы данных (рис. 1). Q = R w v S – это логическая операция соединения (join) двух отношений (таблиц) R и S по некоторому общему атрибуту Y. В данном примере таблица R фрагментирована произвольным образом, а таблица S – по атрибуту соединения Y. На рис. 1 показано, что логический план выполнения соединения двух отношений тиражируется на 'n' процессоров в параллельной системе баз данных (на рисунке показаны 2 процессора). Далее происходит параллельная обработка на каждом процессоре соответствующих фрагментов таблиц R и S. Вследствие того, что таблица R не фрагментирована по атрибуту соединения, при последовательном чтении записей этой таблицы происходит их обработка в операторе exchange, осуществляющем разбор записи и ее межпроцессорный обмен. Таблица S фрагментирована по атрибуту соединения, и записи, читаемые из фрагментов этой таблицы, обрабатываются на каждом процессоре локально.

Оператор exchange является составным оператором и включает в себя четыре оператора: gather, scatter, split и merge. Оператор split – это оператор, который осуществляет разбиение кортежей (записей), поступающих из входного потока, на две группы: свои и чужие. Свои кортежи – те, которые должны быть обработаны на данном процессорном узле. Эти кортежи направляются в выходной буфер оператора split (стрелка вверх). Чужие кортежи, т.е. кортежи, которые должны быть обработаны на процессорных узлах, отличных от данного, помещаются оператором split во входной буфер правого дочернего узла, в качестве ко-

торого фигурирует оператор scatter. Нульарный оператор scatter извлекает кортежи из своего входного буфера и пересылает их на соответствующие процессорные узлы. Нульарный оператор gather выполняет перманентное чтение кортежей из указанного порта со всех процессорных узлов, отличных от данного. Оператор merge, реализующий логическую операцию join, определяется как бинарный оператор, который забирает кортежи из выходных буферов своих дочерних узлов, соединяет их и помещает результат в собственный выходной буфер. Таким образом, с помощью рассмотренных операций оператор exchange реализует полноценный межпроцессорный обмен записями в параллельной системе баз данных при обработке запроса методом фрагментарного параллелизма.

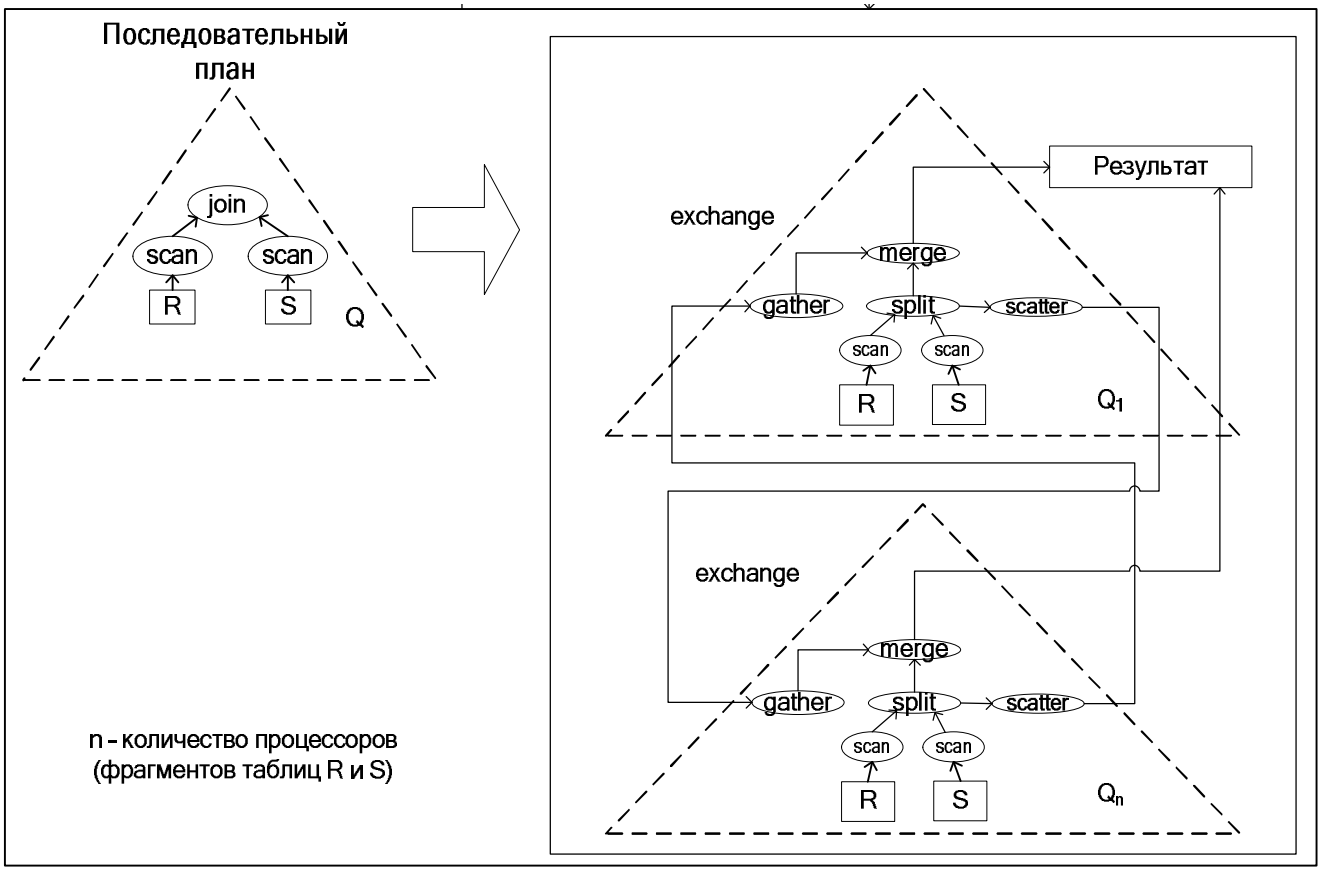

Рис. 1. Обработка запроса Q = R w v S в параллельной системе баз данных.

