КІБЕРНЕТИКА

В.Н. РЕДЬКО, Д.Б. БУЙ, И.С. КАНАРСКАЯ, А.С. СЕНЧЕНКО

УДК 004.655

ТОЧНЫЕ ОЦЕНКИ ВРЕМЕННО́Й СЛОЖНОСТИ РЕАЛИЗАЦИИ АЛГОРИТМОВ ТЕОРЕТИКО-МНОЖЕСТВЕННЫХ ОПЕРАЦИЙ В ТАБЛИЧНЫХ АЛГЕБРАХ

Аннотация. Исследованы алгоритмы, реализующие операции пересечения, объединения и разности в табличных алгебрах. Предложены модификации наиболее распространенных алгоритмов, позволяющие сократить количество вычислений. На основе оценки сложности в худшем случае и в среднем для модифицированных алгоритмов найден наиболее быстрый алгоритм для каждой операции. Разработана программная система, экспериментально подтверждающая теоретические оценки.

Ключевые слова: сложность алгоритма, база данных, табличная алгебра.

ВВЕДЕНИЕ

В настоящее время наиболее распространены реляционные (табличные) базы данных (БД), основанные на реляционных моделях, введенных Э. Коддом [1, 2]. Реляционная алгебра базируется на теории множеств, а также теории отношений. Табличные алгебры, изложенные в [3–5], построены на основе реляционных алгебр Кодда и существенно их развивают. Они составляют теоретический фундамент языков запросов современных табличных БД. Элементы носителя табличной алгебры уточняют реляционные структуры данных, а сигнатурные операции строятся на базе основных табличных манипуляций в реляционных алгебрах и SQL-подобных языках.

Оптимизация запросов — одно из наиболее важных и интересных направлений исследований и разработок в области БД. Его значимость определяется тем, что от развитости блока оптимизации запросов зависит общая производительность любой SQL-ориентированной БД. При решении задач оптимизации используют самые разнообразные подходы и методы из различных областей математики и информатики. На протяжении последних лет эти факторы привлекают к данному направлению внимание многих исследователей [6–10].

В данной работе рассматриваются три традиционные бинарные теоретико-множественные операции: пересечение, объединение и разность таблиц. Предложены алгоритмы их выполнения и найдена точная сложность этих алгоритмов, что можно применять для оптимизации запросов.

Существенно, что при этом явно используется специфика таблиц — множеств односхемных строк (точные определения приведены ниже). Одним из проявлений этой ситуации является тот факт, что найденные функции сложности зависят от количества строк и атрибутов исходных таблиц.

Отметим также, что анализ доступной литературы показал: несмотря на естественную постановку задачи в работе, аналогичные результаты по данному направлению отсутствуют (по крайней мере, авторам они не известны).

1. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

Зафиксируем некоторое непустое множество атрибутов $\mathcal{R} = \{A_1, \dots, A_n\}$. Произвольное подмножество множества $\mathcal{R}$, в частности пустое, назовем схемой. Строкой s схемы R называется множество пар $s = \{(A'_1, d_1), \dots, (A'_k, d_k)\}$, проекция которого по первой компоненте равна R, причем атрибуты $A'_1, \dots, A'_k$ попарно различны, т.е. строка является функциональным бинарным отношением. Элементы d_i (значения атрибутов) выбираются из некоторого зафиксированного множества. Таблица T схемы R — конечное множество строк схемы R, количество строк в таблице T обозначим $|T|$. Далее, если не оговорено противное, рассматриваются таблицы схемы R с количеством атрибутов n. На множестве всех таких таблиц введены параметрические операции объединения ($\underset{R}{\cup}$), пересечения ($\underset{R}{\cap}$) и разности ($\underset{R}{-}$) таблиц как ограничения одноименных теоретико-множественных операций.

Табличной алгеброй называется частичная алгебра с носителем (множеством всех таблиц произвольной схемы) и девятью операциями: объединением, пересечением, разностью, проекцией, селекцией, активным дополнением, соединением, делением и переименованием; определения шести из этих операций, не используемых в данной работе, приведены в [3, 11].

Рассмотрим основные подходы к оценке сложности алгоритмов, следуя [10]. Вычислительной сложностью алгоритма называется функция зависимости «объема» работы, которая им выполняется, от «размера» входных данных. Исследуют два вида вычислительной сложности: временну́ю, оценивающую время выполнения алгоритма (количество выполняемых вычислений), и емкостную, оценивающую количество ячеек памяти, необходимое для работы алгоритма. Считаем, что сложность элементарных операций одинакова (т.е. используется так называемый равномерный вес). Выделяют три типа временно́й сложности алгоритма: сложность в худшем случае, сложность для почти всех входов и сложность в среднем. Исследование временно́й сложности алгоритма в худшем случае существенно проще, чем исследование других двух типов сложности, хотя при анализе временно́й сложности обработки запросов наиболее приемлемым является использование сложности в среднем, поскольку именно она дает возможность оценить эффективность обработки запросов.

В монографии [12] выделены основные преобразования таблиц: добавление $ADD(T, s)$, удаление $DEL(T, s)$ строки s и замена $CH(T, s, s_1)$ строки s строкой s_1 в таблице T. В алгоритмах, предложенных в данной работе, используются первые два преобразования.

Постулируем, что сложность преобразований $ADD(T, s)$ и $DEL(T, s)$ в худшем случае и в среднем равняется числу атрибутов в схеме, т.е. n.

Далее рассмотрим алгоритмы выполнения операций пересечения, объединения и разности в табличных алгебрах: для каждой операции сначала исследуется наиболее естественный (на наш взгляд) алгоритм, после этого предлагаются его модификации, позволяющие уменьшить количество вычислений. Затем вычисляются временны́е сложности в худшем случае и в среднем для каждого из этих алгоритмов.

