АЛГОРИТМ ОПТИМИЗАЦИИ АЛЬТЕРНАТИВНЫХ СОЕДИНЕНИЙ ТАБЛИЦ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ В УПРАВЛЕНИИ ОРГАНИЗАЦИИ

Дятчина Д.В., Муравейко А.Ю.
(*Липецкий государственный технический университет, Липецк*)
pak@stu.lipetsk.ru, amur@stu.lipetsk.ru, dd.8383@mail.ru

Объективное оценивание эффективности управления организациями возможно лишь на основе реальной информации, полученной при реализации мероприятий совершенствования производства. Оперативность информации может быть обеспечена с помощью использования автоматизированных информационных систем (АИС), ориентированных на минимизацию времени формирования отчетов и позволяющих пользователю самостоятельно создавать отчеты без участия разработчиков.

Время формирования отчета прямо пропорционально времени выполнения составляющих его запросов. Существующие подходы оптимизации запросов направлены на улучшение запросов, содержащих инвариантную схему соединения таблиц [1]. Однако, в базах данных (БД) сложных структур при динамичном изменении объема таблиц ранее запланированные варианты операций соединения с течением времени могут оказаться не оптимальными в плане скорости их выполнения.

Время выполнения запроса можно представить в виде формулы: $\tau = \sum_{i=1}^{n}([\tau_{от\_i} + \tau_{з\_i} + \frac{V_i * \tau_б}{V_б}] * x_i) + \tau_{соед}$, где x_i=1, если *i*-ая таблица, принадлежит запросу; 0 – иначе; *n*–количество таблиц; $V_б$ – объем блока; V_i – объем і-й таблицы; $\tau_{от\_i}$ – время открытия *i*-й таблицы; $\tau_{з\_i}$ – время закрытия *i*-й таблицы; $\tau_б$ – время чтения блока; $\tau_{соед}$ – общее время выполнения операций соединения.

Для выбора оптимального маршрута соединения таблиц из нескольких семантически альтернативных, представим схему БД

в виде графа, выполнив переход от таблиц к вершинам и от связей к дугам. Каждой вершине графа сопоставим нагрузку e_i – время доступа и чтения таблицы, каждой дуге сопоставим нагрузку p_j – время на соединение инцидентных ей таблиц. Таким образом, для выбора оптимального маршрута соединения необходимо решить задачу оптимизации на графе с нагруженными вершинами и дугами.

Задача оптимизации на графе состоит в выборе минимально нагруженного подграфа при условии, что результирующий подграф является связным:

(1) $$f(G) = \sum_{i=1}^{n} x_i * e_i + \sum_{j=1}^{m} y_j * p_j \to \min,$$

где $e_i = \tau_{от\_i} + \tau_{з\_i} + \tau_б \frac{V_i}{V_б}$, e_i – нагрузка на *i*-ю вершину; x_i = 1, если *i*-ая вершина, принадлежит подграфу, 0 – иначе; *n* – количество вершин; y_j= 1, если *j*-ая дуга принадлежит подграфу, 0 – иначе; *m* – количество дуг; p_j – нагрузка на *j*-ю дугу.

Для задачи (1) существуют методы решения (например [2]), но они ограниченны определенной предметной областью и специфической структурой графа. Поэтому для случая, когда граф имеет произвольную структуру, разработан следующий алгоритм оптимизации на графе:

Шаг 1. Инициализируем граф (структуру, нагрузки, список отмеченных вершин).

Шаг 2. Текущий граф поиска приравниваем введенному графу.

Шаг 3. Если есть несоединенные отмеченные вершины, то переходим к Шагу 4, иначе – Конец алгоритма.

Шаг 4. Инициализируем новый список (дерево) маршрутов поиска.

Шаг 5. Если есть перспективные маршруты поиска, то переходим на Шаг 6, иначе – на Шаг 9.

Шаг 6. Анализируем текущий список маршрутов на достижение искомой цели с пометками дальнейшей перспективности родительского маршрута.

Шаг 7. Создаем новый список маршрутов, длиннее на одну дугу, с анализом перспективности.

Шаг 8. Заносим новый список маршрутов в текущий список маршрутов и переходим к Шагу 5.

Шаг 9. Присоединяем найденный путь соединения 2-х отмеченных вершин (промежуточный результат) к накапливаемому итоговому результату.

Шаг 10. Осуществляем перестройку текущего графа и переходим к Шагу 3.

После присоединения очередного промежуточного результата (ПР) к накапливаемому итоговому результату граф, на котором проводится оптимизация, перестраивается следующим образом:

1) Вершины, образующие ПР, объединяются в одну псевдовершину.

2) Нагрузка на данной вершине устанавливается равной общей нагрузке ПР.

3) Ребра и вершины, составляющие ПР, удаляются. Вместо них создается псевдовершина.

4) Если имеются дуги, соединяющие не принадлежащую ПР вершину с несколькими вершинами, принадлежащими ПР, то остается дуга с минимальной стоимостью, а остальные удаляются.

5) Дуги соединяющие ПР с остальными вершинами заносятся в особый список соответствия, с помощью которого на следующей итерации в общий результат записываются исходные дуги, а не дуги перестроенного графа.

Каждый поиск ПР состоит из итераций, включающих анализ маршрутов текущего уровня и генерацию маршрутов следующего уровня. Итерации продолжаются до исчерпания возможных вариантов маршрутов соединения данной отмеченной вершины с ближайшей отмеченной вершиной.