Преобразование Лапласа-Стилтьеса времени соединения двух таблиц

Ниже получены преобразования Лапласа-Стилтьеса (ПЛС) времени соединения таблиц A и B в параллельной системе баз данных. ПЛС позволяют оценивать не только математические ожидания случайных величин, но и моменты более высокого порядка. Предполагается, что записи таблиц фильтруются и соединяются по неиндексированным атрибутам (т.е. рассматривается наихудший случай). Также предполагается, что таблица B фрагментирована по атрибуту соединения, а таблица A – нет. ПЛС получены для двух способов соединения: NLJ (соединение с помощью вложенных циклов) и HJ (хешированное соединение).

Приведенная ниже формула определяет производящую функцию (ПФ) чис-

ла записей исходной фрагментированной таблицы A i-го узла, передаваемых другим процессорам в соответствии с функцией распределения этого узла (s = 0). Записи, распределенные на i-й процессор (z_i), обрабатываются на этом узле, а остальные записи передаются другим процессорам по команде exchange.

$$V_{Ai}(s, z_1,..., z_n) = G_{Ai}(f_D(s) f_M^2(s) f_P(s)(1 - P_A(1 - q_{Ain}(z_1,..., z_n)))), \quad (1)$$

где $G_{Ai}(z) = z^{\frac{V_A}{n}}$ – производящая функция числа записей исходной фрагментированной таблицы A, обрабатываемых в i-м процессоре (узле), заметим, что эта функция может быть более сложной; V_A/n – число записей таблицы A, которые хранятся в i-м узле; n – число узлов; P_A – вероятность, что запись таблицы A удовлетворяет условию поиска по этой таблице (условию фильтрации); $f_D(s) f_M^2(s) f_P(s)$ – учитывает, что каждая запись таблицы A считывается с диска, сохраняется и читается из ОП, обрабатывается процессором, $q_{Ain}(z_1,..., z_n)$ определяется следующими рекуррентными формулами:

$$\begin{aligned} &q_{Ain}(z_1,..., z_n) = (1 - p_{Ain}) q_{Ain-1}(z_1,..., z_{n-1}) + p_{Ain} z_n, \\ &q_{Ain-1}(z_1,..., z_{n-1}) = (1 - p_{Ain-1}) q_{Ain-2}(z_1,..., z_{n-2}) + p_{Ain-1} z_{n-1}, \ ..., \\ &q_{Ai1}(z_1) = (1 - p_{Ai1}) + p_{Ai1} z_1, \ p_{Ai1} \equiv 1; \end{aligned} \quad (2)$$

p_{Aij} – это вероятность, что запись передается из i-го узла в j-й узел при условии, что она не была передана в узлы n, ..., j + 1.

Формула (1) выводится на основании свойств производящих функций [10]. Действительно:

$$G_{Ai}(1 - \mathrm{P}_A(1 - z)) \quad (3)$$

– это производящая функция числа записей, удовлетворяющих условию поиска по таблице A,

$$q_{Ain}(z_1,..., z_n) \quad (4)$$

– производящая функция числа записей, передаваемых процессорам в соответствии с функцией распределения, для одной произвольной записи таблицы A (для случая, когда таблица фрагментирована не по атрибуту соединения; если таблица фрагментирована по атрибуту соединения, то $q_{Ain}(z_1,..., z_n) = z_i$).

Поясним формулы (2). С вероятностью p_{Ain} запись передается в n-й узел (испытание по схеме Бернулли успешно). С вероятностью (1 – p_{Ain}) запись передается в другие узлы. Производящая функция числа таких записей равна $q_{Ain-1}(z_1,..., z_{n-1})$ и т.д. по рекурсии.

Подставляя (4) в (3), получим (1). Формула (5) определяет производящую функцию числа записей таблицы A, соединяемых в i-м узле.

$$W_{Ai}(z) = \prod_{j=1}^{n} V_{Aj}(0, 1_1,..., 1_{i-1}, z, 1_{i+1},..., 1_n), \quad (5)$$

где $V_{Aj}(s, z_1,..., z_n)$ определяется выражением (1).

При выводе формулы (5) учитывалось, что число записей, обрабатываемых в i-м узле равно сумме случайного числа записей, поступающих в i-й узел из всех n узлов в соответствии с их функциями распределения. Производящая функция

суммы случайного числа записей равна произведению производящих функций.

