УДК 681.324

СПОСОБ РЕШЕНИЯ ЗАДАЧИ ПАРАЛЛЕЛЬНОЙ ОБРАБОТКИ ЗАПРОСОВ К БАЗАМ ДАННЫХ

С.В. Осиевский
(представил проф. А.В. Королёв)

Приведена модель сети Петри для анализа процедуры формирования очереди выполнения запросов пользователей. Предложен алгоритм построения дерева достижимости сети, проведен сравнительный анализ с алгоритмом Литтла.

Исследование жизненного цикла баз данных позволяет выделить основные задачи, возникающие на этапе эксплуатации баз данных (БД). На данном этапе группой администратора баз данных (АдБД) решаются задачи эффективного управления процессами реорганизации БД, реструктуризации канонических и логических структур БД, эффективного обслуживания множества запросов и прикладных программ БД, обеспечения целостности и непротиворечивости данных, развития структур БД при изменениях требований на обработку и ограничений на структуры данных [1]. Все эти задачи связаны с разработкой научно-обоснованной методологии сопровождения и поддержки БД в актуальном состоянии. Эффективная методология поддержки функционирования и развития БД обеспечивает решение следующих частных задач:

1) анализ стратегий и выбор оптимальных периодов реорганизации БД с учетом различных типов структурных изменений предметных областей пользователей;

2) реорганизация канонических и логических структур распределенных и локальных БД с учетом ограничений, накладываемых на структуры данных;

3) реструктуризация канонических структур БД и информационных требований пользователей при изменении требований на обработку и развитие предметных областей;

4) разрешение различного рода конфликтов и противоречий, возникающих в процессе функционирования БД, и обеспечение согласованного использования ее общих информационных ресурсов.

Решение каждой из вышеперечисленных задач основывается на соответствующих методах и принципах и для их комплексного решения требуется прежде всего нахождение оптимального решения для каждой задачи индивидуально.

Рассмотрим вопрос представления моделей запросов пользователей посредством сетей Петри и соответственно возможность нахождения

оптимального решения анализа сети Петри для различных типов запросов. Решение данной задачи может быть использовано АдБД для разрешения различного рода конфликтов и противоречий, возникающих в процессе функционирования БД, и обеспечения согласованного использования ее общих информационных ресурсов.

Анализ процессов обслуживания запросов пользователей и транзакций позволяет выделить четыре основных типа операций, выполняемых над БД [2]: выборка данных из БД в соответствии с условиями запроса пользователя; обновление данных в БД; добавление новых элементов в БД (операция вставки данных); удаление устаревших или ненужных элементов из БД (операция удаления данных). Операция выборки не приводит к изменению состава или структуры БД. Она может выполняться всеми категориями пользователей БД: конечными пользователями, прикладными и системными программистами, группой администратора банка данных. Остальные операции связаны с изменением состава и (или) структуры БД. Поэтому эти операции над БД находятся в компетенции группы администратора базы данных, который принимает решение о модификации БД.

Структура запросов пользователей с учетом введенных четырех основных (базовых) типов операций над информационными ресурсами БД может быть представлена в виде последовательности тех или иных операций над информационными ресурсами БД, выполняемыми в соответствии с условиями запросов.

Структуру запроса при его обслуживании можно представить в виде двудольного ориентированного графа $\mathbf{G_K\ (N,F)}$, где первому типу вершины графа (множеству **N**) соответствуют логические записи, выбираемые при выполнении **k**-го запроса, а второму типу вершин (множеству **F**) соответствуют типы операций манипулирования данными (выборки, обновления, вставки или удаления), выполняемые над соответствующими записями БД. Ориентация дуг графа $\mathbf{G_K(N,F)}$ означает следующее: дуга от элемента множества **N** к элементам множества **F** – использование данных в операции, а от элемента множества **F** к элементу множества **N** – формирование результатов выполнения той или иной операции. На рис. 1 приведен двудольный граф, построенный для запросов первого типа.

Результатом операции $\mathbf{f_1}$, является формирование в рабочей области оперативной памяти запроса искомых записей $\mathbf{N_1^{ОП}}$, результатом операции $\mathbf{f_2}$ – формирование искомых записей $\mathbf{N_2^{ОП}}$, результатом операции $\mathbf{f_3}$ – формирование искомых записей $\mathbf{N_{рез}^{ОП}}$ с учетом результатов операции $\mathbf{f_2}$.