2. АЛГОРИТМЫ ПЕРЕСЕЧЕНИЯ, ОБЪЕДИНЕНИЯ И РАЗНОСТИ ТАБЛИЦ

2.1. Операция пересечения таблиц

Алгоритм А1.1. С каждой строкой $s_1 \in T_1$ сравниваем все строки таблицы T_2. Если существует такая строка $s_2 \in T_2$, что $s_1 = s_2$, то строку s_1 добавляем в результирующую таблицу T и переходим к следующей строке таблицы T_1.

Алгоритм А1.2. С каждой строкой $s_1 \in T_1$ сравниваем все строки таблицы T_2. Если существует такая строка $s_2 \in T_2$, что $s_1 = s_2$, то строку s_1 добавляем в результирующую таблицу T, строку s_2 удаляем из таблицы T_2 и переходим к следующей строке таблицы T_1.

Алгоритм А1.3. Если $|T_1| \le |T_2|$, то с каждой строкой $s_1 \in T_1$ сравниваем все строки таблицы T_2. Если не существует такой строки $s_2 \in T_2$, что $s_1 = s_2$, то строку s_1 удаляем из таблицы T_1, а если такая строка существует, то переходим к следующей строке таблицы T_1; результат содержится в таблице T_1. Если $|T_1| > |T_2|$, то таблицы T_1 и T_2 меняются местами.

Алгоритм А1.4. Если $|T_1| \le |T_2|$, то с каждой строкой $s_1 \in T_1$ сравниваем все строки таблицы T_2. Если не существует такой строки $s_2 \in T_2$, что $s_1 = s_2$, то строку s_1 удаляем из таблицы T_1, а если существует такая строка $s_2 \in T_2$, что $s_1 = s_2$, то строку s_2 удаляем из таблицы T_2 и переходим к следующей строке таблицы T_1; результат содержится в таблице T_1. Если $|T_1| > |T_2|$, то таблицы T_1 и T_2 меняются местами.

2.2. Операция объединения таблиц

Алгоритм А2.1. Добавляем все строки $s_1 \in T_1$ в новую таблицу T (которая содержит результат), затем все строки s_2 таблицы T_2 сравниваем со строками таблицы T_1. Если не существует такой строки s_1, что $s_2 = s_1$, то строку s_2 добавляем в таблицу T. Если такая строка существует, то переходим к следующей строке таблицы T_2.

Алгоритм А2.2. Добавляем все строки $s_1 \in T_1$ в новую таблицу T (которая содержит результат), затем все строки s таблицы T сравниваем со строками таблицы T_2. Если существует такая $s_2 \in T_2$, что $s = s_2$, то удаляем строку s_2 из таблицы T_2 и переходим к следующей строке таблицы T. После этого добавляем все оставшиеся строки таблицы T_2 в таблицу T.

Алгоритм А2.3. Если $|T_1| \le |T_2|$, то с каждой строкой $s_2 \in T_2$ сравниваем все строки таблицы T_1. Если существует такая строка $s_1 \in T_1$, что $s_2 = s_1$, то удаляем строку s_1 из таблицы T_1 и переходим к следующей строке таблицы T_2; после этого добавляем все оставшиеся строки из таблицы T_1 в таблицу T_2, в которой содержится результат. Если $|T_1| > |T_2|$, то таблицы T_1 и T_2 меняются местами.

2.3. Операция разности таблиц

Алгоритм А3.1. С каждой строкой $s_1 \in T_1$ сравниваем все строки таблицы T_2: если не существует такой строки $s_2 \in T_2$, что $s_1 = s_2$, то строку s_1 добавляем в новую результирующую таблицу T, а если такая строка существует, то переходим к следующей строке таблицы T_1.

Алгоритм А3.2. С каждой строкой $s_1 \in T_1$ сравниваем все строки таблицы T_2: если существует такая строка $s_2 \in T_2$, что $s_1 = s_2$, то строку s_2 удаляем из таблицы T_2 и переходим к следующей строке таблицы T_1. Если такой строки не существует, то строку s_1 добавляем в новую таблицу T, которая содержит результат.

Алгоритм А3.3. С каждой строкой $s_1 \in T_1$ сравниваем все строки таблицы T_2: если существует такая строка $s_2 \in T_2$, что $s_1 = s_2$, то строку s_1 удаляем из таблицы T_1. Результат содержится в таблице T_1.

Алгоритм А3.4. С каждой строкой $s_1 \in T_1$ сравниваем все строки таблицы T_2: если существует такая строка $s_2 \in T_2$, что $s_1 = s_2$, то строку s_1 удаляем из таблицы T_1, а строку s_2 — из таблицы T_2. Результат содержится в таблице T_1.

Алгоритм А3.5. С каждой строкой $s_2 \in T_2$ сравниваем все строки таблицы T_1: если существует такая строка $s_1 \in T_1$, что $s_2 = s_1$, то строку s_1 удаляем из таблицы T_1 и переходим к следующей строке таблицы T_2. Результат содержится в таблице T_1.

3. ВЫЧИСЛЕНИЕ СЛОЖНОСТИ АЛГОРИТМОВ

3.1. Сложность в худшем случае

Найдем сложность в худшем случае для всех ранее предложенных алгоритмов, в которых полагаем $|T_1| = m_1$ и $|T_2| = m_2$. Поскольку в каждом алгоритме количество вычислительных действий зависит от количества строк, принадлежащих пересечению исходных таблиц (далее эти строки будем называть строками таблицы $T_1 \bigcap\limits_R T_2$), для вычисления сложности каждого алгоритма сначала была найдена функциональная зависимость сложности от числа строк таблицы $T_1 \bigcap\limits_R T_2$, а затем, в качестве сложности в худшем случае, найден максимум этой функции, с учетом того, что $j = \left| T_1 \bigcap\limits_R T_2 \right|$ — натуральное число.