Производящая функция результирующего числа записей, полученных после соединения таблиц A и B в i-м узле, определяется следующим выражением:

$$W_{ABi}(z) = W_{Ai}(G_{Bi}(1 - P_B h_{AB}(1 - z)), \quad (6)$$

где $G_{Bi}(z) = z^{\frac{V_B}{n}}$ – производящая функция числа записей исходной фрагментированной таблицы B, обрабатываемых в i-м узле; V_B/n – число записей таблицы B, которые хранятся в i-м узле. Предполагается, что таблица B фрагментирована по атрибуту соединения, h_{AB} – вероятность, что две записи из таблиц A и B удовлетворяют условию соединения.

$$h_{AB} = \sum_{j=1}^{|D_A|} h_{Aj} h_{Bk}, \quad (7)$$

где $|D_A|$ – мощность домена атрибута соединения в таблице A; h_{Aj} – вероятность, что атрибут соединения в записи таблицы A принимает значение $d_{Aj} \in D_A$; h_{Bk} – вероятность, что атрибут соединения в записи таблицы B принимает значение $d_{Bk} \in D_B : d_{Bk} = d_{Aj}$; D_B – домен атрибута соединения в таблице B. Докажем (6):

$$W_{Ai}(G_{Bi}(z)) \quad (8)$$

– это производящая функция числа записей в декартовом произведении соединяемых таблиц,

$$1 - P_B h_{AB}(1 - z) \quad (9)$$

– это производящая функция числа записей, удовлетворяющих условию фильтрации (P_B) и условию соединения (h_{AB}), для одной записи таблицы B (действует схема испытания Бернулли). Вероятность h_{AB} получена по формуле полной вероятности (см. (7)). Подставляя (9) в (8), получаем (6).

Получим формулу для ПЛС времени соединения таблиц по методу NLJ в i-м узле. Введем следующую функцию

$$H_{Ai}(s, z) = V_{Ai}(s, f_N^1(s), ..., f_N^{i-1}(s), z, f_N^{i+1}(s), ..., f_N^n(s)) \times \\ \times \prod_{j=1, j \neq i}^{n} V_{Aj}(0, 1_1, ..., 1_{i-1}, z, 1_{i+1}, ..., 1_n), \quad (10)$$

где $V_{Ai}(\cdot)$ определяется выражением (1).

Функция (10) определяет время чтения записей таблицы A в i-м узле (s), время пересылки записей таблицы другим узлам по команде exchange ($\varphi_N(s)$), где они обрабатываются процессорами принимающих узлов, а также число записей таблицы, соединяемых в i-м узле (z). ПЛС времени соединения таблиц по методу NLJ имеет вид

$$\Psi_i^{NLJ}(s) = H_{Ai}(s, G_{Bi}(f_D(s) f_M^2(s) f_P(s)(1 - P_B h_{AB}(1 - f_N(s))))), \quad (11)$$

где $H_{Ai}(\cdot)$ определяется выражением (10); $G_{Bi}(1 - P_B h_{AB}(1 - z))$ – производящая функция числа записей таблицы B, удовлетворяющих условию поиска по таблице B (P_B) и условию соединения h_{AB}, см. (7).

Поясним (11). Следует отметить, что таблица B, сканируемая во внутрен-

нем цикле, фрагментирована по атрибуту соединения. Каждая запись из A сравнивается по атрибуту соединения со всеми записями из B. Каждая запись из B должна быть прочитана с диска, сохранена и прочитана из ОП и обработана в процессоре (произведение $f_D(s)f_M^2(s)f_P(s)$, при этом учитывается наличие общесистемного буфера СУБД, т.е. кэша). В формуле (11) учитывается, что чтение и обработка записей из B повторяется для каждой записи из A (это связано с тем, что в скобочном шаблоне для каждой записи из A выполняется команда reset для правого сына, т.е. для B). С вероятностью $P_B h_{AB}$ результирующая запись передается по команде exchange в узел ($\varphi_N(s)$), где выполняется сборка и формируется окончательный результат.

Получим ПЛС времени соединения таблиц по методу HJ (алгоритм Grace) в i-м узле. Таблица B фрагментирована по атрибуту соединения. ПЛС времени соединения таблиц по методу HJ имеет вид

$$\Psi_i^{HJ}(s) = H_{Ai}(s,(f_D(s)f_M^2(s))^{w_A})\times$$
$$G_{Bi}(f_D(s)f_M^2(s)f_P(s)(1-P_B(1-(f_D(s)f_M^2(s))^{w_B})))\times \qquad (12)$$
$$W_{Ai}(G_{Bi}(1-P_B+P_B(1-\frac{1}{r}+\frac{1}{r}f_P(s))(1-h_{AB}+h_{AB}f_N(s)))),$$

где $H_{Ai}(s, z)$ определяются выражением (10), $G_{Bi}(z)$ – производящая функция числа записей фрагментированной исходной таблицы B, обрабатываемых в i-м узле, $W_{Ai}(z)$ определяется выражением (5).

Поясним формулу (12). Записи фрагментированной таблица A читаются в i-м узле ($H_{Ai}(s, \cdot)$). Записи таблицы B должны быть прочитаны с диска, сохранены и прочитаны из ОП, а также обработаны в процессоре с целью их фильтрации ($G_{Bi}(f_D(s)f_M^2(s)f_P(s)(1-P_B(\cdot)))$). Записи таблиц A и B, поступившие в i-й узел, подвергаются хэшированию ($(f_D(s)f_M^2(s))^{w_A}$ и $(f_D(s)f_M^2(s))^{w_B}$).

Если хэш-таблица создающей таблицы A полностью сохраняется в оперативной памяти, то записи и создающей таблицы (A), и зондирующей таблицы (B) читаются с диска и обрабатываются один раз (это чтение уже учтено, поэтому ($w_A = w_B = 0$). Если нет, то каждая таблица читается с диска (1-е обращение к диску, оно уже учтено), она хэшируется в ОП и хэш-группы сохраняются на диске (2-е обращение к диску). После этого пары хэш-групп таблиц A и B (одна пара имеет одинаковое значение хэш-ключа) читаются с диска (3-е обращение к диску). Поскольку размеры хэш-групп создающей таблицы A могут отличаться, то в рассматриваемом случае $w_A \le 2$ и $w_B \le 2$. Поясним третий сомножитель в формуле (12). Хэш-группы (разделы) таблиц A и B соединяются в оперативной памяти. Каждая запись таблицы A хранится в одном из r разделов (число записей таблицы A определяется ПФ $W_{Ai}(\cdot)$). Произвольная запись из B (их число определяется ПФ $G_{Bi}(1 - P_B + P_B z)$) с вероятностью $1/r$ попадает в тот раздел, где хранится запись из A, и выполняется операция сравнения значений атрибутов соединения ($\varphi_P(s)$). С вероятностью h_{AB} сравнение будет успешным и результат соединения двух записей будет передан в узел сборки ($\varphi_N(s)$).

