УДК 004.582

МОДЕЛИ НАГРУЗОЧНОГО ТЕСТИРОВАНИЯ СИСТЕМ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

А.А. Блажко, А.Ю. Левченко, А.С. Пригожев

Одесский национальный политехнический университет
65044, Одесса, пр. Шевченко 1
тел. 8 (048) 779-75-66, 8 (048) 779-71-06
bblack@ieee.org, aleksandra.levchenko@gmail.com, prigozhev@matrix.odessa.ua

Модели нагрузочного тестирования СУБД. Рассматриваются вопросы, посвящённые построению моделей нагрузочного тестирования СУБД корпоративных информационных систем (КИС) для автоматизированной настройки конфигурационных параметров СУБД. В качестве модели, представляющей распределение запросов, предлагается использование марковских моделей с интеграцией в них структуры запроса. Предложена архитектура адаптивной экспертной системы настройки параметров СУБД.

The Models of DBMS benchmark. The article represents the DBMS benchmark models for corporate information systems implementation for automated DBMS parameters configuration. It is offered to use markov models with query structure integration as the query distribution model. The adaptive expert system architecture for DBMS parameters configuration is offered.

Введение

Обеспечение высокого уровня производительности корпоративных информационных систем (КИС), построенных на базе систем управления базами данных (СУБД), - это сложная, но приоритетная задача, которая требует профессионального подхода к реализации. Современные СУБД становятся все более сложными системами, что повышает сложность процесса управления ими со стороны системных администраторов. В то же время остается актуальным удовлетворение требований по временным характеристикам работы программ КИС, работающих с базами данных (БД): сокращение времени выполнения отдельных запросов к БД, увеличение пропускной способности по обработке запросов (количество запросов, обработанных в единицу времени).

Среди методов повышения производительности КИС на базе СУБД отметим настройку параметров конфигурации. Исследование производительности КИС на базе СУБД предполагает использования нагрузочного тестирования с целью получения времени отклика для операций на разных нагрузках в довольно широких диапазонах изменения конфигурационных параметров СУБД. Это позволит подобрать соответствующую заданной КИС конфигурацию параметров СУБД.

Существующие решения в области тестирования производительности СУБД

Тестирование может выполняться с использованием стандартных тестов или специализированных. Стандартные тесты оценки производительности БД КИС используют заранее заданный набор транзакций и их описаний. Любой тест определяется: структурой БД, транзакциями, выполняющими доступ к БД. Стандартные тесты эффективны, если доказана адекватность их структуры структуре БД КИС, а описание транзакций адекватно описанию запросов, выполняемых программами КИС. Иначе необходимо создавать специализированные тесты. В настоящее время существуют следующие стандартные тесты: тесты семейства TPC (A,B,C,D,E,H,W); ANSI SQL Standard Scalable and Portable benchmark (AS3AP); тест Wisconsin Benchmark; Set Query Benchmark тест; Engineering Database Performance тест [1–6]. Рассмотрим подробнее каждый из указанных тестов.

С целью анализа и сравнения вычислительных систем между собой в 1988 г. был создан Совет по оценке производительности обработки транзакций (TPC - Transaction Processing Performance Council) [1, 2]. Основной задачей организации является точное определение тестовых пакетов для оценки систем обработки транзакций и баз данных. Методики тестирования TPC пользуются наибольшей популярностью и включают следующие тесты оценки производительности: TPC-A, TPC-B, TPC-C, TPC-D, TPC-E, TPC-H и TPC-W. Хотя тесты TPC не представляют собой тесты для непосредственной оценки производительности баз данных, системы реляционных баз данных являются ключевыми компонентами любой системы обработки транзакций.