Заметим, что для каждой строки $s_1 \in T_1$ при поиске равной ей строки s_2 наибольшее количество вычислений выполняется в случае, когда перебрано максимально возможное количество строк таблицы T_2 (m_2), а при сравнении с s_1 других (отличных от s_2) строк таблицы T_2 выполнено максимально возможное количество сравнений значений атрибутов (n). Также отметим, что при сравнении строк текущей таблицы T_1 со всеми строками таблицы T_2, если заранее известно, что $\left| T_1 \bigcap\limits_R T_2 \right| = j$, то необходимо осуществить не более чем $m_2 + (m_2 - 1) + \ldots + (m_2 - j + 1) = jm_2 - \dfrac{j(j-1)}{2}$ операций сравнения строк. Эта величина достигается в случае, когда для нахождения одной из строк (s^1) таблицы $T_1 \bigcap\limits_R T_2$ выполнено m_2 сравнений со всеми строками таблицы T_2, для некоторой другой строки (s^2) таблицы $T_1 \bigcap\limits_R T_2$ — $m_2 - 1$ сравнение (если для строки s^1 выполнено точно j сравнений, то эта строка в таблице T_2 рассматривалась последней, а поскольку $s^1 \neq s^2$, строка s^2 в таблице T_2 не может быть рассмотрена последней), … , для m-й строки (s^m) таблицы $T_1 \bigcap\limits_R T_2$ необходимо выполнить $m_2 - j + 1$ сравнение.

В алгоритме А1.1 для каждой из $m_1 - j$ строк таблицы T_1, не принадлежащих таблице T_2, выполняется не более чем $m_2 n$ вычислений, а для j строк, принадлежащих таблице $T_1 \bigcap\limits_R T_2$, при поиске в таблице T_2 равных им строк в худшем случае выполняется $n[m_2 + (m_2 - 1) + \ldots + (m_2 - j + 1)]$ вычислений; также j раз проводится процедура добавления строки из таблицы T_1 в таблицу T, поэтому сложность алгоритма А1.1 равна

$$(m_1 - j)m_2 n + n[m_2 + (m_2 - 1) + (m_2 - 2) + \ldots + (m_2 - j + 1)] + jn = m_1 m_2 n - \frac{n}{2} j(j-3).$$

Эта функция принимает максимальное значение при $j = 1$ и $j = 2$, которое в обоих случаях равно $m_1 m_2 n + n$.

В алгоритме А1.2 осуществляют те же действия, что и в алгоритме А1.1, а также j раз выполняется процедура удаления строки из таблицы T_2, поэтому его сложность равна

$$(m_1 - j)m_2 n + n[m_2 + (m_2 - 1) + (m_2 - 2) + \ldots + (m_2 - j + 1)] + 2jn = m_1 m_2 n - \frac{n}{2} j(j - 5).$$

Эта функция принимает максимальное значение при $j = 2$ и $j = 3$, которое в обоих случаях равно $m_1 m_2 n + 3n$.

Вычислим сложность в худшем случае для алгоритма А1.3. Для сравнения величин m_1 и m_2 необходимо одно вычисление. Пусть для определенности $m_1 \le m_2$. Для каждой из $m_1 - j$ строк таблицы T_1, не принадлежащих таблице T_2, при поиске в таблице T_2 равной ей строки выполняется не более чем $m_2 n$ вычислений, а для j строк, принадлежащих таблице $T_1 \underset{R}{\cap} T_2$, в худшем случае выполняется $n[m_2 + (m_2 - 1) + \ldots + (m_2 - j + 1)]$ вычислений; также $m_1 - j$ раз проводится процедура удаления строки из таблицы T_1, поэтому сложность алгоритма А1.3 равна

$$(m_1 - j)m_2 n + n[m_2 + (m_2 - 1) + \ldots + (m_2 - j + 1)] + (m_1 - j)n + 1 =$$

$$= m_1 m_2 n + m_1 n + 1 - \frac{n}{2} j(j + 1).$$

Эта функция принимает максимальное значение при $j = 0$, равное $m_1 m_2 n + m_1 n + 1$. Для общего случая сложность в худшем случае алгоритма А1.3 равна $m_1 m_2 n + \min\{m_1, m_2\} n + 1$.

В алгоритме А1.4 осуществляют те же действия, что и в алгоритме А1.3, а также j раз выполняется процедура удаления строки из таблицы T_2, поэтому сложность алгоритма А1.4 для случая $m_1 \le m_2$ равна $m_1 m_2 n + m_1 n + 1 - \frac{n}{2} j(j - 1)$. Эта функция принимает максимальное значение при $j = 0$ и $j = 1$, которое в обоих случаях равно $m_1 m_2 n + m_1 n + 1$. Для общего случая сложность алгоритма А1.4 равна $m_1 m_2 n + \min\{m_1, m_2\} n + 1$.

Для алгоритма А2.1 при добавлении всех строк таблицы T_1 в новую таблицу T выполняется $m_1 n$ вычислительных действий. Затем для каждой из $m_2 - j$ строк таблицы T_2, не принадлежащих таблице T_1, при поиске совпадающей с ней строки проводится не более чем $m_1 n$ вычислений, также при выполнении процедуры добавления строки из таблицы T_2 в таблицу T для каждой из этих строк выполняется n вычислений; для j строк, принадлежащих таблице $T_1 \underset{R}{\cap} T_2$, в худшем случае выполняется $n[m_1 + (m_1 - 1) + \ldots + (m_1 - j + 1)]$ вычислений, поэтому сложность алгоритма А2.1 равна

$$m_1 n + (m_2 - j)m_1 n + (m_2 - j)n + n[m_1 + \ldots + (m_1 - j + 1)] = m_1 m_2 n + m_1 n + m_2 n - \frac{n}{2} j(j + 1).$$

Эта функция принимает максимальное значение при $j = 0$, в этом случае ее значение равно $m_1 m_2 n + m_1 n + m_2 n$.

Для алгоритма А2.2 при добавлении всех строк таблицы T_1 в новую таблицу T выполняется $m_1 n$ вычислительных действий. Затем при сравнении строк таблицы T со строками таблицы T_2 для каждой из $m_1 - j$ строк таблицы T, не принадлежащих таблице T_2, выполняется не более чем $m_2 n$ вычислений, а для j строк таблицы T, принадлежащих таблице T_2, выполняется не более чем $n[m_2 + (m_2 - 1) + \ldots + (m_2 - j + 1)]$ вычислений, также j раз выполняется процедура удаления строки из таблицы T_2. После этого при добавлении всех оставшихся строк таблицы T_2 в таблицу T выполняется $(m_2 - j)n$ вычислительных действий.