Так как узлы в параллельной системе баз данных идентичны, то (11) и (12)

определяют ПЛС времени реализации плана соединения, т.е. времени выполнения исходного запроса.

Оценка математического ожидания времени соединения таблиц

В приведенных выше формулах встречаются ПЛС $\varphi_D(s)$, $f_M^2(s)$, $\varphi_P(s)$ и $\varphi_N(s)$. Они имеют следующий смысл [4]: $\varphi_D(s)$ – ПЛС времени чтения записи БД из подсистемы ввода/вывода; $f_M^2(s)$ – ПЛС времени записи и последующего чтения записи БД из ОП; $\varphi_P(s)$ – ПЛС времени обработки записи БД в процессоре; $\varphi_N(s)$ – ПЛС времени передачи записи БД по высокоскоростной шине, связывающей процессоры системы.

В табл. 1 приведены выражения для указанных преобразований для различных архитектурных решений, где n – процессоры (узлов) в параллельной системе баз данных; λ и μ – интенсивности поступления записей от одного процессора и их обработки в соответствующем ресурсе (D – внешней памяти, M – ОП, P – процессоре, N – шине); L – среднее число записей, читаемых из RAID-массива за одно многоблочное чтение при сканировании таблицы СУБД, по существу, величина $(1 - 1/L)$ – вероятность чтения записи из общесистемного буфера СУБД (т.е. из кэша).

Таблица 1

<table>
<tr><th></th><th>SE</th><th>SD</th><th>SN</th></tr>
<tr><td>$f_D(s)$</td><td colspan="2">$1-\frac{1}{L}+\frac{1}{L}\frac{m_D-nl_D}{m_D-nl_D+Ls}$</td><td>$1-\frac{1}{L}+\frac{1}{L}\frac{m_D}{m_D+Ls}$</td></tr>
<tr><td>$f_M(s)$</td><td rowspan="2">$\frac{m_M-nl_M}{m_M-nl_M+s}$
(обмен между процессорами осуществляется через ОП)</td><td colspan="2">$\frac{m_M}{m_M+s}$</td></tr>
<tr><td>$f_N(s)$</td><td colspan="2">$\frac{m_N-nl_N}{m_N-nl_N+s}$</td></tr>
<tr><td>$f_P(s)$</td><td colspan="3">$\frac{m_P}{m_P+s}$</td></tr>
</table>

Наличие произведения '$n\lambda$' в ПЛС означает, что ресурс является разделяемым в соответствующей архитектуре (т.е. к нему возможна очередь). Хотя поведение параллельной системы баз данных при выполнении запроса описывается в виде замкнутой системы массового обслуживания (СМО), но, как видно из табл. 1, здесь в качестве модели разделяемого ресурса используется разомкнутая СМО M/M/1. Это объясняется несколькими причинами.

1. Для замкнутых СМО даже в предположении их экспоненциальности не существует простых аналитических формул, позволяющих оценивать характеристики системы [11]. Сложность расчета замкнутых СМО существенно возрастает с увеличением числа процессоров, т.е. 'n'.

2. Как правило, в параллельной системе баз данных присутствует не более одного разделяемого ресурса, который можно охарактеризовать как "узкое место" (чаще всего это внешняя память). В этом случае можно показать, что все осталь-

ные ресурсы можно считать неразделяемыми. Если предположить, что время обработки в оставшемся разделяемом ресурсе распределено по экспоненциальному закону, то распределение вероятностей числа требований в обслуживающих аппаратах (ОА) замкнутой СМО зависит от средних значений времени обработки в неразделяемых ресурсах, т.е. не зависит от вида функций распределения времени обслуживания в этих ресурсах [11]. Поэтому можно считать, что время обработки требований в неразделяемых ресурсах распределено по экспоненциальному закону. Таким образом, замкнутая модель сводится к классической модели "ремонтника", которая при достаточно большом 'n' близка по характеристикам к разомкнутой СМО M/M/1 [12].

3. В работах [12, 13] показано, что для разомкнутой СМО GI/GI/1 без прерывания обработки справедлив закон сохранения работы. В этом случае среднее время ожидания в очереди не зависит от дисциплины обслуживания этой очереди. То есть для дисциплин LIFO ("последний пришел, первый обслужен") , RS ("случайная выборка из очереди" – Ethernet на шине), с пакетной обработкой (LIFO или RS внутри пакета – "лифтовый поиск" в SCSI-диске), с относительным приоритетом (и др.) можно для расчета среднего времени пребывания использовать формулы для дисциплины FIFO ("первый пришел, первый обслужен").

4. Если использовать СМО M/M/1, то для перехода от разделяемого ресурса к неразделяемому достаточно в выражении для ПЛС убрать произведение '$n\lambda$' (см. табл. 1).

5. Реально для модели ресурса можно оценить только параметры λ и μ. Поэтому использовать для расчетов другую разомкнутую СМО, отличную от M/M/1, проблематично.