Тестовая система TPC-C является развитием методик TPC-A и TPC-B, которые были отменены в 1995 г. Она имитирует более сложную, чем в предыдущих тестах, схему обработки заказов, а его результаты, следовательно, наиболее приемлемы для получения объективных оценок. База данных в тесте TPC-C состоит из девяти типов таблиц большого размера, содержащих широкий диапазон значений. Результаты теста даются в числе транзакций в такой системе в минуту (tpmC). Соотношение цена/производительность измеряется как стоимость одной транзакции в данном тесте. Каждый принятый TPC результат содержит подробную информацию об использованном оборудовании, программном обеспечении (ПО) и стоимости владения. Если необходимо, можно пересчитать этот параметр, используя свои данные с учетом скидок, предоставляемых именно вашей компании производителями, и получить собственные данные. С точки зрения бизнес-процессов TPC-C симулирует компьютерную среду, в которой множество пользователей выполняют транзакции с использованием данных, хранящихся в СУБД. Подобная бизнес-активность характерна для систем ввода заказов, не зависимо

от области деятельности, будь-то система управления предприятием или заказ авиабилетов. Этот тест включает симуляцию ввода заказов, выдачи накладных, внесения информации о платежах, проверки состояния заказов и отслеживания состояния склада. Нагрузка на систему и структура базы данных выбраны таким образом, чтобы они не несли в себе специфики какой-то конкретной отрасли, но отражали усредненный бизнес, связанный с управлением, продажами или дистрибуцией неких товаров или услуг.

Если тесты TPC-A, B и C моделируют системы OLTP, то тест TPC-D ориентирован уже на несколько другой круг задач, а именно: системы принятия решений или DSS – Decision Support System. Эти системы характеризуются работой с более сложными запросами, возможностью моделирования хода выполнения транзакций для анализа возникающих ситуаций и т.д. В этом тесте используются 17 аналитических сложных запросов, которые могут использоваться широкопрофильным поставщиком при расчете цен и скидок, минимальных требований заказчика, общего анализа и прогнозирования рынка и управления поставками.

Тест TPC-E использует базу данных для моделирования работы брокерской фирмы с клиентами, которые генерируют транзакции, связанные со сделками, запросами по счетам и рыночными исследованиями. Брокерская фирма в свою очередь взаимодействует с финансовыми рынками, осуществляя операции от имени и по поручению своих клиентов, и обновляет учетную информацию. Мерой производительности в тесте TPC-E служит количество транзакций в секунду (tpsE). Хотя в основе бизнес-модели TPC-E заложена брокерская фирма, но схема базы данных, совокупность данных, транзакции и правила исполнения разработаны таким образом, чтобы широко представлять современные OLTP-системы.

Тест TPC-H оценивает производительность систем поддержки принятия решений (СППР). Он состоит из набора сложных, бизнес-ориентированных запросов, которые нельзя было предусмотреть заранее. Данные в таблицах и запросы подобраны так, чтобы отражать некоторую усредненную по индустрии бизнес-активность. Типичные запросы составлены так, чтобы соответствовать основным типам запросов в СППР: ценообразование и скидки, управление прибылью, исследование предпочтений покупателей, исследования рынка и т. п. Результат измеряется в запросах в час (QphH). Важное отличие TPC-H от его предшественника, TPC-D, – возможность продолжать обработку транзакций с тестовой базой данных. Таким образом, выполнение запросов TPC-H не требует отключения СУБД от других операций.

Тесты TPC-W определяют производительность транзакций системы для задач электронной коммерции. В этом тесте нагрузка ложится не только на сервер базы данных, но и на Web-сервер, кэш-сервер, систему хранения изображений и СУБД. В тесте симулируются нагрузки, для которых характерно следующее: множественные соединения с клиентами (браузерами); создание, обновление и доступ к динамическим Web-страницам; использование consistent Web-объектов; одновременное исполнение множества разнородных транзакций; OLTP-модель; сложная структура СУБД; целостность транзакций; конкурентный доступ к данным и изменение данных. Тесты TPC-W – одни из самых комплексных, проводимых TPC. В этом тесте симулируется доступ к 14 типичным страницам в типичных Интернет-задачах (базовая страница, поиск, новые продукты, лучшие покупки и т. п.). При этом выбор страниц осуществляется не по случайному механизму, а в соответствии с элементами навигации на них. Поведение покупателя тоже воссоздается в соответствии с типичными моделями поведения, полученными в реальных системах и усредненными.

Тест Wisconsin Benchmark применяется для измерения производительности при выполнении реляционных запросов к системам баз данных [3]. Тест включает три таблицы, одна с 1000 записями, названная ONEKTUP, а две другие – с 10,000 записями, названными TENKTUP1 и TENKTUP2. Каждая таблица состоит из 13 целочисленных атрибутов и трех 52-байтовых строковых атрибутов.