Поэтому сложность алгоритма А2.2 равна

$$m_1 n + (m_1 - j)m_2 n + n[m_2 + \ldots + (m_2 - j + 1)] + jn + (m_2 - j)n =$$
$$= m_1 m_2 n + m_2 n + m_1 n - \frac{n}{2} j(j-1).$$

Эта функция принимает максимальное значение при $j = 0$ и $j = 1$, которое в обоих случаях равно $m_1 m_2 n + m_2 n + m_1 n$.

Найдем сложность в худшем случае алгоритма А2.3. Для сравнения величин m_1 и m_2 необходимо одно вычисление. Пусть для определенности $m_1 \le m_2$. Для каждой из $m_2 - j$ строк таблицы T_2, не принадлежащих таблице T_1, при поиске в таблице T_1 равной ей строки выполняется не более чем $m_1 n$ вычислений, а для j строк, принадлежащих таблице $T_1 \bigcap\limits_R T_2$, в худшем случае выполняется $n[m_1 + (m_1 - 1) + \ldots + (m_1 - j + 1)]$ вычислений, также j раз проводится процедура удаления строки из таблицы T_1. После этого $m_1 - j$ раз выполняется процедура добавления строк из таблицы T_1 в таблицу T_2. Поэтому сложность алгоритма А2.3 равна

$$(m_2 - j)m_1 n + n[m_1 + (m_1 - 1) + \ldots + (m_1 - j + 1)] + jn + (m_1 - j)n + 1 =$$
$$= m_1 m_2 n + m_1 n + 1 - \frac{n}{2} j(j-1).$$

Эта функция принимает максимальное значение при $j = 0$ и $j = 1$, которое в обоих случаях равно $m_1 m_2 n + m_1 n + 1$. Для общего случая сложность алгоритма А2.3 равна $m_1 m_2 n + \min\{m_1, m_2\} n + 1$.

В алгоритме А3.1 для каждой из $m_1 - j$ строк таблицы T_1, не принадлежащих таблице T_2, при поиске в таблице T_2 равной ей строки выполняется не более чем $m_2 n$ вычислений, также $m_1 - j$ раз проводится процедура добавления строк из таблицы T_1 в таблицу T. Для j строк, принадлежащих таблице $T_1 \bigcap\limits_R T_2$, в худшем случае выполняется $n[m_2 + (m_2 - 1) + \ldots + (m_2 - j + 1)]$ вычислений. Таким образом, сложность алгоритма А3.1 равна

$$(m_1 - j)m_2 n + (m_1 - j)n + n[m_2 + (m_2 - 1) + \ldots + (m_2 - j + 1)] = m_1 m_2 n + m_1 n - \frac{n}{2} j(j+1).$$

Эта функция принимает максимальное значение при $j = 0$, в этом случае ее значение равно $m_1 m_2 n + m_1 n$.

В алгоритме А3.2 осуществляются те же действия, что и в алгоритме А3.1, а также j раз проводится процедура удаления строк из таблицы T_2, поэтому его сложность равна

$$(m_1 - j)m_2 n + (m_1 - j)n + n[m_2 + (m_2 - 1) + \ldots + (m_2 - j + 1)] + jn =$$
$$= m_1 m_2 n + m_1 n - \frac{n}{2} j(j-1).$$

Эта функция принимает максимальное значение при $j = 0$ и $j = 1$, которое в обоих случаях равно $m_1 m_2 n + m_1 n$.

Для алгоритма А3.3 для каждой из $m_1 - j$ строк таблицы T_1, не принадлежащих таблице T_2, при поиске совпадающей строки в худшем случае выполняется $m_2 n$ вычислений, а для строк, принадлежащих таблице $T_1 \bigcap\limits_R T_2$, — $n[m_2 + (m_2 - 1) + \ldots + (m_2 - j + 1)]$ вычислений; также j раз проводится процедура удаления строки из таблицы T_1. Поэтому сложность алгоритма А3.3 равна

$$(m_1 - j)m_2 n + n[m_2 + (m_2 - 1) + (m_2 - 2) + \ldots + (m_2 - j + 1)] + jn = m_1 m_2 n - \frac{n}{2} j(j-3).$$

Эта функция принимает максимальное значение при $j = 1$ и $j = 2$, которое в обоих случаях равно $m_1 m_2 n + n$.

В алгоритме А3.4 выполняются те же действия, что и в алгоритме А3.3, а также j раз проводится процедура удаления строк из таблицы T_2, поэтому его сложность равна

$$(m_1 - j)m_2 n + n[m_2 + (m_2 - 1) + \ldots + (m_2 - j + 1)] + 2nj = m_1 m_2 n - \frac{n}{2} j(j-5).$$

Эта функция принимает максимальное значение при $j = 2$ и $j = 3$, которое в обоих случаях равно $m_1 m_2 n + 3n$.

Для алгоритма А3.5 для каждой из $m_2 - j$ строк таблицы T_2, не принадлежащих таблице T_1, при поиске совпадения в худшем случае выполняется $m_1 n$ вычислений, а для j строк, принадлежащих таблице $T_1 \bigcap\limits_R T_2$, в худшем случае выполняется $n[m_1 + (m_1 - 1) + \ldots + (m_1 - j + 1)]$ вычислений; также j раз выполняется процедура удаления строки из таблицы T_1. Поэтому сложность алгоритма А3.5 равна

$$(m_2 - j)m_1 n + n[m_1 + (m_1 - 1) + \ldots + (m_1 - j + 1)] + jn = m_1 m_2 n - \frac{n}{2} j(j-3).$$

Эта функция принимает максимальное значение при $j = 1$ и $j = 2$, которое в обоих случаях равно $m_1 m_2 n + n$.