Дифференцируя выражения (11) и (12) как сложные функции по s в нуле, можно получить моменты случайного времени (ξ) обработки запроса в параллельной системе баз данных (ПСБД) для различных способов соединения таблиц (NLJ и HJ):

$$M_x = -\Psi'(0),\ M_{x^2} = \Psi''(0),\ s_x^2 = M_{x^2} - M_x^2. \qquad (13)$$

После дифференцирования с учетом выражений, приведенных в табл. 1, получим следующие формулы для расчета математических ожиданий времени выполнения соединения таблиц для различных архитектур ПСБД:

$$M_{SE} = \frac{Q_D}{m_D - n l_D} + \frac{Q_M + Q_N}{m_M - n l_M} + \frac{Q_P}{m_P}, \qquad (14)$$

$$M_{SD} = \frac{Q_D}{m_D - n l_D} + \frac{Q_M}{m_M} + \frac{Q_N}{m_N - n l_N} + \frac{Q_P}{m_P}, \qquad (15)$$

$$M_{SN} = \frac{Q_D}{m_D} + \frac{Q_M}{m_M} + \frac{Q_N}{m_N - n l_N} + \frac{Q_P}{m_P}. \qquad (16)$$

Здесь и далее индекс ξ будем опускать. Выражения для Q_D, Q_M, Q_N, Q_P для методов соединения NLJ и HJ приведены в табл. 2. При получении формул для HJ предполагалось, что хэш-таблица создающей таблицы A полностью сохраняется в оперативной памяти, что упрощает выражение (в этом случае $w_A = w_B = 0$).

Таблица 2

	NLJ	HJ
Q_D	$\frac{V_A}{n}\left(1+\frac{P_A V_B}{n}\right)$	$\frac{1}{n}(V_A+V_B)$
Q_P		$\frac{1}{n}(V_A+V_B+\frac{V_A P_A V_B P_B}{rn})$
Q_M	$\frac{2V_A}{n}\left(1+\frac{P_A V_B}{n}\right)$	$\frac{2}{n}(V_A+V_B)$
Q_N	$\frac{V_A P_A}{n^2}((n-1)+V_B P_B h_{AB})$	

Параметры, используемые в формулах табл.2, определены ранее. Прямой расчет по формулам (14) и (15) затруднен, так как достаточно сложно одновременно оценить параметры λ_D и λ_M разделяемых ресурсов (диска и ОП). Но в системе из двух таких ресурсов, как правило, только один из них является "узким местом" (обычно это диск). То есть в среднем в нем накапливается требований намного больше, чем в другом разделяемом ресурсе. Наличие "узкого места" легко определить (табл. 3). В этом случае другой разделяемый ресурс можно считать неразделяемым. Тогда замкнутая модель ПСБД сводится к модели "ремонтника", которая при достаточно больших 'n' близка по характеристикам к СМО M/M/1 с интенсивностью входного потока $n\lambda$. Параметр λ легко рассчитывается (см. табл. 3). Расчетные формулы для математического ожидания представлены в последней колонке табл. 3.

Таблица 3

	Условие наличия "узкого места"	"Узкое место"	Определяемый параметр λ	Формула для оценки λ	Формула для оценки математического ожидания (МО) времени выполнения соединения двух таблиц
SE	$\frac{Q_D}{m_D} >> \frac{Q_M+Q_N}{m_M}$	Диск	λ_D	$\frac{Q_D}{\frac{Q_M+Q_N}{m_M}+\frac{Q_P}{m_P}}$	$\frac{Q_D}{m_D-nl_D}+\frac{Q_M+Q_N}{m_M}+\frac{Q_P}{m_P}$
	$\frac{Q_D}{m_D} << \frac{Q_M+Q_N}{m_M}$	Память	λ_M	$\frac{Q_M+Q_N}{\frac{Q_D}{m_D}+\frac{Q_P}{m_P}}$	$\frac{Q_M+Q_N}{m_M-nl_M}+\frac{Q_D}{m_D}+\frac{Q_P}{m_P}$
SD	$\frac{Q_D}{m_D} >> \frac{Q_N}{m_N}$	Диск	λ_D	$\frac{Q_D}{\frac{Q_M}{m_M}+\frac{Q_N}{m_N}+\frac{Q_P}{m_P}}$	$\frac{Q_D}{m_D-nl_D}+\frac{Q_M}{m_M}+\frac{Q_N}{m_N}+\frac{Q_P}{m_P}$
	$\frac{Q_D}{m_D} << \frac{Q_N}{m_N}$	Сеть	λ_N	$\frac{Q_N}{\frac{Q_D}{m_D}+\frac{Q_M}{m_M}+\frac{Q_P}{m_P}}$	$\frac{Q_D}{m_D}+\frac{Q_M}{m_M}+\frac{Q_N}{m_N-nl_N}+\frac{Q_P}{m_P}$
SN	нет	Сеть	λ_N		

Эти формулы отличаются от выражений (14) и (15) тем, что для некритического разделяемого ресурса отсутствует произведение nλ, т.е. этот ресурс считается неразделяемым.

Расчеты по формулам, приведенным в табл.3, следует выполнять в следующей последовательности:

выбрать требуемую архитектуру – колонка 1;

выявить "узкое место" системы (оно, как правило, присутствует, так как загрузки ресурсов невозможно сбалансировать) – колонки 2, 3;

рассчитать параметр λ для "узкого места" – колонки 4, 5;

если для "узкого места" выполняется неравенство $(n\lambda/\mu) \geq 1$, то уменьшить число процессоров n и повторить пункт 4; если это неравенство выполняется и для $n = 1$, то считать данное архитектурное решение неудачным, поскольку в этом случае все требования будут ждать обслуживания в разделяемом ресурсе ("узкое место" очень медленное) и процессоры будут простаивать; иначе перейти к пункту 5;

рассчитать математическое ожидание времени выполнения запроса в ПСБД – колонка 6.