Тест ANSI SQL Standard Scalable and Portable (AS3AP) применяется для измерения производительности в смешанной среде обработки транзакций, реляционных запросов и выполнения служебных функций [4]. Тест содержит пять таблиц. Одна, содержащая одну запись и одно поле в ней, необходима для измерения загрузки. Другие четыре таблицы сгенерированы генератором БД с соответствующим масштабированием: uniques – таблица, в котрой все атрибуты содержат уникальные значения; Hundred – таблица, в которой большинство атрибутов имеют точно 100 уникальных значений и коррелируемы; Tenpct – таблица, в которой большинство атрибутов имеют 10% уникальных значений; Updates – таблица, используемая для операций модификации. Атрибуты таблиц характеризуются общими типами данных: знаковое и беззнаковое целое, вещественное, числовое, строки с фиксированной и переменной длиной. Диапазон значений атрибута определяется его типом и размером таблицы. Для некоторых численных типов включаются два типа диапазона.

The Set Query benchmark тест также предназначен для измерения производительности системы при выполнении реляционных запросов [5]. При этом обрабатываемые запросы являются гораздо более сложными, чем у двух предыдущих тестов, и встречаются, в основном, в системах поддержки принятия решений. Тест содержит четыре характеристики: портируемость, функциональный охват, селективный охват, масштабируемость. Чтобы выбрать желаемое значение селективности таблица BENCH содержит 13 индексируемых атрибутов. Из них 12 не упорядочены (случайно сгенерированы), а значение кардинальности (количество уникальных значений) находится в диапазоне от 2 до 500,000. Каждый такой атрибут имеет целое значение в диапазоне от 1 до кардинальности, которая соответствует имени атрибута.

Engineering Database Performance тест используется для тестирования баз данных, специализирующихся на хранении технических данных, так называемых CASE и CAD приложений [6]. Тест является независимым от модели данных в СУБД.

Постановка задачи исследования

Существующие синтетические тесты обладают одним существенным недостатком. Как правило, они моделируют поведение синтетической корпоративной информационной системы с заранее заданным количеством запросов. Кроме того, данные тесты содержат значительно меньшее количество таблиц в базе данных, а количество данных в таблице не полностью отражает реальное наполнение таблицы БД КИС. Поэтому актуальным представляется построение специализированных тестов СУБД, позволяющих не только сравнивать по производительности различные СУБД, но и вырабатывать определённые рекомендации по их оптимизации. Основным параметром оценки производительности СУБД со стороны пользователей является время получения ответа данной СУБД. В то же время для тестирования производительности СУБД перспективным представляется использование операционных профилей БД КИС и марковских моделей. Использование данного инструментария позволит построить более точные тесты производительности, чем синтетические, так как будет основано на моделях реального поведения пользователей КИС.

Цель настоящего исследования – построение архитектуры такой системы, а также моделей, используемых в процессе нагрузочного тестирования СУБД.

Марковская модель нагрузочного тестирования на основе запросов КИС

Основными факторами, влияющими на время отклика СУБД, является количество и тип операций в запросе, количество строк и атрибутов обрабатываемых таблиц, а также интенсивность поступления запросов на сервер в единицу времени.

Довольно часто в реальных КИС некоторые запросы появляются непосредственно в группе, один за другим. При этом вероятность поступления каждого запроса в базу данных по отдельности достаточно мала, однако при поступлении одного запроса вероятность поступления других запросов изменяется. Учитывая этот факт, для построения операционного профиля пользователя удобно использовать марковские модели, хорошо известные из теории систем массового обслуживания.

