ОПТИМИЗАЦИЯ РАСПРЕДЕЛЕННЫХ ХРАНИЛИЩ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ И ОБЪЕКТНОГО МОДЕЛИРОВАНИЯ

Лаздынь С.В, Петров А.В.
Донецкий национальный технический университет,
Кафедра автоматизированных систем управления
e-mail: slazd@ukr.net, petrovolexandr@gmail.com

Abstract

Lazdyn S.V., Petrov A.V. Optimization of distributed data warehouses with using genetic algorithm and object modeling. *In given article applied at present approaches to optimization of the distributed data warehouses are analyzed. The new approach to optimization of the distributed data warehouses based on join application of genetic algorithm and object model of distributed data warehouse is offered. Also the structure of a chromosome for the modified genetic algorithm is presented.*

Характеристика проблемы

Распределенные хранилища данных (РХД) необходимы для хранения и обеспечения быстрого доступа к данным в условиях различных организаций, филиалы которых удалены друг от друга. В современном мире технологии распределенных хранилищ данных нашли широкое применение в информационных системах банков, правоохранительных органов, межрегиональных и международных организаций.

При проектировании и создании РХД одной из главных задач является обеспечение наиболее высокой скорости получения данных пользователями. Это особенно важно ввиду распределенности хранилища данных, так как передача больших объемов данных по сети может значительно увеличить длительность выполнения запросов и время реакции системы.

Краткий анализ проведенных исследований по оптимизации РХД

В настоящее время используется несколько методов оптимизации распределенных хранилищ данных, а именно: жадные алгоритмы на основе линейной модели стоимости и модели сетевой полезности, метод муравьиных колоний, метод кадрирования хранилища данных, оптимизация с помощью AsmL. Рассмотрим основные особенности указанных методов.

При использовании жадного алгоритма для оптимизации распределенных хранилищ данных в качестве условия оптимальности используется линейная модель стоимости данных [1]. Согласно данной модели стоимость ответа на запрос это количество строк во всех таблицах и представлениях, к которым произойдет обращение для ответа на этот запрос. Повышение эффективности достигается за счет создания материализованных представлений для запросов с наибольшей стоимостью.

При использовании модели сетевой полезности [2] для оценки эффективности хранилища данных используется усовершенствованная модель стоимости материализованных представлений. Стоимость материализованного представления складывается из двух составляющих: стоимость времени ответа на запросы – QRB и стоимость поддержки материализованного представления – MCB. Стоимость представления v рассчитывается как: $NB_v = QRB_v + MCB_v$. Оптимальным считается хранилище данных с наименьшей суммарной стоимостью материализованных представлений.

Работа метода жадного алгоритма состоит в том, что оптимальное решение строится постепенно, шаг за шагом, причем на каждом шаге оптимизируется решение только текущего шага, не рассматривая оптимальность конечного результата. Предполагается, что последовательность локально оптимальных решений дает глобально оптимальное решение.

К достоинствам данного метода можно отнести наглядность, простоту реализации. Главным же недостатком является то, что последовательность локально оптимальных решений не обязательно приведет к глобальному оптимуму. При этом, вряд ли удастся достигнуть оптимального решения только с помощью материализации запросов с наибольшей стоимостью. Данный метод легко применим для нераспределенных хранилищ данных. Однако его сложно применить для оптимизации распределенных хранилищ данных, так как при выборе представлений для материализации нужно учитывать не только их размеры, стоимость хранения и т.п., но и их физическое местоположение на одном из узлов хранилища данных. Также существенное влияние на эффективность РХД оказывает изменение его физических параметров, таких как производительность серверов, пропускная способность каналов связи и т.п.

Метод муравьиных колоний [3] основан на моделировании поведения муравьев, способных быстро находить кратчайший путь от муравейника к источнику пищи и адаптировавшись к изменяющимся условиям, вновь находить оптимальный путь. Идея лежащая в основе метода муравьиных колоний – параллельный итеративный поиск в нескольких вычислительных потоках с учетом информации об эффективности результатов полученных на предыдущих итерациях [4].

Для оптимизации с помощью метода муравьиных колоний хранилище данных представляется в виде четырехмерного куба из кортежей:

<Patt(f),Meas(f),Attr(f),R>,

где Patt(f) – множество измерений, Meas(f) – множество показателей (фактов), Attr(f) – множество атрибутов, R – множество функциональных зависимостей $a_i \to a_j$ определенных между парами атрибутов в Attr(f), где $a_i \to a_j$ обозначает два случая, когда a_j однозначно определяет a_i и a_i транзитивно определяет a_j.