3.2. Сложность в среднем

Сложность в среднем алгоритмов А1.1–А3.5 зависит от многих параметров исходных таблиц. Пусть $|T_1| = m_1$, $|T_2| = m_2$, $m = \min\{m_1, m_2\}$, T_1, T_2 — таблицы схемы $R = \{A_1, \ldots, A_n\}$, мощности домена значений атрибутов $A_1, \ldots, A_n$ равны соответственно $q_1, \ldots, q_n$ и $q_1 \cdot \ldots \cdot q_n = Q$. Таким образом, существует ровно Q строк, каждая из которых независимо от остальных может входить или не входить в каждую исходную таблицу Будем считать, что распределение значений по каждому атрибуту в обеих таблицах равномерное. Несложно видеть, что если $m_1 + m_2 \le Q$, то минимальное количество строк таблицы $T_1 \bigcap\limits_R T_2$ равно 0, в противном случае минимальное количество строк таблицы $T_1 \bigcap\limits_R T_2$ равно $(m_1 + m_2) - Q$; это минимальное количество строк обозначим z. Из определения пересечения таблиц очевидно вытекает, что максимально возможное количество строк таблицы $T_1 \bigcap\limits_R T_2$ равно m. Поскольку все предложенные алгоритмы используют сравнение строк в таблицах T_1 и T_2 относительно их равенства и сложность этих алгоритмов зависит от количества строк таблицы $T_1 \bigcap\limits_R T_2$, вычислим значения вероятности $P(j)$ того, что таких строк будет в точности j ($j \in \{z, \ldots, m\}$). Затем для каждого алгоритма найдем значение $W(j)$ среднего количества вычислительных действий при условии $|T_1 \bigcap\limits_R T_2| = j$. Будем считать сложностью в среднем величину $\sum\limits_{j=z}^{m} P(j) \cdot W(j)$.

Сначала определим вероятность $P(j)$. Эти j строк, принадлежащие таблице $T_1 \bigcap\limits_R T_2$, можно выбрать C_Q^j способами, строки таблицы T_1, не принадлежащие

таблице T_2, — $C_{Q-j}^{m_1-j}$ способами, а строки таблицы T_2, не принадлежащие таблице T_1, — $C_{Q-m_1}^{m_2-j}$ способами. Всего существует $C_Q^{m_1}$ вариантов выбора строк таблицы T_1 и $C_Q^{m_2}$ вариантов выбора строк таблицы T_2. Таким образом, вероятность $P(j)$ равна

$$P(j)=\frac{C_Q^j C_{Q-j}^{m_1-j} C_{Q-m_1}^{m_2-j}}{C_Q^{m_1} C_Q^{m_2}}=\frac{m_1!\,m_2!\,(Q-m_1)!(Q-m_2)!}{j!(m_1-j)!(m_2-j)!(Q-m_1-m_2+j)!Q!}.$$

Для вычисления $W(j)$ найдем среднее количество вычислительных действий W', необходимое для сравнения строк $s^1=\{(A_1,d_1^1),\dots,(A_n,d_n^1)\}\in T_1$ и $s^2=\{(A_1,d_1^2),\dots,(A_n,d_n^2)\}\in T_2$ в случае $s^1\neq s^2$. Вероятность того, что $d_1^1=d_1^2$, равна $\frac{1}{q_1}$, соответственно вероятность того, что $d_1^1\neq d_1^2$, равна $1-\frac{1}{q_1}=\frac{q_1-1}{q_1}$. Если $d_1^1\neq d_1^2$, то проверка равенства строк s^1 и s^2 заканчивается, при этом выполнено одно вычислительное действие. Пусть $d_1^1=d_1^2$. Тогда вероятность того, что $d_2^1=d_2^2$, равна $\frac{1}{q_1q_2}$, а вероятность того, что $d_2^1\neq d_2^2$, равна $\frac{1}{q_1}\left(1-\frac{1}{q_2}\right)=$ $=\frac{q_2-1}{q_1q_2}$. Если $d_2^1\neq d_2^2$, то проверка равенства строк s^1 и s^2 заканчивается, при этом выполнено два вычислительных действия.

Аналогично, если выполняются $d_1^1=d_1^2,\dots,d_{j-1}^1=d_{j-1}^2$, то вероятность того, что $d_j^1=d_j^2$, равна $\frac{1}{q_1}\dots\frac{1}{q_j}$, а вероятность того, что $d_j^1\neq d_j^2$, равна $\frac{q_j-1}{q_1\dots q_j}$. Во втором случае проверка равенства строк s^1 и s^2 заканчивается, при этом выполнено j вычислительных действий. Поскольку случай $s^1=s^2$ невозможен (его вероятность согласно предыдущим вычислениям равна $\frac{1}{Q}$), значения всех найденных вероятностей необходимо разделить на $1-\frac{1}{Q}=\frac{Q-1}{Q}$. Таким образом,

$$W'=\frac{Q}{Q-1}\left(1\frac{q_1-1}{q_1}+2\frac{q_2-1}{q_1q_2}+\dots+n\frac{q_n-1}{Q}\right).$$

При нахождении $W(j)$ считаем, что строки в таблицах упорядочены следующим образом. В таблице T_1 сначала расположен блок из $\frac{m_1-j}{2}$ строк таблицы $T_1\underset{R}{-}T_2$, затем — блок из j строк таблицы $T_1\underset{R}{\cap}T_2$, далее — блок из $\frac{m_1-j}{2}$ оставшихся строк таблицы $T_1\underset{R}{-}T_2$. В таблице T_2 упорядочение аналогично: сначала

расположен блок из $\frac{m_2 - j}{2}$ строк таблицы $T_2 \underset{R}{-} T_1$, затем — блок из j строк таблицы $T_1 \underset{R}{\cap} T_2$, далее — блок из $\frac{m_2 - j}{2}$ оставшихся строк таблицы $T_2 \underset{R}{-} T_1$.