Как основу для построения такой марковской модели возьмём файл журнала БД, в котором, как правило, содержатся запросы и временные метки их поступления (рис. 1).

```
>>> 2008-09-09 17:28:12 EEST||10.71.25.1(32915)||postgres||9289|| <<<LOG:
statement: select сокрзначение,полноезначение from константы where таблица =
'личность' and колонка  ='пол';
>>> 2008-09-09 17:28:12 EEST||10.71.25.1(32915)||postgres||9289|| <<<LOG:
statement: select сокрзначение,полноезначение from константы where таблица =
'личность' and колонка  ='пол';
>>> 2008-09-09 17:28:12 EEST||10.71.25.1(32915)||postgres||9289|| <<<LOG:
statement: select сокрзначение,полноезначение from константы where таблица =
'сведучеба' and колонка ='статус' order by полноезначение desc;
```

Рис. 1. Фрагмент файла журнала

Данный файл содержит время поступления запроса, идентификатор клиента, которым в данном случае является его IP-адрес, а также сам запрос. Входными параметрами для построения марковской модели кроме файла журнала, является также интервал времени, на основе которого строится данная марковская модель и минимальная глубина построения данной марковской модели, т.е. количество запросов, которое анализируется перед тем, как модель вернётся в начальное состояние.

Рассмотрим построение марковской модели для нагрузочного тестирования СУБД на основе запросов КИС. Пусть существует некоторое упорядоченное множество запросов Q к некоторой БД КИС. Оно включает в себя все запросы q_i ($i = 1...n$, n – количество запросов), выполненные в БД, а также специальный «пустой запрос» q, который означает, что в данный момент времени запросов к БД не поступало, и модель переходит в начальное состояние (состояние ожидания запроса) . Каждому запросу из БД сопоставляется состояние этого запроса s_i, а начальное состояние модели обозначим s_0. Далее из файла журнала анализируется заданный временной интервал T, разбитый с определённым шагом Δt на значения t_j ($j = 1..m$, m – число значений на указанном интервале). Таким образом, представленный файл журнала можно формализовать в виде множества троек вида

$$L = \{l_j \mid l_j = (t_j, id, q_i)\}. \quad (1)$$

Далее рассмотрим процесс построения марковской модели по множеству L. В начале процесса анализа модель находится в состоянии s_0. В случае, если в момент времени t_j пришёл запрос q_i от клиента id, то одна тройка с указанными параметрами отмечается во множестве L, и далее не рассматривается. Модель переходит в состояние s_i, при этом вероятность перехода в указанное состояние будет

$$P(s_i) = \frac{n_{si}}{m} \quad (2)$$

где n_{si} –количество переходов в состояние s_i, m –общее количество разбиений временного интервала T. В результате вычислений формируется четвёрка вида

$$(s_{n-1}, s_n, q_i, p), \quad (3)$$

где s_{n-1} – предыдущее состояние модели, s_n – новое состояние модели, q_i – запрос, вызвавший переход в состояние s_n, p – вероятность, рассчитанная по формуле (2). Множество троек (3) представляет собой марковскую модель нагрузочного тестирования СУБД.

Далее указанный процесс повторяется, используя в качестве базового состояния s_n. Процесс построения модели завершается в том случае, если всё тройки во множестве L отмечены.

Для реализации алгоритма введём операции $MARK(l_i)$ – отметки определённого элемента множества, операцию $M(L)$, возвращающую как результат мощность множества L, $ВЕРОЯТНОСТЬ(q_i)$ – результатом операции является вероятность, рассчитанная по формуле (2), ++ – увеличение на 1. Тогда схема алгоритма построения данной модели выглядит, как показано на рис. 2.

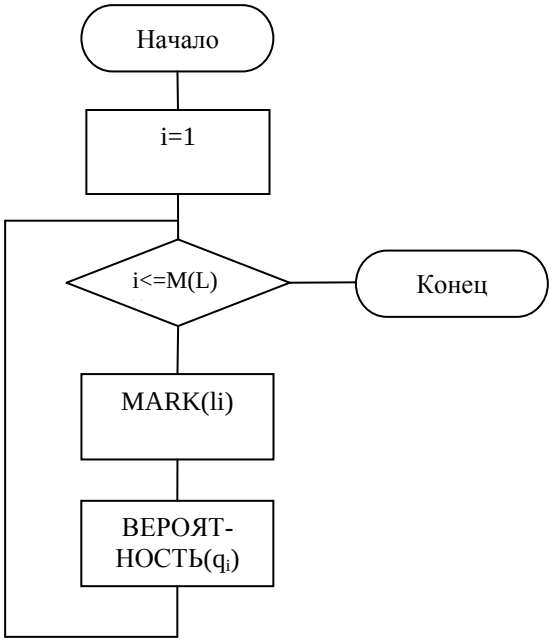

Рис. 2. Схема алгоритма построения марковской модели нагрузочного тестирования

Следует отметить, что марковскую модель можно строить как для каждого клиента БД (как было показано выше), так и для всей БД в целом. В последнем случае не рассматривается компонент id элемента множества L.