Целевая функция метода муравьиных колоний обеспечивает минимизацию стоимости выполнения запросов и задается формулой (1):

$$\min z = \sum_{i \in Q} \sum_{j \in P_i} \sum_{k \in Ti} c_{ijk} x_{ijk}, \quad (1)$$

где z – оптимальное решение, Q – множество запросов, P_i – множество измерений участвующих в запросе Q_i, T_i – множество показателей отбираемых в запросе Qi, c_{ijk} – стоимость выполнения i-го запроса, для j-го измерения и k-го показателя, x_{ijk} – равен либо 1, либо 0. x_{ijk} равен 1 тогда и только тогда, когда запрос i выполнился для j-го измерения и получил показатель k.

Недостатками данного подхода являются: высокая трудоемкость анализа полученных результатов, сильная зависимость от настроечных параметров, которые подбираются только исходя из экспериментов. Также, не смотря на то, что муравьиные алгоритмы гарантируют сходимость метода, время сходимости не определено и может быть сколь угодно велико. Кроме того, при оптимизации не учитывается распределенность хранилища данных. Описанный подход применим в основном для нераспределенных хранилищ.

Метод кадрирования хранилища данных DWS (Data Warehouse Striping) [5] организован на комбинации двух методов: равномерного распределения таблицы фактов между узлами распределенного хранилища данных с репликацией таблиц измерений на каждом из узлов РХД и приближенного выполнения запросов. Метод приближенного выполнения запросов состоит в следующем: в случае выхода из строя одного из узлов РХД результат запроса аппроксимируется на основе доступных данных из других узлов РХД.

Однако, технология DWS применима только к хранилищам данных, содержащим небольшие таблицы измерений. При наличии больших таблиц измерений возрастают требования к объему жестких дисков узлов хранилища данных, так же падает скорость доступа к данным из-за чрезмерного объема таблиц измерений.

Главный недостаток данного метода оптимизации состоит в том, что он применим только для распределенных хранилищ данных с высокоскоростными соединениями между уз-

лами. Для эффективной реализации данного метода необходимы соединения по скорости сравнимые с соединениями в локальных вычислительных сетях. Поскольку записи таблицы фактов распределены равномерно между узлами, то любой запрос будет требовать обращения к большинству, а чаще всего – ко всем узлам РХД, что потребует значительных временных затрат на транспортировку данных.

Также существует подход к оптимизации РХД с помощью языка моделирования AsmL [6]. Язык AsmL представляет математический аппарат для моделирования компьютерных систем. РХД представляется в виде конечного автомата. При моделировании РХД с помощью AsmL такие его компоненты как таблицы фактов, измерений, материализованные представления представляются в виде состояний системы. Запросы с вертикальной и горизонтальной фрагментацией представляются переходами системы.

Целевая функция задачи оптимизации выглядит следующим образом:

$$\min q \cos t = \sum_{q} f_q \cdot \sum_{f \in F_q} s(f), \qquad (2)$$

где f_q частота обращения запроса q к фрагменту f , s(f) – размер фрагмента f, F_q – множество фрагментов используемых запросом [6].

Модель, используемая при оптимизации, достаточно полно описывает основные компоненты логической архитектуры РХД, такие как измерения, таблицы фактов, материализованные представления, запросы. Вместе с тем в модели не отражены компоненты физической архитектуры: сервера, каналы связи, сетевые устройства и т.п. Кроме того, используемый критерий оптимальности распределения данных между узлами хранилища данных зависит только от объема, избыточности данных и частоты обращения к фрагментам. В данном критерии не учитывается скорость получения пользователями ответов на запросы, что является главным показателем эффективности хранилища данных.

Рассмотренные методы оптимизации РХД, кроме метода оптимизации с использованием AsmL, не учитывают влияние на эффективность технических характеристик РХД. Также эти методы не учитывают распределение данных между узлами РХД.

Постановка задачи оптимизации РХД

Распределенные хранилища данных создаются для обеспечения аналитикам быстрого доступа к распределенным данным. Поэтому оптимальным мы будем считать такое РХД, в котором среднее время выполнения пользовательских запросов на выборку было бы минимальным.