Вычислим значение $W(j)$ для алгоритма А 1.1. Для каждой из $(m_1 - j)$ строк таблицы T_1, не принадлежащих таблице T_2, выполнено m_2 сравнений. Для j строк таблицы T_1, принадлежащих таблице $T_1 \underset{R}{\cap} T_2$, в среднем выполнено $j\frac{m_2 + 1}{2}$ сравнений: $j\frac{m_2 - 1}{2}$ сравнений с неравными строками и j сравнений с равными строками; при каждом из j сравнений, давших равенство, было выполнено по $2n$ вычислительных действий, поэтому

$$W(j) = \left((m_1 - j)m_2 + j\frac{m_2 - 1}{2} \right) W' + 2jn = \left(m_1 m_2 - \frac{j}{2}(m_2 + 1) \right) W' + 2jn.$$

Для алгоритма А1.2 считаем, что для строк таблицы T_1, предшествующих строкам, принадлежащим таблице $T_1 \underset{R}{\cap} T_2$ (в среднем их $\frac{m_1 - j}{2}$), выполнено m_2 сравнений с неравными им строками, для каждой из $\frac{m_1 - j}{2}$ последующих строк — $(m_2 - j)$ сравнений с неравными им строками таблицы T_2. Для первой строки таблицы T_1, принадлежащей таблице $T_1 \underset{R}{\cap} T_2$, в среднем выполнено $\frac{m_2 - 1}{2}$ сравнений, для второй — $\frac{m_2 - 2}{2}$ сравнений, для j-й — $\frac{m_2 - j}{2}$ сравнений с неравными им строками таблицы T_2. Также при каждом из j сравнений, определивших равенство, выполнено по $3n$ вычислительных действий, поэтому

$$W(j) = \left(\frac{(m_1 - j)}{2} m_2 + \frac{(m_1 - j)}{2}(m_2 - j) + \frac{m_2 - 1}{2} + \ldots + \frac{m_2 - j}{2} \right) W' + 3jn =$$

$$= \left(m_1 m_2 - \frac{j}{2}\left(m_1 + m_2 + \frac{1}{2} - \frac{j}{2} \right) \right) W' + 3jn.$$

В алгоритме А1.3 количество сравнений, определивших неравенство строк, совпадает с аналогичным количеством из алгоритма А 1.1 (в алгоритме А1.3 таблицу T_1 заменяет таблица с меньшим количеством строк). Кроме того, в алгоритме А1.3 $(\min\{m_1, m_2\} - j)$ раз выполняется процедура удаления строки из таблицы с меньшим количеством строк. Также необходимо $jn + 1$ вычисление для сравнения равных строк и m_1 с m_2, поэтому

$$W(j) = \left(m_1 m_2 - \frac{j}{2}(m_2 + 1) \right) W' + (\min\{m_1, m_2\} - j)n + jn + 1 =$$

$$= \left(m_1 m_2 - \frac{j}{2}(\max\{m_1, m_2\} + 1) \right) W' + \min\{m_1, m_2\} n + 1.$$

В алгоритме А1.4 количество сравнений строк, определивших их неравенство, совпадает с аналогичным количеством из алгоритма А1.2. Кроме того, в алгоритме А1.4 $(\min\{m_1, m_2\} - j)$ раз выполняется процедура удаления строки из табли-

цы с меньшим количеством строк. Также необходимо $2jn+1$ вычисление для сравнения равных строк, их удаления из таблицы с большим количеством строк и сравнения m_1 с m_2, поэтому

$$W(j)=\left(m_1m_2-\frac{j}{2}\left(m_1+m_2+\frac{1}{2}-\frac{j}{2}\right)\right)W'+(\min\{m_1,m_2\}-j)n+2jn+1=$$

$$=\left(m_1m_2-\frac{j}{2}\left(m_1+m_2+\frac{1}{2}-\frac{j}{2}\right)\right)W'+\min\{m_1,m_2\}n+jn+1.$$

В алгоритме А2.1 количество сравнений, определивших неравенство строк, совпадает с аналогичным количеством из алгоритма А1.1. Кроме того, в алгоритме А2.1 m_1 раз выполняется процедура добавления строки из таблицы T_1 в новую таблицу T и (m_2-j) раз процедура добавления строки из T_2 в T. Также выполняется jn вычислений для сравнения равных строк, поэтому

$$W(j)=\left(m_1m_2-\frac{j}{2}(m_2+1)\right)W'+m_1n+(m_2-j)n+jn=$$

$$=\left(m_1m_2-\frac{j}{2}(m_2+1)\right)W'+m_1n+m_2n.$$

В алгоритме А2.2 число сравнений, определивших неравенство строк, совпадает с аналогичным числом из алгоритма А1.2. Кроме того, в алгоритме А2.2 m_1 раз выполняется процедура добавления строки из таблицы T_1 в новую таблицу T, (m_2-j) раз — строки из T_2 в T и j раз — удаления строки из T_2. Также выполняется jn вычислений для сравнения равных строк, поэтому имеем

$$W(j)=\left(m_1m_2-\frac{j}{2}\left(m_1+m_2+\frac{1}{2}-\frac{j}{2}\right)\right)W'+m_1n+(m_2-j)n+2jn=$$

$$=\left(m_1m_2-\frac{j}{2}\left(m_1+m_2+\frac{1}{2}-\frac{j}{2}\right)\right)W'+m_1n+m_2n+jn.$$

В алгоритме А2.3 количество сравнений строк, определивших их неравенство, совпадает с аналогичным количеством из алгоритма А1.2. Кроме того, в алгоритме А2.3 $(\min\{m_1,m_2\}-j)$ раз выполняется процедура удаления строки из таблицы с большим количеством строк. Также необходимо $2jn+1$ вычисление для сравнения равных строк, их добавления в таблицу с большим количеством строк и сравнения m_1 с m_2, поэтому

$$W(j)=\left(m_1m_2-\frac{j}{2}\left(m_1+m_2+\frac{1}{2}-\frac{j}{2}\right)\right)W'+(\min\{m_1,m_2\}-j)n+2jn+1=$$

$$=\left(m_1m_2-\frac{j}{2}\left(m_1+m_2+\frac{1}{2}-\frac{j}{2}\right)\right)W'+\min\{m_1,m_2\}n+jn+1.$$

