Уфа : УГАТУ, 2009 Вестник УГАТУ • Управление, ВТ и И Т. 13, № 2 (35). С. 198–205

УДК 004.658

В. В. МАРТЫНОВ, Е. Н. ПРОШИН

ПОДХОД К АНАЛИЗУ ХАРАКТЕРИСТИК ПРОИЗВОДИТЕЛЬНОСТИ БД

Статья посвящена оценке производительности программно-аппаратного комплекса системы управления базой данных (СУБД) и его улучшению с учетом экономической эффективности. Рассматриваются наиболее общие проблемы, с которыми сталкиваются администраторы СУБД, дается теоретический подход к их решению. *Производительность*; *информационная система*; *система управления базами данных*

При определении производительности информационной системы (ИС), в основе которой лежит СУБД, необходимо провести полное исследование программно-аппаратного комплекса, на котором функционирует ИС, а не только проводить измерение отдельных компонентов производительности СУБД. Современные программно-аппаратные комплексы настолько сложны, что проблема увеличения производительности может иметь решение в изменении любого из компонентов входящих в состав комплекса.

Анализу характеристик влияющих на производительность СУБД посвящена данная статья.

1. СОСТОЯНИЕ ВОПРОСА

Обновление аппаратного обеспечения не всегда сможет решить вопрос увеличения производительности ИС, а в некоторых случаях может ее и понизить. Важно определить, в чем заключаются «узкие места» ИС, какие действия следует предпринять для их преодоления, понять какую следующую проблему можно ожидать, и правильно наметить стратегию развития ИС.

Для крупных корпораций и баз данных (БД) больших объемов проблема определения производительности последних имеет еще один очень важный аспект: определения ИТ инфраструктуры для длительной промышленной эксплуатации приложений, т. е. в конечном итоге к определению первоначальных инвестиций в оборудование и базовое ПО. Так как различное оборудование и различные платформы могут в разы обеспечивать более высокую производительность одной и той же СУБД и на длительное время.

Проблемы выявления «узких мест» производительности СУБД напрямую связаны с методами и метриками измерений производительности и технологией их выполнения.

2. ПОСТАНОВКА ЗАДАЧИ

Как правило, СУБД могут функционировать на нескольких платформах, а нередко и в разных редакциях, предназначенных для решения разнообразных задач или обслуживания различного количества данных и пользователей. Из последних тенденций развития СУБД следует отметить поддержку XML и Web-сервисов.

Известно, что по назначению базы данных можно подразделить на:

- оперативные, или OLTP-базы данных (On-Line Transaction Processing) – обычно в эти базы данных осуществляется интенсивный ввод данных, а вот число адресованных к ним сложных запросов невелико. Критичным для таких БД является время обработки транзакций;
- хранилища данных (OLAP, On-Line Analytical Processing), применяемые, как правило, в аналитических приложениях и системах поддержки принятия решений – к ним обычно адресуется большое число запросов, в том числе и сложных, но ввод данных в них не столь интенсивен и в основном осуществляется периодически (раз в сутки, месяц, год). Требование ко времени обработки транзакций здесь менее критично.

На серверах БД имеет место быть большое разнообразие выполняемых приложений, которые, как правило, стараются загрузить различные части системы. Далеко не все приложения интенсивно используют процессорные ресурсы,

Контактная информация: (347) 273-08-22

и не все из них связаны с интенсивным вводом/выводом. Поэтому смесь таких приложений на одной системе может обеспечить достаточно равномерную загрузку всех ресурсов. Естественно неправильно подобранная смесь может дать совсем противоположенный эффект.

Имея набор целевых показателей производительности конечного пользователя и стоимостных ограничений, можно спрогнозировать возможности определенного набора компонентов, которые включаются в конфигурацию системы. Подобная оценка сложна и связана с неточностью измерений. Доступная потребителю информация о самих системах, операционных системах, программном обеспечении инфраструктуры (СУБД и мониторы обработки транзакций), как правило, носит очень общий характер.