В качестве критерия эффективности РХД выбрано среднее время выполнения пользовательских запросов на выборку данных. В ходе выполнения запроса часть данных может быть получена из материализованных представлений, часть данных может быть выбрана непосредственно из таблиц фактов. Кроме того, в случае, если данные выбираются из удаленных от источника узлов хранилища данных, потребуется передача результатов запроса по сети. Таким образом, время выполнения одного запроса складывается из временных затрат на выборку данных из фрагментов таблицы фактов, из материализованных представлений и временных затрат на передачу данных.

Пусть задано не пустое множество фрагментов таблицы фактов F и множество материализованных представлений M. Для заданной структуры компонентов физической архитектуры РХД необходимо найти такое размещение фрагментов таблицы фактов и материализованных представлений по узлам РХД, при котором среднее время выполнения запросов пользователей было бы минимальным.

Среднее время выполнения запросов пользователей в распределенном хранилище данных рассчитывается как сумма произведений времен выполнения запросов и частот выполнения запросов деленная на количество запросов:

$$\min \quad \overline{T_q} = \frac{\sum_{i=1}^{N}\left(\sum_{j=1}^{M_i} t_j + \sum_{k=1}^{F_i} t_k + \sum_{l=1}^{N_i} t_l\right)\cdot q_i}{N}, \qquad (3)$$

где $\overline{T_q}$ – среднее время выполнения запросов пользователей к таблицам хранилища данных, N – количество пользовательских запросов на выборку данных, M_i – количество обращений к материализованным представлениям для выполнения i-го запроса, F_i – количество обращений к фрагментам таблицы фактов для выполнения i-го запроса, N_i – количество передач данных по сети для выполнения i-го запроса, t_j – время выборки данных из материализованного представления, t_k – время выборки данных из таблицы фактов, t_l – время передачи данных по сети источнику запроса, q_i – частота возникновения запроса.

Данный критерий учитывает такие важные особенности распределенного хранилища данных как репликация данных на различных узлах РХД и степень материализации данных. Чем больше возможных запросов будет реализовано с помощью материализованных представлений, тем меньше будет обращений непосредственно к таблицам фактов, что сократит среднее время выполнения запросов. Это обусловлено тем, что время выборки данных из материализованных представлений существенно меньше времени выборки данных из таблицы фактов, так как материализованные представления содержат только часть записей из таблицы фактов. Наилучшим было бы такое решение, при котором вообще не происходило бы обращений к таблице фактов, однако это потребовало бы больших затрат на хранение большого количества материализованных представлений. Однако, такое решение не является приемлемым, так как в этом случае на многих узлах хранилища данных придется размещать большие объемы редко используемых данных, что повлечет за собой излишние затраты на хранение данных и переполнение устройств хранения данных. Помимо этого, возникнет необходимость в хранении и синхронизации на всех узлах РХД больших объемов информации. При этом возрастает время, затрачиваемое на репликацию данных.

Ограничения в данной задаче должны учитывать возможности РХД по размещению и обновлению данных на множестве узлов компьютерной системы. К ним относятся следующие ограничения: на время синхронизации данных, на объем размещаемых данных на узлах хранилища данных и на размещение, хотя бы одного экземпляра таблицы фактов в РХД. Рассмотрим подробнее эти ограничения.

Ограничение на среднее время синхронизации данных. При наличии большого количества материализованных представлений и репликаций данных процессы синхронизации могут на длительные промежутки времени занимать каналы связи и серверы, что в свою очередь приведет к задержкам выполнения пользовательских запросов к хранилищу. Поэтому необходимо ограничить дублирование данных и создание материализованных представлений, таким образом, чтобы среднее время синхронизации не превышало предельно допустимой величины.

Среднее время синхронизации данных должно быть меньше либо равно заложенному при проектировании граничному значению времени синхронизации:

$$\overline{T_s} = \sum_{i=1}^{N} t_{S_i} \Big/ N \le T_{SL}, \qquad (4)$$

где $\overline{T_s}$ – среднее время синхронизации данных, t_{S_i} – время синхронизации при добавлении одной записи в i-й фрагмент данных, N – общее количество фрагментов таблицы фактов, T_{SL} – предельное значение среднего времени синхронизации.