Общая модель структуры запросов пользователей на языке сетей Петри формируется путем композиции ее из типовых фрагментов с учетом последовательности реализации отдельных операций и используемых при этом данных. Для построения моделей реализации отдельных типов операций, в виде соответствующих фрагментов на языке сетей

Петри, процедурам манипулирования данными (множество **F**) ставятся в соответствие переходы сетей (множество **T**), а используемым при этом записям БД (множеству **N**) – позиции сетей Петри (множество **P**). В этом случае структура двудольного графа запроса будет однозначно определять структуру соответствующей сети Петри. Рассмотрим построение моделей фрагментов для операции обновления, являющейся одной из составляющих запроса на корректировку содержимого базы данных.

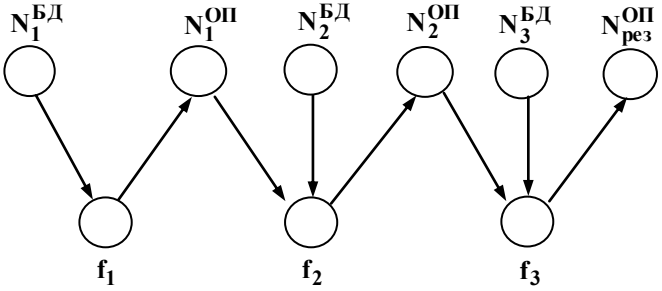

Рис. 1. Пример двудольного графа для запросов на выборку данных

Операция обновления представляет собой процесс изменения содержимого полей экземпляров отобранной для обновления записи БД и загрузки обновленной записи в БД. Данная операция включает следующую последовательность процедур: выборку записи БД, подлежащей обновлению, обновление полей записи на основании данных обновления; загрузку обновленной записи в БД. Для успешного выполнения операции данные обновления должны быть предварительно подготовлены в рабочей области оперативной памяти запросов. Модель операции обновления в виде сети Петри приведена на рис. 2.

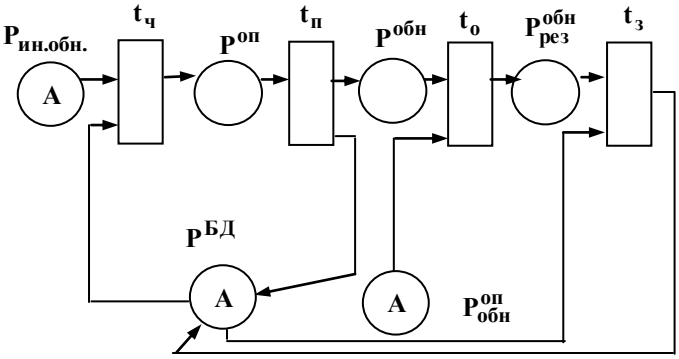

Рис. 2. Модель операции обновления в виде сети Петри

Результатом выполнения операции обновления являются отобранные из обновляемой записи экземпляры или поля (элементы данных), подле-

жащие обновлению на основании заданных алгоритмов обработки данных. Входными позициями перехода (процедуры обновления) являются отобранные на предыдущем шаге экземпляры (поля) записи и подготовленные в оперативной памяти данные обновления, моделируемые позицией $\mathbf{P}_{\text{обн}}^{\text{оп}}$. Срабатывание перехода $\mathbf{t_0}$ и попадание маркера в позицию $\mathbf{P}_{\text{рез}}^{\text{обн}}$ свидетельствует о реализации процедуры обновления и формировании результатов обновления. Запись обновленных данных моделируется переходом $\mathbf{t_з}$. Входными позициями $\mathbf{t_з}$ являются $\mathbf{P}_{\text{рез}}^{\text{обн}}$ и $\mathbf{P}^{\text{БД}}$, первая из которых моделирует результат операции обновления, а вторая – использование обновляемой записи БД, которая в момент выполнения процедуры загрузки должна быть заблокирована от использования ее в других записях (прикладных программах). Выходной позицией перехода $\mathbf{t}$ является обновленная запись БД, моделируемая позицией $\mathbf{P}^{\text{БД}}$. Попадание маркера в $\mathbf{P}^{\text{БД}}$ свидетельствует о завершении операции обновления. Поскольку операция обновления приводит к изменению содержимого обновляемой записи, а маркер в виде буквы **А** соответствует начальному (до обновления) содержимому записи, то попадание маркера в виде буквы **А** в позицию после срабатывания перехода $\mathbf{t_з}$, не позволит отразить факт выполнения операции обновления. Поэтому срабатывание перехода $\mathbf{t_з}$ моделируют попаданием в позицию $\mathbf{P}^{\text{БД}}$ специального маркера в виде звездочки *. Начальная разметка сети Петри для операции обновления имеет вид $\mathbf{M_{00} = (1, 0, 1, 0, 1, 0)}$. Разметка сети Петри, получаемая в результате выполнения операции обновления с учетом специального маркера, будет иметь вид $\mathbf{M_{00} = (0, 0, *, 0, 0, 0)}$.