При наличии стольких неопределенностей попытаемся системно подойти к оценке и выбору конфигурации. Намного проще решить, что определенная конфигурация не сможет обрабатывать некоторые виды нагрузки, чем определить с уверенностью, что нагрузка может обрабатываться с учетом определенных ограничений производительности. Более того, реальное использование систем показывает, что имеет место тенденция заполнения всех доступных ресурсов рабочими данными и процессами. Как следствие, системы, даже имеющие некоторые избыточные ресурсы, со временем не будут воспринимать дополнительную нагрузку.

Для выполнения анализа конфигурации, система на основе СУБД (под которой понимается весь комплекс компьютеров, периферийных устройств, сетей и программного обеспечения) должна рассматриваться как ряд соединенных друг с другом компонентов. Ограничения производительности некоторой конфигурации по любому направлению обычно могут быть предсказаны исходя из анализа наиболее слабых компонентов. Точная оценка полной конфигурации требует ее рассмотрения, как на уровне сети, так и на уровне компонент или подсистем.

Эта же методология может быть использована для настройки системы после ее инсталляции: настройка системы и сети выполняются, как правило, после предварительной оценки и анализа узких мест. Более точно, настройка конфигурации представляет собой процесс определения наиболее слабых компонентов в системе и устранения этих узких мест.

3. ПОДХОДЫ К АНАЛИЗУ «УЗКИХ МЕСТ» ТИПОВОЙ ИС НА ОСНОВЕ СУБД

Как СУБД, так и приложения, ориентированные на использование баз данных, имеют отличия по разным критериям. Подавляющее большинство систем работает по одной и той же концептуально общей реляционной схеме.

В типовую архитектуру многослойной системы могут входить следующие компоненты, влияющие на производительность (рис. 1):

- хранение данных (база данных) – программные настройки СУБД, отвечающие за доступ, хранение и индексирование данных;
- оборудование – аппаратный комплекс, включающий в себя основные компоненты сервера: процессор, оперативная память, дисковая подсистема, сетевой интерфейс;
- архитектура информационной сети – функциональное взаимодействие и сетевая топология компьютерной сети, объединяющей серверы, клиентские рабочие места, а также СУБД с установленным лицензионным программным обеспечением;
- логика доступа к данным – характеризуется запросами и их структурой в существующей модели БД;
- бизнес-процессы, реализованные в ИС – сервисный слой приложения включает бизнес-логику приложения, а также интерфейс системы.

Зачастую информационные приложения, автоматизирующие ту или иную деятельность предприятия, а также логика доступа к данным недоступны или являются сложными, изменение которых ведет к изменению бизнес-процессов приложений. Поэтому далее в статье исследуем характеристики, влияющие на производительность ИС, но не изменяющие семантику бизнес-процессов приложений ИС.

В табл. 1 представлены шесть ключевых областей, которые стоит контролировать при проведении анализа производительности сервера и для поиска «узких мест» СУБД.

Кроме того, к снижению производительности может приводить наличие комбинации «узких мест». Такие проблемы наиболее трудно поддаются локализации, т. к. анализ отдельных компонент не дает очевидных результатов, по которым можно принять решение о наличии «узкого места».

Важным является введение оценочных характеристик компонент и их зависимостей их изменения для проведения оптимизации производительности БД.

4. МОДЕЛИ И ПРИБЛИЖЕНИЯ, ИСПОЛЬЗУЕМЫЕ ПРИ КОНФИГУРАЦИИ ПРОГРАММНО-АППАРАТНОГО КОМПЛЕКСА СУБД

Существует достаточно большое количество теоретических разработок и практических рекомендаций по увеличению производительности СУБД весьма разнящихся в своей основе. Большая часть этих методик [2] используется производителями серверов, которые зависят от того приложения, которое собираются развертывать.

При этом нужно понимать, что если для планирования производительности систем, имеющих четко определенную нагрузку, таких как SAP R/3, Oracle Applications, Lotus Notes, определение необходимой конфигурации затруднительно, то для инфраструктурного ПО на основе СУБД Oracle и SQL Server и т. п., для которых реальная нагрузка может очень сильно отличаться в зависимости от используемого приложения, такие расчеты должны строиться на некоторых приближенных моделях.

В настоящее время существуют следующие основные методики определения необходимой конфигурации: модель, основанная на числе пользователей; модель, основанная на пропускной способности и производительности, и модель, построенная на основании результатов тестов производительности. Рассмотрим их подробнее.