Время синхронизации при добавлении/модификации одной записи во фрагмент таблицы фактов можно выразить в виде максимальной величины из двух значений. Первое значение будет представлять собой максимальную величину из всех сумм времен передачи по каналам

связи и времен добавления данных в фрагмент. Второе значение будет представлять собой максимальную величину из всех сумм времен передачи по каналам связи и времен обновления материализованных представлений.

$$t_{Si} = MAX\left(MAX(t_{Nl} + t_{Al}), MAX(t_{N_j} + t_{U_j})\right) l \in \overline{1,m}, j \in \overline{1,k}, \quad (5)$$

где t_{N_j} – время передачи данных по сети от узла, содержащего i-й фрагмент таблицы фактов, к узлу, содержащему его j-ю копию или к узлу содержащему j-е материализованное представление, t_{A_j} – время добавления/модификации одной записи в j-й фрагмент таблицы фактов, t_{U_j} – время обновления j-го материализованного представления, m – количество требующих обновления фрагментов таблицы фактов, k – количество требующих обновления материализованных представлений.

Ограничение на объемы размещаемых данных. В ходе оптимизации может возникнуть необходимость разместить на одном из узлов РХД дополнительные данные. Однако свободного места на жестких дисках данного узла хранилища данных может быть недостаточно для размещения необходимого объема информации. В этом случае следует либо отказаться от размещения на данном узле дополнительной информации, либо увеличить емкость запоминающих устройств.

Это ограничение можно выразить следующим образом:

$$\sum_{i=1}^{N} S_{ik} < \sum_{j=1}^{M} D_{jk} \forall k, \quad (6)$$

где S_{ik} – размер i-го фрагмента данных (фрагментов таблиц фактов или материализованных представлений) на k-м узле РХД, D_{jk} – размер j-го устройства хранения данных на k-м узле РХД, Мб, N – количество фрагментов данных на k-м узле РХД, M – количество устройств хранения данных на k-м узле РХД, k – номер узла РХД.

Ограничение на размещение хотя бы одного экземпляра таблицы фактов. Распределенное хранилище данных должно содержать хотя бы один экземпляр таблицы фактов. Если это условие не будет выполняться, то хранилище данных будет неполным. Каждый фрагмент таблицы фактов должен содержаться в РХД хотя бы по одному разу.

Данное ограничение можно представить в следующем виде:

$$\sum_{i=1}^{C} F_{ij} \geq 1, j \in \overline{1,N} \quad (7)$$

где F_{ij} – признак наличия в j-м узле хранилища i-го фрагмента таблицы фактов (0 или 1), C – количество узлов в хранилще, N – количество фрагментов таблицы фактов.

Предлагаемый подход к оптимизации РХД.

Большая вычислительная сложность и трудоемкость задачи оптимизации РХД не позволяет использовать для её решения классические методы оптимизации. Поэтому в качестве нового подхода к оптимизации распределенных хранилищ данных предложено использовать генетические алгоритмы (ГА) совместно с объектной моделью РХД. Разработанная объектная модель РХД описывает его важнейшие характеристики и позволяет с приемлемой достоверностью производить моделирование его работы [7].

Суть предлагаемого подхода состоит в следующем: структура и параметры распределенного хранилища данных кодируются в виде хромосомы. В хромосоме содержится информация о размещении материализованных представлений и фрагментов таблицы фактов на узлах РХД. Начальный набор хромосом генерируется случайным образом. В процессе работы ГА генерируются новые хромосомы, являющиеся вариантами структуры РХД. С помощью объектной модели РХД производится вычисление значения критерия эффективности для каждой

из хромосом, который будет представлять собой значение фитнесс-функции ГА. Генетический алгоритм на основе полученного значения фитнесс-функции с использованием операторов отбора, скрещивания и мутации формирует новую хромосому. Схема взаимодействия объектной модели РХД с генетическим алгоритмом представлена на рисунке 1.