Рассмотрим далее непосредственно процесс нагрузочного тестирования СУБД с использованием запросов КИС. Пусть существует марковская модель (рис. 3).

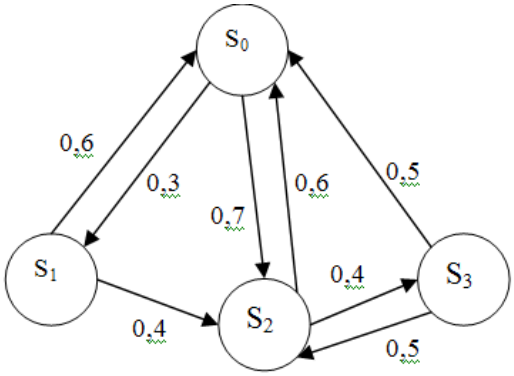

Рис. 3. Марковская модель

Тогда, полагая, что из начального состояния необходимо выполнить, например 10000 запросов легко рассчитать количество переходов в каждом случае, путём умножения вероятности, присвоенной исходящей дуге состояния, на количество запросов этого состояния. Для состояния, отличного от S_0 количество запросов равно сумме количеств переходов по входящим вершинам (рис. 4).

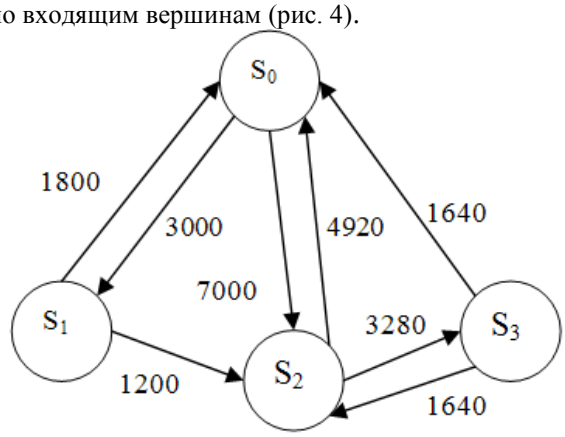

Рис. 4. Граф переходов запросов

Показанная на рис. 4 модель нагрузочного тестирования позволяет более точно смоделировать нагрузку на СУБД от конкретного клиента. При обходе графа, показанного на рис. 4 рёбра выбираются случайным образом. При проходе каждого ребра его счётчик уменьшается на 1 и СУБД отсылается соответствующий запрос. По достижении счётчиком ребра нулевого значения, ребро далее не рассматривается.

По итогам прохождения теста строится отношение «время ответа на конкретный запрос». Это отношение состоит из множества пар, первая компонента которых запрос, вторая – время отклика СУБД на данный запрос. Данное отношение при постоянных параметрах настройки и структуры таблиц и запросов обладает свойством функциональности, т. е. каждому запросу соответствует своё заданное время ответа.

При проведении тестов важным вопросом является получение пропорциональных результатов времени ответа на аналогичных тестах производительности. Актуальность этого вопроса связана с тем, что результаты теста зависят не только от параметров аппаратной и программной настройки сервера СУБД, но и от построения самого теста. Настройку СУБД целесообразно выполнять по результатам двух или более тестов производительности. Поэтому кроме тестирования, основанного на марковских моделях поведения клиентов СУБД, целесообразно проводить тестирование одним или несколькими синтетическими тестами, структура запросов которых в наибольшей степени совпадают со структурой запросов КИС. Для выяснения степени этого соответствия, построенные ранее марковские модели необходимо дополнить информацией о структуре запросов КИС.

Представление структуры запросов КИС в виде графа

Для решения рассмотренной выше задачи предлагается представлять структуру запроса в виде дерева операций. Операциями в данном дереве будут являться обычные операции реляционной алгебры. Для решения задачи оптимизации производительности базы данных дерево операций необходимо расширить, введя в него веса, обозначающие длительность исполнения определённой операции. Все веса реляционных операций могут быть рассчитаны как отношение среднего времени выполнения запроса реляционной операции к максимально возможному времени исполнения указанной операции.