Модель, построенная на числе пользователей. В основе этой модели лежит понятие пользователя. Необходимо определить поведение пользователя, общее число пользователей и число одновременно работающих с приложением пользователей. При этом рассчитывается необходимое число и мощность процессоров, память, дисковое пространство и пропускная способность сетевого оборудования. В качестве исходных данных используется усредненные данные по аналогичным приложениям, имеющимся у производителя.

Модель, построенная на производительности. Если доступна информация о сложности выполняемых транзакций, среднем количестве данных, приходящихся на одного пользователя и некоторых других параметрах, то в качестве базы можно использовать эту модель. В этом случае при расчете необходимой конфигурации предусматривается ряд упрощений, таких как:

- увеличение числа пользователей и выполняемые ими транзакции линейно влияет на нагрузку сервера;
- сложные процессы, такие как составление отчетов, выполняются во время минимальной загрузки системы;
- основное ограничение производительности связано только с вычислительной мощностью (числом процессоров).

Модель на основе тестов производительности. Эта модель наиболее близка к реальной ситуации. Основное допущение для SAP состоит в том, что нагрузки от работы отдельных модулей считаются независимыми и складываются.

Применение этих трех моделей соответствует трем типам знаний заказчика о будущей системе. В первом случае, имеется только самая базовая информация о потенциальном числе клиентов приложения и некоторое понимание их поведения. Во втором случае, постановщик задачи знает некоторые количественные характеристики работы клиентов. В третьем случае – в основе методики лежит четко построенный план внедрения приложения с подробной проработкой деталей.

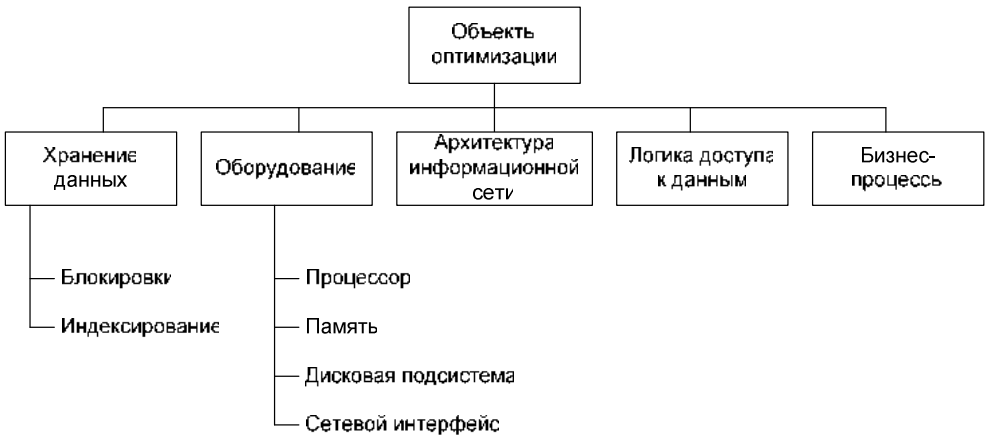

Рис. 1. Объекты оптимизации типовой информационной системы

Таблица 1

Ключевые области анализа производительности СУБД

Категория анализа	Эффекты на СУБД
Процессор	Высокая загрузка процессора указывает на его большую нагрузку различными процессами сервера. Недостаток процессорного времени для процессов, протекающих в СУБД, приводит к замедлению работы сервера.
Память	Недостаточный объем оперативной памяти, включая распределенную память (файл подкачки) или ее недоступность ухудшает общую производительность. Операционная система интенсивно осуществляет обмен с файлом подкачки тогда, когда в нем находятся необходимые страницы или память нужно освободить для редко используемых страниц.
Дисковая подсистема	Медленные диски могут стать причиной снижения производительности I/O (input/output – дисковые операции чтения и записи), что ухудшает работу транзакционного механизма.
Сетевая подсистема	Не правильная конфигурация информационной сети и выбранные каналы связи могут привести к медленной работе системы, функционирующей на сети.
Блокировки	Процесс может вынуждать другие процессы ожидать доступа к ресурсу, замедляя или останавливая их работу.
Индексирование	Операции сканирования таблиц, сортировки, слияния, доступ по мультииндексу сильно загружают систему, поэтому необходима индексация данных для более быстрого доступа к данным.