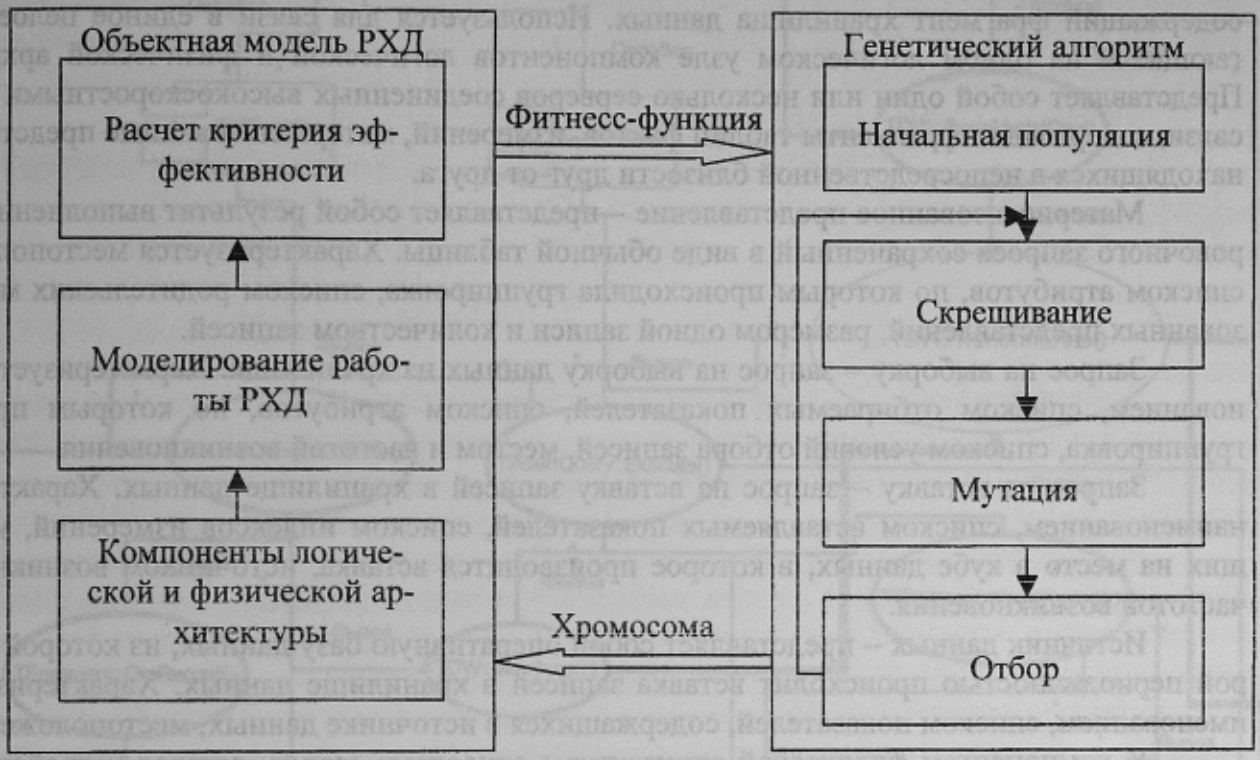

Рисунок 1 - Схема взаимодействия объектной модели РХД с ГА

Работа генетического алгоритма завершится тогда, когда в нескольких популяциях подряд не будет происходить улучшения значения максимума фитнесс-функции. Решение, полученное в ходе работы генетического алгоритма, будет приближенным (квазиоптимальным).

Краткое описание модели РХД

Для моделирования распределенных хранилищ данных и получения их экспериментальных характеристик была разработана объектная модель РХД [7].

В распределенном хранилище данных можно выделить две большие группы компонентов: компоненты логической архитектуры и компоненты физической архитектуры.

К компонентам логической архитектуры относятся: узел распределенного хранилища данных, измерение, атрибут измерения, показатель, фрагмент таблицы фактов, фрагмент таблицы измерения, материализованное представление, запрос на выборку, запрос на вставку, источник данных.

Измерение – представляет собой измерение общего куба данных. Характеризуется наименованием, атрибутами измерения и границами измерения.

Атрибут измерения – описывает вспомогательные поля таблицы измерения. Атрибуты могут находиться в иерархической зависимости, что увеличивает возможности агрегирования данных с различной степенью детализации. Характеризуется наименованием

Показатель – описывает информационное поле таблицы фактов. Характеризуется наименованием, типом данных и списком допустимых способов агрегации данного показателя.

Фрагмент таблицы фактов – представляет собой часть глобальной таблицы фактов горизонтально и/или вертикально фрагментированной. Характеризуется совокупностью индексных полей для связи с измерениями, совокупностью показателей, местоположением, границами, количеством записей и размером записи.

Фрагмент таблицы измерения – представляет собой часть горизонтально фрагментированной одной из глобальных таблицы измерения. Характеризуется местоположением, границами, размером записи и количеством записей.

Узел распределенного хранилища данных – единый программно-аппаратный комплекс, содержащий фрагмент хранилища данных. Используется для связи в единое целое располагающихся на одном логическом узле компонентов логической и физической архитектуры. Представляет собой один или несколько серверов соединенных высокоскоростными каналами связи, содержащих фрагменты таблиц фактов, измерений, материализованные представления и находящихся в непосредственной близости друг от друга.