Пример расчета среднего времени соединения таблиц

Расчет математического ожидания (среднего) времени соединения таблиц в ПСБД был выполнен при следующих значениях характеристик ресурсов.

1. Процессор Intel Xeon 5160. СУБД Oracle 9i. В [3, с. 347] приведены результаты автотрассировки простого запроса соединения: число обработанных записей в двух таблицах равно 640, число процессорных циклов $7 \cdot 10^6$. В [14] для выбранного процессора приведено измеренное значение числа процессорных циклов, выполняемых Oracle в секунду (CPUSPEED) – $1.5 \cdot 10^9$. Для интенсивности обработки записей в процессоре имеем $\mu_P = 640/(7 \cdot 10^6/1.5 \cdot 10^9) = 1{,}3 \cdot 10^5$.

2. Внешняя память RAID10 с $K_Д = 10$ дисками 3.5" Seagate Cheetah 15K.6 ST3146356FC; размер блока чередования (stripe size) $Q_{Бч} = 256$ Кб; среднее время поиска и чтения блока чередования с диска

$t_{БЧ} = t_{подвода} + t_{вращения}/2 + Q_{БЧ}/v_{чтения} = 4 + 4/2 + 256/200 = 7$ мс.

Размер блока СУБД $Q_{БС} = 16$Кб; средняя длина записи таблиц A и B $l_З = 0.256$Кб, размер многоблочного чтения СУБД (переменная СУБД db_file_multiblock_read_count) $q_{МБ} = 80$. Для интенсивности чтения записей БД из массива RAID имеем $\mu_D = q_{МБ} \times Q_{БС}/l_З/(\lceil q_{МБ} \times Q_{БС}/Q_{БЧ}/(K_Д/2) \rceil \cdot t_{БЧ}) = 0.7 \cdot 10^6$, где число 2 в формуле учитывает тот факт, что при чтении полосы RAID-массива (т.е. при одновременном чтении блоков чередования с разных дисков) используется половина дисков массива, так как в RAID10 каждый диск из этой половины имеет зеркальную копию.

3. Оперативная память – DDR3-1600 PC3- 12800. Расчеты показывают, что интенсивность чтения записей базы данных из ОП равна $\mu_M = 10.4 \cdot 10^6$.

4. Высокоскоростная системная шина: для SD – Parallel Sysplex 12x QDR, для SN – ByNet. В обоих случаях узлы соединены по схеме "точка-точка" (т.е.

шина является неразделяемым ресурсом) и интенсивность передачи записей по этим шинам равна $\mu_N = 50.3 \cdot 10^6$.

Значения остальных параметров следующие: число записей в таблицах A и B – $V_A = V_B = 10^6$, вероятность фильтрации записей таблиц A и B – $P_A = P_B = 10^{-3}$, вероятность успешного соединения двух записей – $\eta_{AB} = 10^{-4}$, количество хеш-групп в методе соединения HJ – $r = 10$.

При расчетах использовались формулы из табл. 3 и 2. Для архитектур SE и SD "узким местом" является внешняя память (диск). При вычислениях величину '$n\lambda_D$' необходимо разделить на 2, так как при обработке запросов на чтение от разных процессоров может быть использована не только основная, но и зеркальная половина дисков. Для архитектуры SN произведение '$n\lambda_N$' необходимо исключить из формулы расчета математического ожидания (т.е. обнулить '$n\lambda_N$' в колонке 6 табл. 3 для строки SN), так как в рассматриваемом примере шина является неразделяемым ресурсом. Графики зависимости математического ожидания времени выполнения соединения двух таблиц от числа процессоров 'n' в параллельной системе баз данных приведены на рис. 2 и 3.