Вместе с тем, не следует считать, что чем больше информации мы собрали о системе, тем более точным будет расчет. Основанный на числе пользователей метод дает наиболее грубые оценки, но он гораздо менее чувствителен к ошибкам, в то время, как ошибки в определении типичного поведения пользователей в других методах могут привести к большим ошибкам.

Очевидно, что большое разнообразие деталей даже в типовых приложениях не позволяет точно оценить необходимые вычислительные ресурсы при планировании проектов. В тоже время, крупнейшие производители аппаратного и программного обеспечения наработали огромный опыт в реализации подобных проектов и частично материализовали его в виде специальных приложений – сайзеров[1] [3]. С теоретической точки зрения, сайзеры, основанные на стандартных тестах приложений от производителя, более точные, чем сайзеры, основанные на статистическом анализе большого числа разнообразных инсталляций. При этом следует учитывать, что большинство производителей не указывает конкретный механизм работы сайзера и о нем можно только догадываться из общих соображений.

Кроме того, для приложений масштаба предприятия, сайзеры являются важным, но недостаточным инструментом. Основные производители платформ размещают на своих сайтах большое количество дополнительных материалов по методикам сайзинга, сбора необходимой информации для сайзинга, важным аспектам сайзинга отдельных компонентов систем и другую информацию, изучение которой обязательно для корректной оценки проекта. Кроме того, для действительно больших проектов, производители рекомендуют не использовать сайзеры, а обратиться непосредственно в специализированный центр производителя, где уже с участием специалистов будет составлен предварительный проект.

Для небольших и средних проектов использование сайзеров позволяет значительно сэкономить время избежать некоторых ошибок. При этом сама процедура сбора информации для сайзера и работа с ним служат хорошей схемой для приведения документации и планов проекта в порядок. А сконцентрированный опыт работы производителя в тысячах других проектов может оказать значительную помощь.

5. АНАЛИТИЧЕСКОЕ РЕШЕНИЕ

В [1], для увеличения производительности БД предложены объектно-реляционные логические модели представления данных уменьшающие избыточность данных БД. В соответствии с ГОСТ Р ИСО/МЭК 9126, для описания качества БД предложен критерий надежности, как функция тестируемости

$$E_m = (Q_{тм})/(Q_{ом}), \qquad (1)$$

где $Q_{тм}$ – количество элементов, отработавших в процессе тестирования и отладки, $Q_{ом}$ – общее число элементов БД. Количество элементов БД является функцией сложности БД.

[1] Сайзинг – определение оптимальной конфигурации аппаратно-программных средств.

Тем же ГОСТом установлены две основные субхарактеристики оценки эффективности программных средств, это эффективность ресурсная и эффективность временная. В качестве меры ресурсной эффективности в [1] предложено использовать меру объема индексов БД, а в качестве меры временной эффективности БД – время выполнения одной бизнес транзакции.

В отличие от подхода [1] нами предлагается **комплексное увеличение производительности БД**, позволяющие оптимизировать практически все компоненты средств работы с БД: **аппаратный комплекс, программные настройки СУБД, а также архитектуры информационной сети**.

В качестве меры ресурсной эффективности используются различные характеристики, влияющие на производительность СУБД. Тогда ресурсную эффективность $E_р$ БД можно выразить как $E_р = f(P_{ак}, P_{пк}, P_{ас})$, где $P_{ак}$ – характеристики производительности аппаратного комплекса СУБД, $P_{пк}$ – характеристики производительности программного комплекса СУБД, $P_{ас}$ – характеристики архитектуры информационной сети, на которой функционирует анализируемая с точки зрения производительности СУБД.

В свою очередь **производительность аппаратного комплекса** СУБД $P_{ак}$ зависит от

$$P_{ак} = f(P_{пр}, P_{п}, P_{д}, P_{с}), \quad (2)$$

где $P_{пр}$, $P_{п}$, $P_{д}$, $P_{с}$ – характеристики производительности частей аппаратного комплекса СУБД по процессорной системе, оперативной памяти, дисковой и сетевой подсистемах соответственно.