Материализованное представление – представляет собой результат выполнения группировочного запроса сохраненный в виде обычной таблицы. Характеризуется местоположением, списком атрибутов, по которым происходила группировка, списком родительских материализованных представлений, размером одной записи и количеством записей.

Запрос на выборку – запрос на выборку данных из хранилища. Характеризуется наименованием, списком отбираемых показателей, списком атрибутов, по которым происходит группировка, списком условий отбора записей, местом и частотой возникновения.

Запрос на вставку – запрос на вставку записей в хранилище данных. Характеризуется наименованием, списком вставляемых показателей, списком индексов измерений, указывающих на место в кубе данных, в которое производится вставка, источником возникновения и частотой возникновения.

Источник данных – представляет собой оперативную базу данных, из которой с некоторой периодичностью происходит вставка записей в хранилище данных. Характеризуется наименованием, списком показателей, содержащихся в источнике данных, местоположением.

К компонентам физической архитектуры относятся: сервер, сетевое устройство, канал связи, рабочая станция.

Сервер – сервер, хранящий фрагменты таблицы фактов и/или фрагменты таблиц измерений, материализованные представления, источники данных. Характеризуется наименованием, размером жесткого диска, количеством транзакций выполняемых в секунду, временем освобождения, стоимостью поддержки, сроком эксплуатации.

Сетевое устройство – устройство, управляющее передачей данных. Может представлять собой маршрутизатор, хаб, мост. Характеризуется списком подключенных серверов, скоростью передачи данных, размером пакета и размером заголовка пакета данных.

Канал связи – физическая среда передачи данных, может представлять собой коммутируемый канал связи, выделенную телефонную линию, оптоволоконный кабель, беспроводной канал связи и др. Характеризуется пропускной способностью, помехоустойчивостью, отказоустойчивостью, временем освобождения, а также двумя серверами, находящимися на разных концах канала.

Рабочая станция – пользовательская рабочая станция. Служит источником формирования запросов. Характеризуется наименованием и сетевым устройством, к которому подключена рабочая станция.

Таким образом, выделены типовые компоненты распределенного хранилища данных, среди которых 10 компонентов логической архитектуры и 4 компонента физической архитектуры. Для указанных выше типовых компонентов распределенного хранилища данных были разработаны соответствующие объектные модели, описаны их свойства и методы. Объектные модели разрабатывались в виде классов на языке C++.

Общая модель распределенного хранилища данных была построена как система взаимодействующих объектных моделей типовых компонентов. Схема взаимодействия объектных моделей типовых компонентов представлена на рисунке 2.