Модели остальных операций, выполняемых над данными в процессе реализации вышеперечисленных операций, строятся аналогично вышеизложенной методике.

При использовании моделей сетей Петри основными анализируемыми характеристиками являются достижимость, безопасность, активность и последовательность запусков переходов.

Содержательно задача анализа достижимости сводится к следующей: для заданной сети Петри $\mathbf{S = (P, T, F, H, M_0)}$ с маркировкой μ определить достижима ли из нее некоторая новая маркировка μ' [1]. Поскольку некоторая маркировка сети Петри отражает состояние ее в некоторый момент времени, то смена маркировок моделирует изменение состояния системы. Например, операция обновления данных (рис. 2) приводит к следующей последовательности смены состояний БД. В исходном состоянии (при начальной разметке $\mathbf{M_{00} = (l.0,1,0,l,0)}$ запись, моделируемая позицией $\mathbf{P}^{\text{БД}}$, активна и может быть использована в операции чтения $\mathbf{t_ч}$, данные обновления $\mathbf{P}_{\text{обн}}^{\text{оп}}$ активны, но не могут быть использованы при начальной разметке. При

реализации перехода происходит смена разметки $\mathbf{M_0 \rightarrow M_1 = (0, 1, 0, 0, 1, 0)}$. В этом случае текущее состояние БД характеризуется тем, что запись $\mathbf{P^{БД}}$ становится пассивной и недоступной для других процессов пока не выполнится переход $\mathbf{t_п}$ и т.д.

При моделировании процессов обслуживания множества запросов пользователей на языке сетей Петри используются два основных метода – построение и анализ дерева достижимости маркировок сети и матричные уравнения [3, 4].

Рассмотрим первый из них. Дерево достижимости представляет множество достижимости сети Петри. Множество достижимых маркировок сети можно получить только путем полного анализа функционирования сети, что соответствует построению графа достижимости и его определению [5]. Существуют два типа алгоритмов построения таких графов, различающиеся тем, в какой последовательности выбираются маркировки для анализа: по ширине или по глубине графа. Первый тип объединяет алгоритмы прямого перебора, второй использует идеи алгоритма «ветвей и границ». Алгоритмы прямого перебора основываются на просмотре элементов формируемого множества маркировок **M** в порядке, соответствующем расстоянию рассматриваемой вершины $\mathbf{M_i}$ от начальной $\mathbf{M_0}$. Множество **M** расширяется на каждом шаге за счет включения в него маркировок, порождаемых срабатываниями переходов, возбужденных при всех $\mathbf{M_i}$, включенных в **M** на предыдущем шаге. Реализация алгоритмов данного класса алгоритмов представлена в [6].

Для машинной реализации алгоритмы второго типа являются более приемлемыми, во-первых, из-за того, что предоставляют возможность попутно получать последовательности срабатываний, и, во-вторых, из-за того, что обеспечивают упорядоченность перебора. Примерами таких алгоритмов могут служить алгоритм Литтла и алгоритм локального поиска. Алгоритмы построения множества достижимости имеют экспоненциальную асимптотическую оценку сложности, как по времени, так и по памяти.

Учитывая вышеизложенное применительно к модели обработки запросов пользователей в виде сети Петри предлагается алгоритм построения дерева достижимости, основанный на ранговом подходе, в котором существенно снижена временная сложность за счет отсечения неперспективных вариантов решения на каждом ранге пути.

Пусть все возможные состояния некоторой системы определяются графом **M(F,N)** с **n** вершинами, где вершины соответствуют возможным состояниям системы. Такое пространство состояний можно представить в виде стянутого дерева путей **D** (рис. 3). Дерево всех путей **D** содержит **(n – 1)** горизонтальную линейку и **(n – 1)** ярус. Для прочтения путей на каждой горизонтальной линейке можно бывать только один раз. Исходя из стянутого дерева путей, для произвольной вершины **j** множество путей, ведущих в эту вершину из некоторой вершины **s**,

можно представить в следующем виде:

$$\mathbf{m_s(j) = m_{sj}^{r=1} \cup m_{sj}^{r=2} \cup .... \cup m_{sj}^{r=n-1};\ j = \overline{(1,n-1)}}, \qquad (1)$$

где $\mathbf{m_{sj}^r = \{\mu_{sj}^r\}}$ – подмножества путей из произвольной вершины **s** в некоторую вершину **j** графа **M(F,N)**, ранга **r**.