В алгоритме А3.1 количество сравнений строк, определивших их неравенство, совпадает с аналогичным числом из алгоритма А1.1. Кроме того, в алгоритме А3.1 (m_1-j) раз выполняется процедура добавления строки из таблицы T_1 в новую таблицу T. Также необходимо jn вычислений для сравнения равных строк. Поэтому имеем

$$W(j)=\left(m_1m_2-\frac{j}{2}(m_2+1)\right)W'+jn+(m_1-j)n=\left(m_1m_2-\frac{j}{2}(m_2+1)\right)W'+m_1n.$$

В алгоритме А3.2 число сравнений строк, определивших их неравенство, совпадает с аналогичным числом из алгоритма А1.2. Кроме того, в алгоритме А3.2 (m_1-j) раз выполняется процедура добавления строки из таблицы T_1 в новую таблицу T и необходимо $2jn$ вычислений для сравнения равных строк и их удаления из таблицы T_2. Поэтому запишем

$$W(j)=\left(m_1m_2-\frac{j}{2}\left(m_1+m_2+\frac{1}{2}-\frac{j}{2}\right)\right)W'+(m_1-j)n+2jn=$$

$$=\left(m_1m_2-\frac{j}{2}\left(m_1+m_2+\frac{1}{2}-\frac{j}{2}\right)\right)W'+m_1n+jn.$$

В алгоритме А3.3 количество сравнений строк, определивших их неравенство, совпадает с аналогичным количеством из алгоритма А1.1. Также необходимо $2jn$ вычислений для сравнения равных строк и их удаления из таблицы T_1, поэтому имеем

$$W(j)=\left(m_1m_2-\frac{j}{2}(m_2+1)\right)W'+2jn.$$

В алгоритме А3.4 количество сравнений строк, определивших их неравенство, совпадает с аналогичным количеством из алгоритма А1.2. Также необходимо $3jn$ вычислений для сравнения равных строк и их удаления из таблиц T_1 и T_2, поэтому

$$W(j)=\left(m_1m_2-\frac{j}{2}\left(m_1+m_2+\frac{1}{2}-\frac{j}{2}\right)\right)W'+3jn.$$

В алгоритме А3.5 количество сравнений строк, определивших их неравенство, совпадает с аналогичным количеством из алгоритма А1.2 (в данном случае таблицы из алгоритма А1.2 меняются местами). Также необходимо $2jn$ вычислений для сравнения равных строк и их удаления из таблицы T_1, поэтому

$$W(j)=\left(m_1m_2-\frac{j}{2}\left(m_1+m_2+\frac{1}{2}-\frac{j}{2}\right)\right)W'+2jn.$$

4. ОСНОВНЫЕ РЕЗУЛЬТАТЫ

В табл. 1 приведены сложностные характеристики предложенных алгоритмов. Для сложности в среднем указаны только значения $W(j)$, поскольку значения $P(j)$ во всех алгоритмах одинаковы и зависят только от параметров исходных таблиц.

Таким образом, из предложенных алгоритмов операции пересечения наименьшую сложность в худшем случае имеет алгоритм А1.1, а наибольшую — алгоритмы А1.3 и А1.4; наименьшая сложность в среднем у алгоритмов А1.4 и А1.2, а наибольшая — у алгоритмов А1.1 и А1.3. Для операции объединения наименьшая сложность в худшем случае и в среднем у алгоритма А2.3, наибольшая в худшем случае — у алгоритмов А2.1 и А2.2, а наибольшая в среднем — у алгоритма А2.1. Из предложенных алгоритмов разности наименьшая сложность в худшем случае у алгоритмов А3.3 и А3.5, а наибольшая — у алгоритмов А3.1 и А3.2; наименьшая сложность в среднем у алгоритма А3.5, а наибольшая — у алгоритмов А3.1 и А3.3. Таким образом, из предложенных алгоритмов наиболее быстродействующими являются А1.4, А2.3 и А3.5.

Таблица 1. Сложностные характеристики алгоритмов

Алгоритм	Значение сложности в худшем случае	Значение параметра $W(j)$ для сложности в среднем
A1.1	$m_1 m_2 n + n$	$(m_1 m_2 - \frac{j}{2}(m_2 + 1))W' + 2jn$
A1.2	$m_1 m_2 n + 3n$	$\left(m_1 m_2 - \frac{j}{2}\left(m_1 + m_2 + \frac{1}{2} - \frac{j}{2}\right)\right)W' + 3jn$
A1.3	$m_1 m_2 n + \min\{m_1, m_2\}n + 1$	$\left(m_1 m_2 - \frac{j}{2}(\max\{m_1, m_2\} + 1)\right)W' + \min\{m_1, m_2\}n + 1$
A1.4	$m_1 m_2 n + \min\{m_1, m_2\}n + 1$	$\left(m_1 m_2 - \frac{j}{2}\left(m_1 + m_2 + \frac{1}{2} - \frac{j}{2}\right)\right)W' + \min\{m_1, m_2\}n + jn + 1$
A2.1	$m_1 m_2 n + m_1 n + m_2 n$	$\left(m_1 m_2 - \frac{j}{2}(m_2 + 1)\right)W' + m_1 n + m_2 n$
A2.2	$m_1 m_2 n + m_1 n + m_2 n$	$\left(m_1 m_2 - \frac{j}{2}\left(m_1 + m_2 + \frac{1}{2} - \frac{j}{2}\right)\right)W' + m_1 n + m_2 n + jn$
A2.3	$m_1 m_2 n + \min\{m_1, m_2\}n + 1$	$\left(m_1 m_2 - \frac{j}{2}\left(m_1 + m_2 + \frac{1}{2} - \frac{j}{2}\right)\right)W' + \min\{m_1, m_2\}n + jn + 1$
A3.1	$m_1 m_2 n + m_1 n$	$\left(m_1 m_2 - \frac{j}{2}(m_2 + 1)\right)W' + m_1 n$
A3.2	$m_1 m_2 n + m_1 n$	$\left(m_1 m_2 - \frac{j}{2}\left(m_1 + m_2 + \frac{1}{2} - \frac{j}{2}\right)\right)W' + m_1 n + jn$
A3.3	$m_1 m_2 n + n$	$\left(m_1 m_2 - \frac{j}{2}(m_2 + 1)\right)W' + 2jn$
A3.4	$m_1 m_2 n + 3n$	$\left(m_1 m_2 - \frac{j}{2}\left(m_1 + m_2 + \frac{1}{2} - \frac{j}{2}\right)\right)W' + 3jn$
A3.5	$m_1 m_2 n + n$	$\left(m_1 m_2 - \frac{j}{2}\left(m_1 + m_2 + \frac{1}{2} - \frac{j}{2}\right)\right)W' + 2jn$