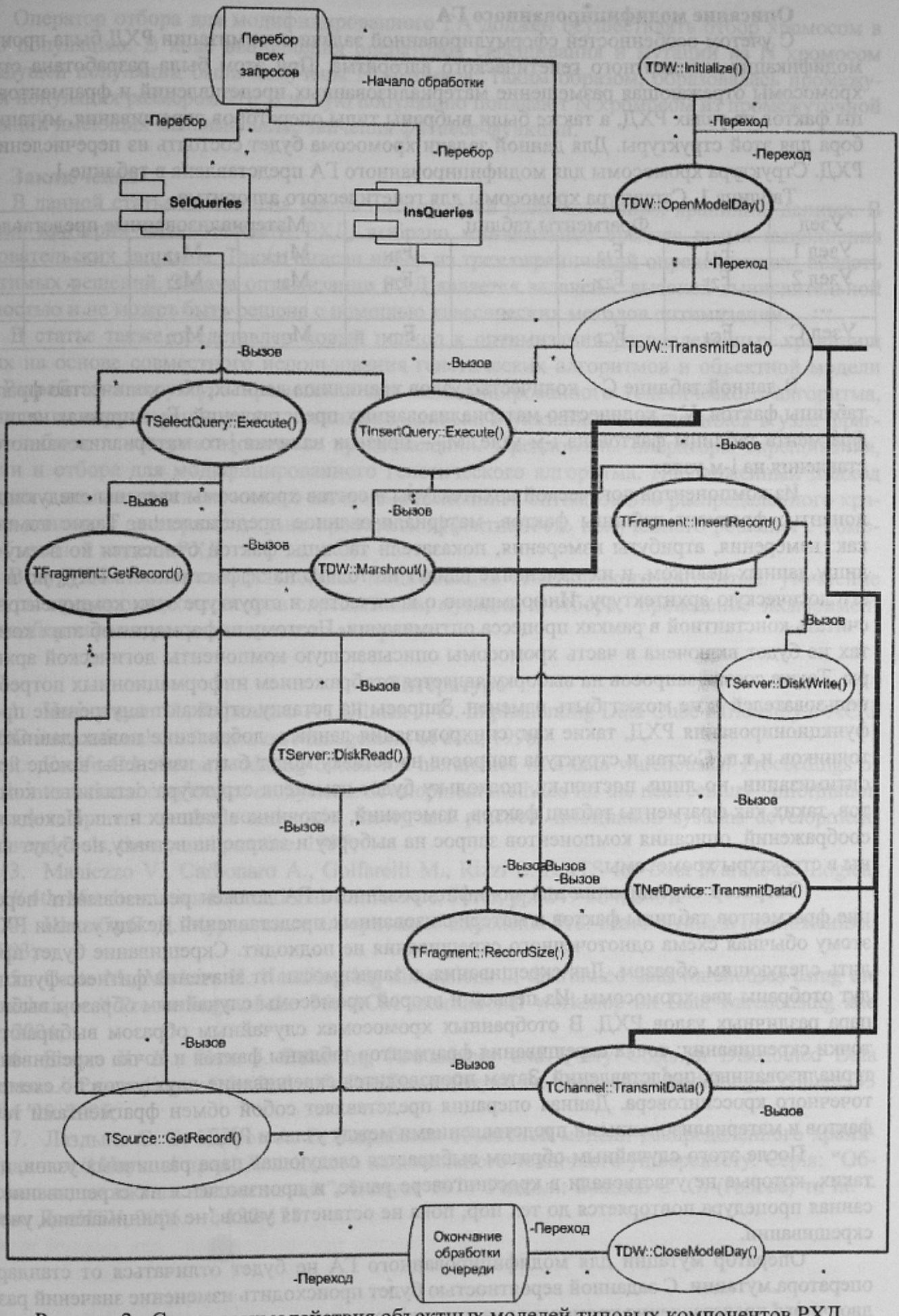

Рисунок 2 - Схема взаимодействия объектных моделей типовых компонентов РХД

Описание модифицированного ГА

С учетом особенностей сформулированной задачи оптимизации РХД была произведена модификация стандартного генетического алгоритма. При этом была разработана структура хромосомы отражающая размещение материализованных представлений и фрагментов таблицы фактов на узлах РХД, а также были выбраны типы операторов скрещивания, мутации и отбора для этой структуры. Для данной задачи хромосома будет состоять из перечисления узлов РХД. Структура хромосомы для модифицированного ГА представлена в таблице 1.

Таблица 1. Структура хромосомы для генетического алгоритма.

Узел	Фрагменты таблиц				Материализованные представления			
Узел 1	F_{11}	F_{12}	…	F_{1N}	M_{11}	M_{12}	…	M_{1V}
Узел 2	F_{21}	F_{22}	…	F_{2N}	M_{21}	M_{22}	…	M_{2V}
…	…	…	…	…	…	…	…	…
Узел C	F_{C1}	F_{C2}	…	F_{CN}	M_{C1}	M_{C2}	…	M_{CV}

В данной таблице C – количество узлов хранилища данных, N – количество фрагментов таблицы фактов, V – количество материализованных представлений, F_{ij} – признак наличия j-го фрагмента таблицы фактов на i-м узле, M_{ij} – признак наличия j-го материализованного представления на i-м узле.

Из компонентов логической архитектуры в состав хромосомы введены следующие компоненты: фрагмент таблицы фактов, материализованное представление. Такие компоненты как: измерения, атрибуты измерения, показатели таблицы фактов относятся ко всему хранилищу данных целиком, и их изменение влияет не только на эффективность РХД, но и на всю его логическую архитектуру. Информацию о количестве и структуре этих компонентов будем считать константной в рамках процесса оптимизации. Поэтому информация об этих компонентах не будет включена в часть хромосомы описывающую компоненты логической архитектуры. Также состав запросов на выборку является отображением информационных потребностей пользователей и не может быть изменен. Запросы на вставку отражают внутренние процессы функционирования РХД, такие как: синхронизация данных, добавление новых данных из источников и т.п. Состав и структура запросов на вставку могут быть изменены в ходе процесса оптимизации, но лишь постольку, поскольку будет изменена структура остальных компонентов, таких как фрагменты таблиц фактов, измерений, источников данных и т.п. Исходя из этих соображений, описания компонентов запрос на выборку и запрос на вставку не будут включены в структуры хромосомы.