Рассмотрим пример построения такого дерева для запроса SELECT * from tbl1 WHERE f2 IN (SELECT f2 FROM tbl2 WHERE f1=1). Согласно стандарту SQL первым выполняется вложенный запрос SELECT f2 FROM tbl2 WHERE f1=1, в результате которого вначале выполняется операция выборки из таблицы, а затем операция проекции по атрибуту f2. Таким образом, в реляционной алгебре данное выражение запишется следующим образом:

$$R = \pi_{f2}(\sigma_{f1=1}(tbl2)). \quad (4)$$

Рассуждая аналогичным образом, первый запрос является типичной операцией соединения двух отношений tbl и R и записывается в реляционной алгебре следующим образом:

$$L = tbl1 \triangleright\!\triangleleft_{(tb1.f2=R.f2)} R. \quad (5)$$

Учитывая формулы (4) и (5), можно синтезировать следующий граф операций (рис. 5):

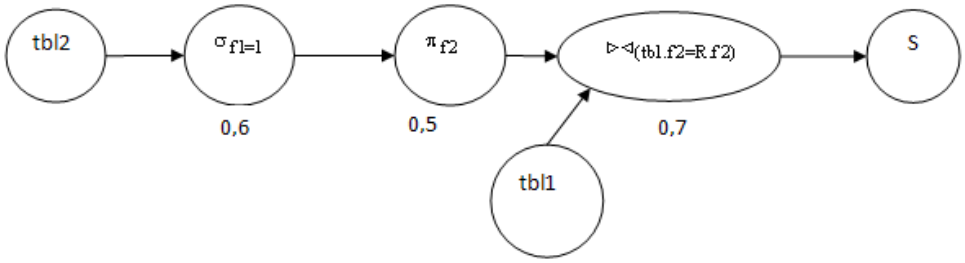

Рис. 5. Граф операций запроса

На рис. 5 вершина S обозначает соответствующую вершину марковской модели. Операторным вершинам на рис. 5 сопоставлены веса операций. В том случае, если один и тот же запрос является частью нескольких различных запросов, то из корневой вершины графа операций может исходить несколько связей.

Архитектура системы для адаптивной настройки СУБД

Для реализации предложенных моделей для конкретной СУБД предлагается реализовать систему экспертной настройки параметров, архитектура которой показана на рис. 6.