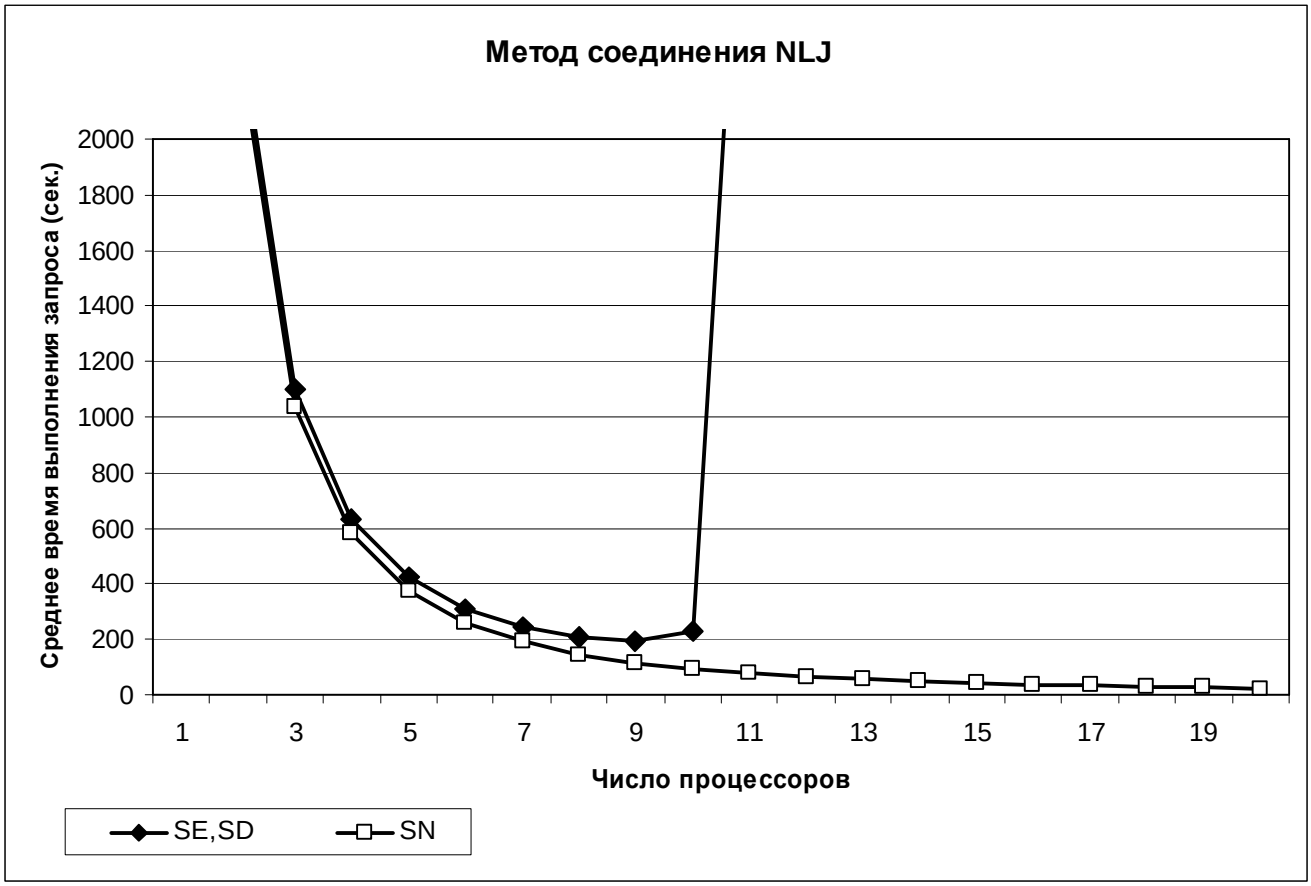

Рис. 2. Зависимость математического ожидания времени выполнения соединения таблиц в архитектурах SE, SD и SN методом NLJ от числа процессоров.

Анализ графиков позволяет сделать несколько выводов:

1. Графики для архитектур SE SD практически совпали. Это объясняется высокими значениями интенсивностей μ_M и μ_N.

2. При $n \geq 3$ среднее время для метода соединения HJ на два порядка мень-

ше среднего времени для NLJ (этого и следовало ожидать для неиндексированной по атрибуту соединения вложенной таблицы B). Более того, $M_{NLJ}(1)\approx 9400$ с., $M_{HJ}(1) \approx 19$ с.

3. При $n \le 7$ архитектуры SE и SD ненамного хуже SN. Следует также отметить, что при n=7 загрузка диска для SE и SD равна 0,63.

4. Для SE и SD при $n = 11$ перегружается дисковая подсистема, и дальнейший рост числа процессоров не имеет смысла. Для архитектуры SN время продолжает уменьшаться с ростом n (здесь нет разделяемых ресурсов). Однако следует иметь в виду, что для снижения времени выполнения соединения методом HJ с 2 сек. до 1 необходимо увеличить количество процессов с 10 до 20 (см. рис.3), а это может оказаться экономически нецелесообразным.