Рассмотрим пример оптимизации связного графа (рис.1) с нагрузками на вершинах (табл.1) и дугах (табл.2) и отмеченными вершинами: e_1, e_{12} и e_{16}. В ходе работы алгоритма была выполнена одна перестройка графа (рис.2): e_1-p_4-e_6-p_{11}-e_{11}-p_{14}-e_{13}-p_{15}-e_{12} => $e_{17;}$ $p_{18}(e_{13,}e_{15})$ => $p_{21}(e_{17,}e_{15})$, $p_1(e_{1,}e_3)$=>$p_{22}(e_{17,}e_3)$, $p_3(e_{1,}e_5)$

=> $p_{23}(e_{17,}e_5)$, $p_{13}(e_{12,}e_{10})$ => $p_{24}(e_{17,}e_{10})$, $p_{16}(e_{12,}e_{14})$ => $p_{25}(e_{17,}e_{14})$, $p_{12}(e_{12,}e_9)$=> $p_{26}(e_{17,}e_9)$, $p_2(e_{1,}e_4)$ =>$p_{27}(e_{17,}e_4)$

Итерации алгоритма приведены в таблицах 3,4. Результат работы алгоритма - дерево (в данном случае цепь), (рис.1) e_1-p_4-e_6-p_{11}-e_{11}-p_{14}-e_{13}-p_{15}-e_{12}-p_{16}-e_{14}-p_{19}-e_{16} со стоимостью 351 сек.

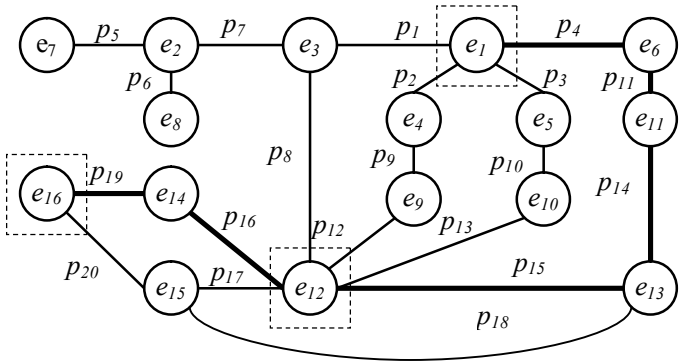

Рис. 1. Оптимизируемый граф. Исходное состояние

Таблица 1. Нагрузки на вершинах

i	1	2	3	4	5	6	7	8
e_i, сек	10	5	50	20	15	5	10	20
i	9	10	11	12	13	14	15	16
e_i, сек	15	10	10	20	5	5	20	30

Таблица 2. Нагрузки на дугах

j	1	2	3	4	5	6	7	8	9	10
p_j, сек	121	61	51	31	31	51	111	141	71	51
j	11	12	13	14	15	16	17	18	19	20
p_j, сек	31	71	61	31	51	51	81	51	71	101

Таблица 3. 1-я итерация алгоритма

№	Уровень	Маршрут	Стоимость	Родитель
	0	1	10	0
Создание 1го уровня				
0	1	1,3	181	0
1	1	1,4	91	0
2	1	1,5	76	0
3	1	1,6	46	0
Создание 2-го уровня				
0	2	1,3,2	297	0
1	2	1,3,**12**	342	0
2	2	1,4,9	177	1
3	2	1,5,10	137	2
4	2	1,6,11	87	3
Создание 3го уровня				
0	3	1,4,9,**12**	268	1
1	3	1,5,10,**12**	218	2
2	3	1,6,11,13	123	3
Создание 4-го уровня				
0	4	1,6,11,13, **12**	194	3
1	4	1,6,11,13, 15	194	3

Создание 5 уровня невозможно

Соединение вершин 1 и 12 завершено

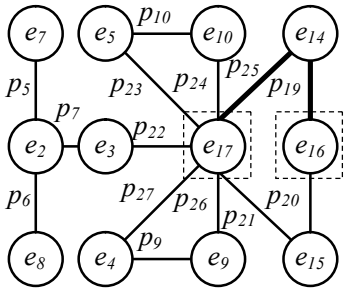

Рис.2. Перестроенный граф

Таблица 4. 2-я итерация алгоритма

№	Уровень	Маршрут	Стоимость	Родитель
На графе 2				
0	0	16	30	0
Создание 1 уровня				
0	1	16,14	106	0
1	1	16,15	151	0
Создание 2 уровня				
0	2	16,14,**17**	351	0
1	2	16,15,**17**	396	1

Создание 3-го уровня невозможно

Соединение вершин 16 и 17 завершено

Все отмеченные вершины соединены. Поиск завершен.

Таблица 5. Альтернативные маршруты

№	Маршруты	Время выполнения, сек
1	1,4,9,12,14,16	445
2	1,4,9,12,15,16	520
3	1,3,12,14,16	519
4	1,3,12,15,16	594
5	**1,6,11,13,12,14,16**	**351**
6	1,6,11,13,12,15,16	396
7	1,5,10,12,14,16	390
8	1,5,10,12,15,16	465
9	1,6,11,13,15,16,14,12	503

Выводы: разработаны методика выбора оптимального маршрута соединения таблиц в БД, имеющих сложную структуру организации данных; алгоритм поиска оптимального маршрута соединения отмеченных вершин на графе, имеющем циклы, с нагруженными вершинами и дугами.

[Работа поддержана грантом РФФИ № 06-07-89150-а]

Литература

1. ГАРСИА-МОЛИНА Г., УЛЬМАН Д., УИДОМ Д. *Системы баз данных. Полный курс.* Пер. с англ.- М.: Издательский дом «Вильямс», 2003 – 1088 с.
2. ПОГОДАЕВ А.К., АННЕНКОВ А.В. *Метод оптимизации графов с нагруженными вершинами* /Вестник ЛГТУ – ЛЕГИ 2001 №1(7) – 37-39с.