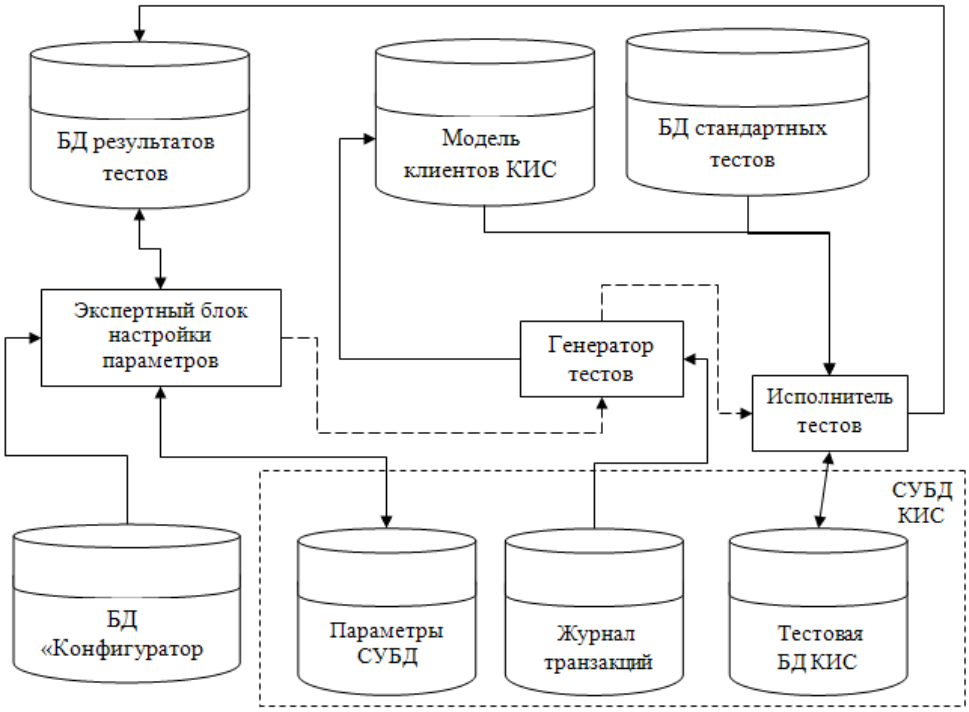

Рис. 6. Архитектура системы экспертной настройки параметров

Блок генератора тестов позволяет на основе журнала транзакций построить марковские модели для тестирования КИС (см. рис. 3), а также структуру запросов БД КИС в виде дерева операций. В дальнейшем эта информация используется исполнителем тестов для проведения нагрузочного тестирования СУБД как с использованием запросов, существующих в ИС, так и с использованием соответствующего синтетического теста, который определяется по подобию характеристик графа операций запроса синтетического теста и запроса КИС. В задачу генератора теста входит также динамическая настройка марковской модели в процессе работы СУБД.

По результатам тестов формируется набор значений времени ответа на тот или иной запрос, который передаётся экспертному блоку настройки параметров, который путём анализа структуры запроса и времени ответа на запрос проводит оптимизацию параметров СУБД.

База данных «Конфигуратор» хранит информацию о конфигурационных параметрах различных СУБД, а также дополнительную информацию о СУБД, такую как средняя скорость в каналах связи локальной сети, количество клиентов в СУБД, тип СУБД (локальная, распределённая однородная, распределённая гетерогенная), аппаратные характеристики серверов, оценка их быстродействия и т.п.

Такая архитектура системы позволяет реализовать адаптируемую к конкретным условиям работы СУБД настройку параметров, а также автоматизированную перенастройку параметров СУБД при изменении её аппаратно-программной конфигурации.

Выводы

В результате проведенного исследования была разработана модель проведения нагрузочного тестирования на основе журнала транзакций БД. Данная модель является расширением известных марковских моделей путём ввода в состояние модели графа структуры запроса. Такое расширение позволяет значительно расширить возможности тестирования и анализа его результатов, так как позволяет оценить частоту как запросов, так и фрагментов запросов. В графе структуры запроса рассматриваются также веса операций, что позволяет оценить длительность выполнения того или иного запроса.

Предложена архитектура адаптивной экспертной системы настройки параметров СУБД, которая позволяет автоматизировать перенастройку параметров при изменении аппаратно-программной конфигурации СУБД на основе журнала транзакций БД.

Перспективным направлением дальнейших исследований является разработка архитектуры и базы знаний для экспертной системы настройки параметров.

1. *Шнитман В.З., Кузнецов С.Д.* Серверы корпоративных баз данных. Информационно-аналитические материалы Центра информационных технологий. – http://www.citforum.ru/database/skdb
2. *Serlin O.* The History of DebitCredit and the TPC. The Benchmark Handbook for Database and Transaction Processing Systems, Jim Gray (Ed.), Morgan Kaufmann, 2nd Ed. – 1993. – P. 21–40.
3. *D. J. DeWitt.* The Wisconsin Benchmark: Past, Present, and Future. The Benchmark Handbook for Database and Transaction Processing Systems, Jim Gray (Ed.), Morgan Kaufmann, 2nd Ed. – 1993. – P. 269–316.
4. *C. Turbyfill, C. Orji and D. Bitton,* AS3AP: an ANSI SQL standard scalable and portable benchmark for relational database systems. The Benchmark Handbook for Database and Transaction Processing Systems, Jim Gray (Ed.), Morgan Kaufmann, 2nd Ed. – 1993 – P. 317–358.
5. *P.E. O'Neil.* The Set Query Benchmark. The Benchmark Handbook for Database and Transaction Processing Systems, Jim Gray (Ed.), Morgan Kaufmann, 2nd Ed. – 1993. – P. 359–395.
6. *R.G.G.Cattel*. The Engineering Database Benchmark. The Benchmark Handbook for Database and Transaction Processing Systems, Jim Gray (Ed.), Morgan Kaufmann, 2nd Ed. – 1993. – P. 397–434.