ЗАКЛЮЧЕНИЕ

Для экспериментального подтверждения теоретических результатов была разработана программная система в среде Lazarus, вычисляющая фактическое количество выполненных действий для каждого из предложенных алгоритмов и сравнивающая их с полученными теоретическими оценками. Система также может вычислять среднее значение фактического количества выполненных действий для любой серии опытов, в каждом из которых исходные таблицы формируются случайным образом. Проведенные эксперименты показали, что уже для относительно небольшого количества опытов (от 100) средние значения фактического количества выполненных вычислений отличаются от расчетных на величину, не превосходящую 0,1%, а при увеличении числа опытов различие между фактическим количеством выполненных вычислений и расчетным уменьшается.

В дальнейшем предполагается распространить полученные результаты на мультитаблицы (таблицы, в которых строки могут повторяться), а также найти оценки сложности других сигнатурных операций табличных алгебр.

СПИСОК ЛИТЕРАТУРЫ

1. Codd E.F. A Relational model of data for large shared data banks. *Communications of the ACM*. 1970. Vol. 13, N 6. P. 377–387.
2. Codd E.F. The relational model for database management: version 2. Reading: Addison-Wesley, 1990. 538 p.

3. Редько В.Н., Буй Д.Б. К основаниям теории реляционных моделей баз данных. *Кибернетика и системный анализ*. 1996. № 4. С. 3–12.
4. Редько В.Н., Брона Ю.И., Буй Д.Б. Реляционные алгебры: операции проекции и соединения. *Кибернетика и системный анализ*. 1997. № 4. С. 89–100.
5. Редько В.Н., Брона Ю.И., Буй Д.Б. Реляционные алгебры: операции деления и переименования. *Кибернетика и системный анализ*. 1997. № 5. С. 3–15.
6. Jarke M., Koch J. Query optimization in database systems. *ACM Computing Surveys*. 1984. Vol. 16, N 2. P. 111–152.
7. Valduriez P. Join Indeces. *ACM Trans. On Database Systems*. 1987. Vol. 12, N 2. P. 218–246.
8. Кузнецов С.Д. Методы оптимизации выполнения запросов в реляционных СУБД. *Итоги науки и техники. Вычислительные науки*. Москва: ВИНИТИ, 1989. Т. 1, № 6. С. 76–153.
9. Мендкович Н.А., Кузнецов С.Д. Обзор развития методов лексической оптимизации запросов. *Тр. ИСП РАН*. 2012. Т. 23, № 6. С. 195–214.
10. Буй Д.Б., Скобелев В.Г. Сложность операций в базах данных (обзор). *Радіоелектронні і комп'ютерні системи*. 2014. № 6. С. 53–59.
11. Редько В.Н., Брона Ю.И., Буй Д.Б., Поляков С.А. Реляційні бази даних: табличні алгебри та SQL-подібні мови. Київ: ВД «Академперіодика», 2001. 198 с.
12. Мейер Д. Теория реляционных баз данных: Пер. с англ. Москва: Мир, 1987. 608 с.

Надійшла до редакції 15.06.2016

В.Н. Редько, Д.Б. Буй, І.С. Канарська, О.С. Сенченко
ТОЧНІ ОЦІНКИ ЧАСОВОЇ СКЛАДНОСТІ РЕАЛІЗАЦІЇ АЛГОРИТМІВ ТЕОРЕТИКО-МНОЖИННИХ ОПЕРАЦІЙ В ТАБЛИЧНИХ АЛГЕБРАХ

Анотація. Досліджено алгоритми, що реалізують операції перетину, об'єднання і різниці в табличних алгебрах. Запропоновано модифікації найбільш поширених алгоритмів, які дозволяють скоротити кількість обчислень. На основі оцінки складності в гіршому випадку і в середньому для модифікованих алгоритмів знайдено найбільш швидкий алгоритм для кожної операції. Розроблено програмну систему, що експериментально підтверджує теоретичні оцінки.

Ключові слова: складність алгоритму, база даних, таблична алгебра.

V.N. Red'ko, D.B. Buy, I.S. Kanarskaya, A.S. Senchenko
PRECISE ESTIMATES OF THE TIME COMPLEXITY OF IMPLEMENTING THE ALGORITHMS OF SET-THEORETIC OPERATIONS IN TABLE ALGEBRA

Abstract. The algorithms implementing intersection, union, and difference in table algebras are investigated. Modifications of the most common algorithms reducing the amount of computation are proposed. Based on the evaluated complexities in the worst case and in the average for the modified algorithms, the fastest algorithm for each operation is found. The program experimentally confirming the theoretical estimates is developed.

Keywords: complexity of algorithms, database, table algebra.

Редько Владимир Никифорович,
академик НАН Украины, доктор физ.-мат. наук, профессор Киевского национального университета имени Тараса Шевченко.

Буй Дмитрий Борисович,
доктор физ.-мат. наук, профессор Киевского национального университета имени Тараса Шевченко, e-mail: dmitriybuy@mail.ru; buy@unicyb.kiev.ua.

Канарская Ирина Сергеевна,
аспирантка Киевского национального университета имени Тараса Шевченко, e-mail: Iren_kiss@mail.ru.

Сенченко Алексей Сергеевич,
кандидат физ.-мат. наук, доцент Киевского национального университета имени Тараса Шевченко, e-mail: senchenko_as@mail.ru.