Оператор скрещивания для модифицированного ГА должен реализовывать перемещение фрагментов таблицы фактов и материализованных представлений между узлами РХД. Поэтому обычная схема одноточечного скрещивания не подходит. Скрещивание будет происходить следующим образом. Для скрещивания, в зависимости от значения фитнесс-функции будут отобраны две хромосомы. Из первой и второй хромосомы случайным образом выбирается пара различных узлов РХД. В отобранных хромосомах случайным образом выбираются две точки скрещивания: точка скрещивания фрагментов таблицы фактов и точка скрещивания материализованных представлений. Затем производится скрещивание двух узлов по схеме двухточечного кроссинговера. Данная операция представляет собой обмен фрагментами таблицы фактов и материализованными представлениями между узлами РХД.

После этого случайным образом выбирается следующая пара различных узлов, причем таких, которые не участвовали в кроссинговере ранее, и производится их скрещивание. Описанная процедура повторяется до тех пор, пока не останется узлов, не принимавших участия в скрещивании.

Оператор мутации для модифицированного ГА не будет отличаться от стандартного оператора мутации. С заданной вероятностью будет происходить изменение значений разрядов двоичного представления хромосомы на противоположные.

Оператор отбора для модифицированного ГА должен осуществлять отбор хромосом в новую популяцию. В ходе выполнения операторов скрещивания и мутации из N хромосом предыдущей популяции были получены N потомков. Таким образом, образовалась промежуточная популяция размером 2N. В новую популяцию попадают N хромосом из промежуточной популяции имеющих максимальные значения фитнесс-функции.

Заключение

В данной статье поставлена задача оптимизации распределенных хранилищ данных. В качестве критерия оптимальности РХД, выбрано минимальное среднее время выполнения пользовательских запросов. Также описан набор из трех ограничений определяющих область допустимых решений. Задача оптимизации РХД является задачей с высокой вычислительной сложностью и не может быть решена с помощью классических методов оптимизации.

В статье также представлен новый подход к оптимизации распределенных хранилищ данных на основе совместного использования генетических алгоритмов и объектной модели РХД. Разработана структура хромосомы для модифицированного генетического алгоритма, состоящая из перечисления узлов хранилища данных, с указанием находящихся в узле фрагментов таблиц и материализованных представлений. Предложены операторы скрещивания, мутации и отбора для модифицированного генетического алгоритма. Предложенный подход позволяет с помощью генетического алгоритма выполнить оптимизацию распределенного хранилища данных и экспериментально проверить эффективность полученного решения с помощью объектной модели РХД.

В будущем возможна дальнейшая модификация генетического алгоритма: уточнение структуры хромосомы, операторов скрещивания, мутации и отбора, проведение экспериментов и подбор параметров генетического алгоритма.

Литература

1. Harinarayan V., Rajaraman A., Ullman J. D. Implementing Data Cube Efficiently.// ACM SIGMOD international conference on Management of data, 1996;

2. Ezeife C.I. Accommodating dimension hierarchies in a data warehouse.// Proceedings of the sixth international conference on Systems development methods for the next century : information systems development: methods and tools, theory and practice: information systems development: methods and tools, theory and practice. 1997;

3. Maniezzo V., Carbonaro A., Golfarelli M., Rizzi S. ANTS for Data Warehouse Logical Design.// 4th-Metaheuristics-International-Conference, Porto, pp. 249-254, 2001.

4. Штовба С. Д. Муравьиные алгоритмы.// Exponenta Pro. Математика в приложениях, №4, 2003;

5. Costa M., Madeira H. Handling big dimensions in distributed data warehouses using the DWS technique.// Proceedings of the 7th ACM international workshop on Data warehousing and OLAP, 2004;

6. Shewe K. D., Zhao J. Balancing Redundancy and Query Costs in Distributed Data Warehouses.// Proceedings of the 2nd Asia-Pacific conference on Conceptual modelling - Volume 43 APCCM '05. 2005;

7. Лаздынь С. В, Петров А. В. Разработка объектной модели распределенного хранилища данных.// Наукові праці Донецького національного технічного університету. Серія: "Обчислювальна техніка та автоматизація". Випуск 107 / Редкол.: Башков Є. О. (голова) та ін. – Донецьк: ДонНТУ, 2006. – с.122-128.