В свою очередь

$$P_{пр} = f(p_{пр}, M_{пр}), \quad (3)$$

где $P_{пр}$ –загрузки процессорной системы, $M_{пр}$ – быстродействие процессора;

$$P_{п} = f(p_{п}, V_{п}, M_{п}), \quad (4)$$

где $P_{сп}$ – свободная оперативная память, $V_{п}$ – объем оперативной памяти, $M_{п}$ – быстродействие оперативной памяти;

$$P_{д} = f(p_{д}, M_{д}), \quad (5)$$

где $P_{д}$ – загрузка дисковой подсистемы, $M_{д}$ – быстродействие дисковой подсистемы;

$$P_{с} = f(p_{с}, M_{с}), \quad (6)$$

где $P_{с}$ – загрузка сетевой подсистемы, $M_{с}$ – скорость передачи данных, определяемая типом сети.

Интервалы изменения характеристик можно записать как:

$$0 \le p_{пр} \le 1,\ 0 < p_{п} \le 1,\ 0 \le P_{д} \le 1,\ 0 \le p_{с} \le 1.$$

Производительность программного комплекса СУБД $P_{пк}$ можно выразить как

$$P_{пк} = f(P_{бк}, P_{инд}, P_{блок}), \quad (7)$$

где $P_{бк}$ – эффективность выбора кэш-буфера СУБД, $P_{инд}$ – эффективность выбора системы индексов, $P_{блок}$ – выбор степени изоляции данных при блокировках к данным.

$$P_{бк} \in (0, V_{б}), \quad (8)$$

где $V_{б}$ – объем кэш-буфера СУБД.

Уровень изоляции определим следующим множеством:

$$B = \{B_{БД}, B_{объект}, B_{экстент}, B_{стр}, B_{зап}\}, \quad (9)$$

где $B_{БД}$, $B_{объект}$, $B_{экстент}$, $B_{стр}$, $B_{зап}$ – соответствующие уровни изоляции БД, объекта, экстента, страницы и записи.

Тогда эффективность степени изоляции данных при блокировках к данным будет выражаться следующей формулой:

$$P_{блок} = f(B). \quad (10)$$

Множество индексов определим следующим множеством:

$$I = \{I_{класт}, I_{некласт}, I_{выч.некласт}, I_{XML}, I_{предст}\}, \quad (11)$$

где $I_{класт}$, $I_{некласт}$, , $I_{выч.некласт}$, I_{XML}, $I_{предст}$ – множество кластеризованных и некластеризованных индексов, индексов в вычисляемых полях, XML-индексов и индексов в представлениях.

$$I_{класт} = \{I_{ун.класт}, I_{ун.сост.класт}\},\ I_{класт} \subset I, \quad (12)$$

где $I_{ун.класт}$, $I_{ун.сост.класт}$ – множество уникальных и составных кластеризованных индексов.

$$I_{некласт} = \{I_{пр.некласт}, I_{покр.некласт}\},\ I_{некласт} \subset \mathrm{I}, \quad (13)$$

где $I_{пр.некласт}$, $I_{покр.некласт}$ – множество простых и покрывающих некластеризованных индексов.

$$I_{XML} = \{I_{перв.XML}, I_{втор.XML}\},\ I_{XML} \subset I, \quad (14)$$

где $I_{перв.XML}$, $I_{втор.XML}$ – множество первичных и вторичных XML-индексов.

Каждый индекс имеет свой уровень фрагментации, а именно k – коэффициент насыщенности дерева $B+$.

Соответственно для реляционной БД объем индексов определяется следующим образом [1]

$$V_{инд} = U_i U_s \frac{k^{\frac{\ln(N)}{\ln(k)}+1} - 1}{k-1}, \quad (15)$$

где $U_i = U_a \cdot k_i$ – количество индексов на таблице (количество атрибутов, коэффициент индексирования), U_i – количество сущностей, N – количество записей в таблице.

При исследовании фактической производительности оказывается, что наиболее часты-

ми проблемами индексирования являются (в порядке важности) отсутствие индексов и лишние, ненужные индексы. Поэтому целесообразно произвести индексирование на основе таблиц, а не отдельных запросов.