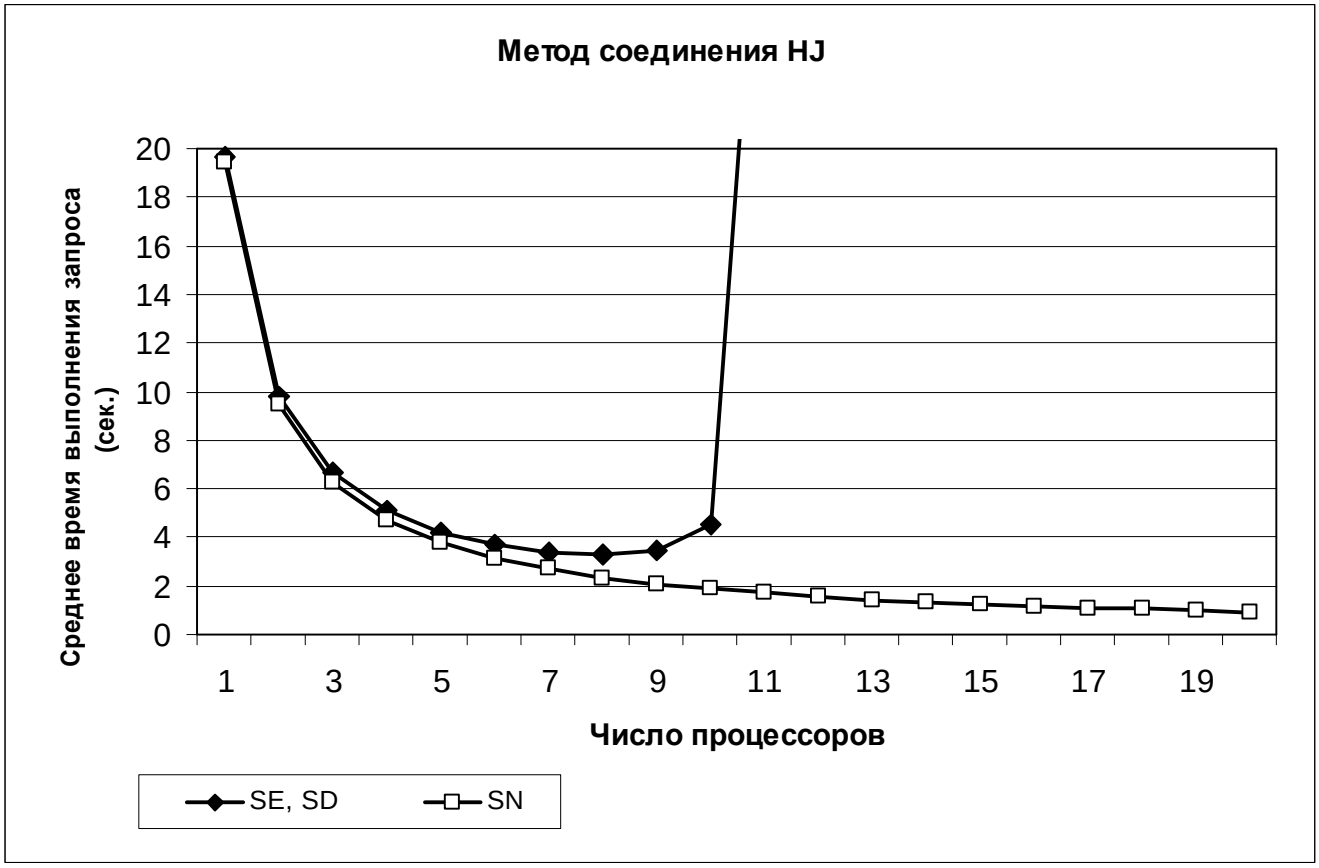

Рис. 3. Зависимость математического ожидания времени выполнения соединения таблиц в архитектурах SE, SD и SN методом HJ от числа процессоров.

Заключение

Получены преобразования Лапласа-Стилтьеса (ПЛС) случайного времени соединения таблиц в параллельной системе баз данных для различных архитектур (SE, SD, SN) и разных методов реализации соединения (NLJ, HJ). Эти преобразования позволяют оценивать не только математические ожидания случайного времени, но и моменты более высоких порядков (например, дисперсии).

Получены выражения для математических ожиданий времени выполнения операции соединения для указанных выше вариантов. Рассмотрен практический пример расчета, позволивший сделать рад нетривиальных выводов.

В дальнейшем предполагается использовать аппарат ПЛС для оценки времени выполнения аналитических запросов к хранилищу данных, реализованному на основе параллельной системы баз данных и использующему специальные планы соединения таблиц измерений и фактов.

ЛИТЕРАТУРА

1. *Оззу М. Тамер, Валдуриз Патрик.* Распределенные и параллельные системы баз данных: [Электронный ресурс]. [http://citforum.ru/database/classics/distr_and_paral_sdb/]. Проверено 26.11.2010.
2. *Соколинский Л. Б., Цымблер М. Л.* Лекции по курсу "Параллельные системы баз данных": [Электронный ресурс]. [http://pdbs.susu.ru/CourseManual.html]. Проверено 04.12.2010.
3. *Льюис Дж.* Oracle. Основы стоимостной оптимизации. – СПб: Питер, 2007.
4. *Григорьев Ю.А., Плужников В.Л.* Оценка времени выполнения запросов и выбор архитектуры параллельной системы баз данных// Информатика и системы управления. – 2009. – № 3. – С. 3-12.
5. Производительность СУБД Oracle Database 11g при работе на сервере Sun SPARC Enterprise M9000: [Электронный ресурс]. [http://ru.sun.com/sunnews/press/2010/2010-05-18.jsp]. Проверено 26.11.2010.
6. *Варфоломеев В.А., Лецкий Э.К., Шамров М.И., Яковлев В.В*. Лекции по курсу "Операционные системы и программное обеспечение на платформе zSeries": [Электронный ресурс]. [http://www.intuit.ru/department/os/ibmzos/]. Проверено 26.11.2010.
7. *Болинджер Керри.* Врожденный параллелизм: [Электронный ресурс]. [http://www.osp.ru/os/2006/02/1156526/_p1.html]. Проверено 26.11.2010.
8. *Левин Лев.* Teradata совершенствует хранилища данных: [Электронный ресурс]. [http://www.pcweek.ru/themes/detail.php?ID=71626]. Проверено 26.11.2010.
9. Oracle Real Application Clusters Administration and Deployment Guide 11g Release 1 (11.1): [Электронный ресурс]. [http://download.oracle.com/docs/cd/B28359_01/rac.111/b28254/admcon.htm/]. Проверено 26.11.2010.
10. *Григорьев Ю.А., Плутенко А.Д.* Теоретические основы анализа процессов доступа к распределенным базам данных. – Новосибирск: Наука, 2002.
11. *Жожикашвили В.А, Вишневский В.М.* Сети массового обслуживания. Теория и применение к сетям ЭВМ. – М.: Радио и связь, 1988.
12. *Клейнрок Л.* Теория массового обслуживания. – М.: Машиностроение, 1979.
13. *Бронштейн О.И., Духовный И.М.* Модели приоритетного обслуживания в информационно-вычислительных системах. – М.: Наука, 1976.
14. Форум/Использование СУБД/Oracle/CPUSPEED на IntelXeon 5500 (Nehalem): [Электронный ресурс]. [http://www.sql.ru]. Проверено 02.12.2010.

E-mail:
Григорьев Ю.А. – grigorev@iu5.bmstu.ru
Плужников В.Л. - VPluzhnikov@croc.ru