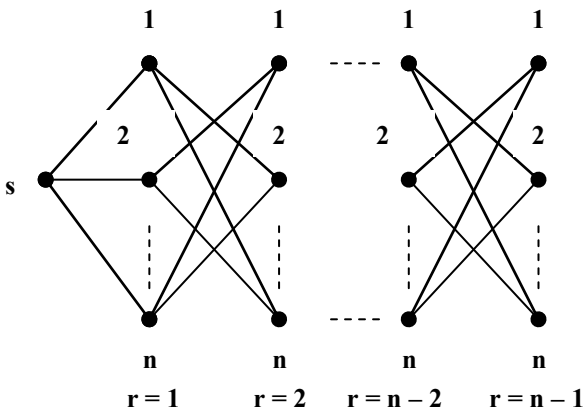

Рис. 3. Стянутое дерево всех путей **D** графа **M(F,N)**

Используя граф **D** и введя правила формирования путей следующего ранга мы можем из произвольной вершины **s** поэтапно строить пути произвольного ранга вплоть до ранга **r = n - 1**. Таким образом, для решения задачи определения кратчайшего гамильтонового пути в графе **D** нужно построить кратчайшие пути ранга **r = n - 1** от вершины **1, 2, …, n** ко всем остальным вершинам графа и потом среди них выбрать кратчайший. Если на основе подмножеств путей $\mathbf{m_{sj}^{r=1}}$ в графе **D** строить подмножества $\mathbf{m_{sj}^{r=2}}$ и так далее до $\mathbf{m_{sj}^{r=n-1}}$, то мы вынуждены будем построить **(n-1)!** путей, поэтому для формирования путей вводится процедура А, позволяющая отсекать неперспективные пути. Для отсечения неперспективных вариантов в процедуре А предлагается использовать принцип оптимизации по направлению к вершине **p**, при формировании путей следующего ранга $\mathbf{m_{sp}^{r+1}}$ на основе путей предыдущего ранга $\mathbf{m_{sj}^r}$, который определяется следующим рекуррентным соотношением

$$\mathbf{\mu_{sp}^{r+1} = \min_j \{\{\mu_{sj}^r\} \cup (j,p)\};\ j = \overline{(1,n)};\ p = \overline{(1,n)};\ j \neq p}\ , \qquad (2)$$

где **(j,p)** - ребро графа **D**; **n** - число различных вершин в графе **D**.

Процедура А. Из вершины **s** строятся все возможные пути ранга **r = 1** ко всем вершинам графа **D**, далее, используя пути ранга **r = 1**, строятся все возможные пути ранга **r = 2** и на их основе формируются пути следующего ранга, с использованием рекуррентного соотношения (2), и т. д. до построения путей ранга **r = n - 1**. (Следует отметить, что если в процессе применения рекуррентного соотношения (2) возникают несколько кратчайших путей одинаковой длины, то необходимо их все строить на следующем ранге.)

Можно выделить следующие особенности работы процедуры А. В процессе ее работы может возникать две ситуации. Первая, когда процедура А на каждом шаге построила пути в множества $\mathbf{m}_{sj}^{r}$, т.е. принцип оптимальности работы процедуры не нарушался, и вторая, когда к одной из вершин ни одного пути построить нельзя. Последнее обстоятельство возможно в двух случаях:

а) если анализируемый граф неполный и к вершине **p** не существует пути некоторого ранга **r = k**;

б) когда некоторая вершина **p** войдет во все пути предыдущего ранга.

Нетрудно видеть, что в первой ситуации процедура А не теряет оптимальное решение, а во второй ситуации (случай б) оптимальное решение может быть потеряно, поскольку принцип оптимальности работы процедуры нарушается. Это означает, что на основе процедуры А мы можем построить только приближенный алгоритм решения задачи. Однако после нарушения принципа оптимальности последующее продление путей с использованием рекуррентного соотношения (2) позволит минимально отклоняться от оптимального решения.

На рис. 4 и 5 показаны зависимости числа обработанных векторов и элементарных операций от размерности решаемой задачи, для алгоритма использующего подходы «метода ветвей и границ» и соответственно для разработанного алгоритма на основе рангового подхода. Из представленных рисунков следует, что с увеличением размерности задачи для разработанного алгоритма число элементарных операций и число обрабатываемых векторов стабилизируется и в худшем случае решение имеет сложность $C*n^3$, где C – константа меньше 1, в то время, когда для алгоритма Литтла наблюдается существенный рост обеих показателей.