Нами предлагается метод, обеспечивающий максимальное увеличение производительности БД, используя наименьшее количество индексов. Создадим так называемую матрицу CRUD M_{CRUD} (create, retrieve, update, delete), в которой поля будут соответствовать полям таблицы, а столбцы – процедурам и запросам, которые осуществляют доступ к таблице и формируют схему данных (табл. 2). В M_{CRUD} будут использованы следующие аббревиатуры: S – для отбираемых столбцов; O – для упорядочения по столбцу; W – для ссылок на столбец в предложении WHERE, G – для группировки по функции.

Здесь $\Pi_i \in \Pi$ – множество полей таблицы;

$З_j \in З$ – множество запросов использующих таблицу;

$s_{ij} \in S$ – множество обозначающее включение поля Π_i в запросе $З_j$ (s_{ij} = 1, если поле i присутствует в запросе j и 0 – в ином случае);

$_{oij} \in O$ – множество обозначающее упорядочивание по полю Π_i в запросе $З_j$ (o_{ij} = 1, если поле i присутствует в запросе j и 0 – в ином случае);

$w_{ij} \in W$ – множество обозначающее использование условия по полю Π_i в запросе $З_j$ (w_{ij} = 1, если поле i присутствует в запросе j и 0 – в ином случае);

$g_{ij} \in G$ – множество обозначающее группировку по полю Π_i в запросе $З_j$ (g_{ij} = 1, если поле i присутствует в запросе j и 0 – в ином случае).

Этот процесс требует в первую очередь определения кластеризованного индекса. А затем по данной матрице определяем индексы по наиболее часто встречающимся полям таблицы в запросах, поэтому некоторые индексы могут быть использованы сразу несколькими процедурами.

Таким образом, эффективность выбора системы индексов будет выражаться следующей формулой:

$$P_{\text{инд}} = f(I, M_{\text{CRUD}}, V_{\text{инд}}). \qquad (16)$$

Далее рассмотрена задача построения математической модели общей **структуры информационной сети** предприятия, использующей СУБД.

Введены описания следующих множеств:

M – множество видов серверов, с учетом установленного лицензионного программного обеспечения и возможностью формирования баз данных, $i \in M$;

P – множество вариантов установки серверов, обслуживающих запросы к базам данных, $p \in P$;

Q – множество СУБД, хранящих соответствующую часть базы данных, $q \in Q$. Среди них имеются объектно-ориентированные СУБД и реляционные СУБД;

N – множество клиентских рабочих мест (КРМ), формирующие запросы к БД посредством СУБД, $j \in N$.

Множество N будем считать известным (оно определено исходя из существующей схемы информационной сети). Множество Q состоит из 3 элементов: инсталляция на сервере p только объектно-ориентированной СУБД, только реляционной СУБД или обеих СУБД.

Объект БАЗА ДАННЫХ $s \in S$ характеризуется объемом b_s, Мбайт и временем отклика t_s, сек.

Основные параметры **объекта СЕРВЕР** $i \in M$ являются K_i – стоимость аппаратуры и программного обеспечения, исходя из стоимостей ОС, СУБД и прикладных программ – капиталоемкость сервера и C_i – стоимость эксплуатации сервера, выраженная в затратах в течение года. В перечне типов сервера введем «Нуль-сервер», производительность и стоимостные характеристики которого – нулевые. Выбор «Нуль-сервера» означает отказ от его установки в данном месте.

Множество вариантов установки сервера $p \in P$ характеризуется:

1) местом его размещения;

2) x_p^i – выбором типа сервера $i \in M$ (x_p^i = = 1, если в качестве сервера выбран аппаратно-программный комплекс с номером $p \in \in P$, и 0 – в ином случае);

3) y_p^s – перечнем, количеством и видом обслуживаемых баз данных (y_p^s = 1, если база данных $s \in S$ размещена на сервере $p \in P$ и 0 – в ином случае);

4) z_p^j – перечнем, количеством и видом обслуживаемых клиентских рабочих мест (z_p^j = 1, если клиентское рабочее место $j \in N$ закреплено за сервером p или 0 – в ином случае).

Таблица 2

Матрица CRUD

Поле	$З_1$		$З_j$	…	$З_{Nz}$
$П_1$	s_{11}, o_{11}, w_{11}, g_{11}		s_{1j}, o_{1j}, w_{1j}, g_{1j}		s_{1Nz}, o_{1Nz}, w_{1Nz}, g_{1Nz}
$П_i$	s_{i1}, o_{i1}, w_{i1}, g_{i1}		s_{ij}, o_{ij}, w_{ij}, g_{ij}		s_{iNz}, o_{iNz}, w_{iNz}, g_{iNz}
…					
$П_{Np}$	S_{Np1}, o_{Np1}, w_{Np1}, g_{Np1}		s_{Npj}, o_{Npj}, w_{Npj}, g_{Npj}		s_{NpNz}, o_{NpNz}, w_{NpNz}, g_{NpNz}

Объект СУБД $q \in Q$ характеризуется x_q^i – выбор закрепления СУБД $q \in Q$ за сервером $i \in M$ (x_q^i =1, если для установки на сервере выбрана СУБД $q \in Q$ или 0 – в ином случае).

Объект КЛИЕНТСКОЕ РАБОЧЕЕ МЕСТО $j \in N$ характеризуется:

1) K_j – стоимостью аппаратной и программной частей КРМ, включая стоимость ОС, прикладного ПО и ПО доступа к СУБД (клиентской части СУБД);

2) C_j – стоимостью эксплуатации клиентского рабочего места, выраженной в денежных затратах в год;

3) z_q^j – закрепление клиентского рабочего места за определенной СУБД $q \in Q$ (z_q^j = = 1, если клиентское рабочее место $j \in N$ подключено к СУБД $q \in Q$ или 0 – в ином случае).

Определим управляемые факторы задачи (ее неизвестные):

1) x_p^i – индикатор выбора типа сервера $m \in M$;

2) y_p^q – индикатор объектно-ориентированной или реляционной СУБД, установленной на сервере $p \in P$ (y_p^q = 1, если СУБД $q \in$ $\in Q$ размещена на сервере $p \in P$ и 0 – в ином случае);

3) z_p^j – индикаторы перечня обслуживаемых клиентских рабочих мест, закрепленных за сервером $p \in P$, (z_p^j = 1, если КРМ $j \in \in N$ прикреплено к серверу $p \in P$ и 0 – в ином случае);

4) z_q^j – индикаторы перечня обслуживаемых клиентских рабочих мест, нуждающихся в обращении к СУБД $q \in Q$, (z_q^j = 1, если КРМ $j \in N$ использует в своей работе СУБД $q \in Q$ и 0 – в ином случае).

Ограничения задачи складываются из:

1) Соотношений связи между логическими переменными модели:

$$\sum_{i \in M} x_p^i = 1, p \in P \quad (17)$$

(на каждом месте размещения реализуется сервер некоторого типа, возможно 0 – сервер);

$$\sum_{i \in M} x_q^i = 1, q \in Q \quad (18)$$

(на каждом сервере установлена некоторая СУБД);

$$\sum_{q \in Q} y_q^s = 1, s \in S \quad (19)$$

(каждая база данных обслуживается некоторой СУБД);

$$\sum_{p \in P} z_p^j = 1, j \in N \quad (20)$$

(каждый пользователь закреплен за каким-либо сервером);

$$\sum_{q \in Q} z_q^j = 1, j \in N \quad (21)$$

(каждый пользователь использует в своей работе некоторую СУБД).

2) Границы переменных и условий неотрицательности:

$$0 \le x_p^i \le 1, 0 \le y_p^s \le 1, 0 \le z_p^j \le 1,$$

$$i \in M, p \in P, s \in S, j \in N;$$

3) Дискретности переменных:

x_p^i, y_p^s, z_p^j – целые, $i \in M, p \in P, s \in S, j \in$ $\in N$;

Пусть Z^* – капитальные затраты на внедрение ИС и Z^{**} – текущие затраты эксплуатации ИС. При расчете показателя Z^* учитываются затраты на приобретение серверов, клиентских рабочих мест и другого компьютерного оборудования. При расчете показателя Z^{**} учитываются затраты эксплуатации серверов ИС.