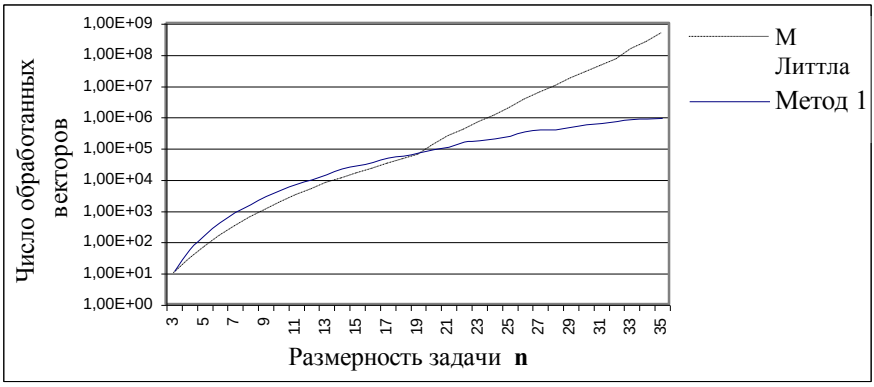

Рис. 4. Зависимость числа обработанных векторов от размерности задачи

Таким образом, предлагаемый способ решения задачи параллельной обработки запросов к базам данных может быть использован АБД для

составления оптимальных расписаний выполнения запросов к базам данных, как на этапах реорганизации, так и на этапах сопровождения функционирования систем управления включающих базы данных. При этом предполагается существенное уменьшение времени на реализацию

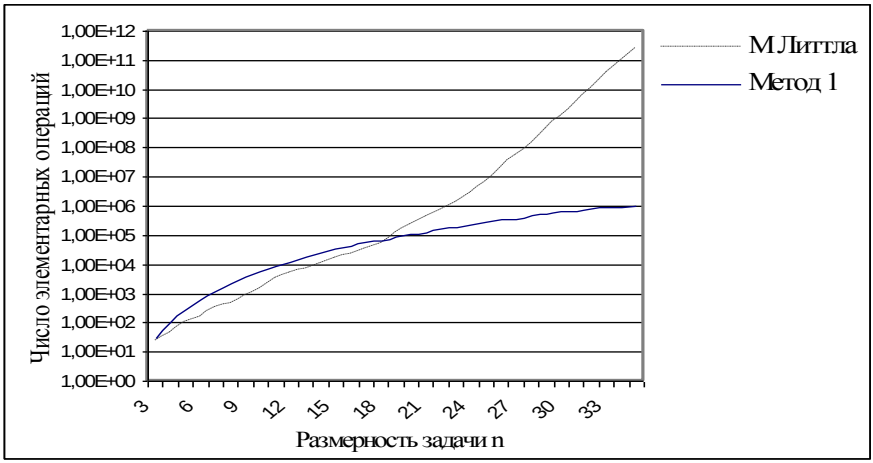

Рис. 5. Зависимость числа элементарных операций от размерности задачи

процедуры формирования очереди выполнения операций в запросах за счет уменьшения числа элементарных операций и числа обрабатываемых векторов в процедуре формирования очереди выполнения операций запроса.

ЛИТЕРАТУРА

1. *Кульба В.В., Ковалевский С.С., Косяченко С.А. Теоретические основы проектирования оптимальных структур распределенных баз данных. – М.: Синтег, 1999. – 461 с.*
2. *Томас Коннолли, Каролин Бег. Базы данных: проектирование, реализация и сопровождение. – М.: Вильямс, 2000. – 1114 с.*
3. *Мамиконов А.Г., Кульба В.В. Использование сетей Петри при проектировании систем обработки данных. – М.: Наука, 1988. – 103 с.*
4. *Котов В.Е. Сети Петри. – М.: Наука, 1984. – 160 с.*
5. *Ачасова С.М., Бандман О.Л. Корректность параллельных вычислительных процессов. – Н-ск: Наука, 1990. – 253 с.*
6. *Гудман С., Хидетниеми С. Введение в разработку и анализ алгоритмов. – М.: Мир, 1981. – 365 с.*

Поступила 1.08.2002

***ОСИЕВСКИЙ Сергей Валерьевич**, адъюнкт кафедры Харьковского военного университета. В 1997 году окончил Харьковский военный университет. Область научных интересов – управление процессами сопровождения и развития баз данных в АСУ.*