Представленные экономические показатели позволяют рассматривать несколько вариантов задачи и ее целевой функции:

1) минимизации времени отклика при возможном ограничении капиталовложений и текущих затрат:

$$t_s = K_0 + K_1 \cdot b_s \to \min,$$
$$Z^* = \sum_{p \in P} \sum_{i \in M} K_i \cdot x_p^i + \sum_{j \in N} K_j \le Z^*_{\max}, \quad (22)$$
$$Z^{**} = \sum_{p \in P} \sum_{i \in M} C_i \cdot x_p^i \le Z^{**}_{\max};$$

2) минимизации капиталовложений при возможном ограничении текущих затрат и времени отклика:

$$Z^* = \sum_{p \in P} \sum_{i \in M} K_i \cdot x_p^i + \sum_{j \in N} K_j \to \min,$$
$$t_s = K_0 + K_1 \cdot b_s \le t_{\max}, \quad (23)$$
$$Z^{**} = \sum_{p \in P} \sum_{i \in M} C_i \cdot x_p^i \le Z^{**}_{\max};$$

3) минимизации текущих затрат при возможном ограничении капиталовложений и времени отклика:

$$Z^{**} = \sum_{p \in P} \sum_{i \in M} C_i \cdot x_p^i \to \min,$$
$$t_s = K_0 + K_1 \cdot b_s \le t_{\max}, \quad (24)$$
$$Z^* = \sum_{p \in P} \sum_{i \in M} K_i \cdot x_p^i + \sum_{j \in N} K_j \le Z^*_{\max}.$$

Представленные математические модели приводят к линейной задаче целочисленной оптимизации с булевскими переменными. Показано, что т. к. возможно внедрение ИС с большим количеством пользователей, вариантов установки сервера и используемых БД, то ее решение является достаточной сложной задачей. В связи с этим предложено использовать для решения задач метод ветвей и границ.

ВЫВОДЫ

В данной статье изложены основные теоретические идеи, лежащие в основе оптимизации производительности ИС. Использование представленных математических моделей позволяет решить следующие задачи:

1) минимизации времени отклика при возможном ограничении капиталовложений и текущих затрат;

2) минимизации капиталовложений (затрат на внедрение ИС) при возможном ограничении текущих затрат и времени отклика;

3) минимизации текущих затрат при возможном ограничении капиталовложений и времени отклика.

Данные модели были апробированы в ходе модернизации программно-аппаратного комплекса системы правления гостиницей Epitome PMS в ОАО «Аврора», в ведении которой находится гостиничный комплекс «Президент Отель», Россия, Уфа. В частности, были проведены работы по оптимизации производительности БД данной системы, которые привели к уменьшению времени отклика клиентов системы на 27%.

СПИСОК ЛИТЕРАТУРЫ

1. **Масленников В. А., Левков А. А.** Проблемы организации структуры данных в сверхбольших базах данных // Науч.-техн. журнал «Системы управления и информационные технологии». 2007. № 3.1(29). С. 169–176.

2. **Елашкин М.** Планирование проектов с помощью сайзеров [Электронный ресурс] // Журнал «CIO». 2003. № 4 (14). [Электронный ресурс] (http://offline.cio-world.ru/2003/14/26010/).

3. **Petrossian T., Matzou A., Wong K.** An Introduction to System Sizing for Data Warehousing Workloads // IBM Redbooks, [Электронный ресурс] (http://www.redbooks.ibm.com/abstracts/redp3896.html).

4. **Martynov V. V., Proshin E. N.** Designing distributed DBMS with reference to a DB of materials for large machine-building association // The 10th Intern. workshop on Computer Science and Information Technologies Csit'2007. V. 2. P. 222–228.

ОБ АВТОРАХ

Мартынов Виталий Владимирович, проф., зав. каф. эконом. информатики, рук. БРЦНИТ. Дипл. инж.-мех. (МПИ, 1981). Д-р техн. наук по АСУ (УГАТУ, 2000). Иссл. в обл. информ. систем, иссл. операций, прикл. геометрии.

Прошин Евгений Николаевич, асс. каф. эконом. информатики. Дипл. спец. по мат. обесп. и админ. информ. систем (УГАТУ, 2006). Иссл. в обл. информ. систем и технологий.