УДК: 004.3

1.11. АНАЛИЗ МЕТОДОВ И ИНСТРУМЕНТОВ ОПТИМИЗАЦИИ ФУНКЦИОНИРОВАНИЯ ВЫСОКОНАГРУЖЕННЫХ СИСТЕМ

Харазян А. А.[1], ведущий разработчик
[1]Высшая школа экономики, Москва, Россия

В данной статье выполнен анализ методов и инструментов оптимизации работы высоконагруженных систем. Приведено описание понятия высоконагруженной системы, основные причины возрастания нагрузки. В частности, была обоснована необходимость реализации индивидуальной системы в зависимости от предъявляемых к ней требований и функциональных задач, а также зависимость нагрузки на систему от уровня ее масштабирования, количества работающих с ней пользователей, а также объема обрабатываемых данных. Представлены наиболее распространенные пути оптимизации работы высоконагруженных систем.

Введение

В большинстве случаев под термином «высоконагруженная система» разработчики подразумевают веб-сервисы, выполняющие роль сайта, с которым одновременно работает большое число пользователей. Однако к числу данных систем можно отнести различного рода информационные системы и бизнес-приложения, имеющие существенные объемы данных. Именно по этой причине вопросы оптимизации высоконагруженных систем актуальны не только в веб-разработке, но и при реализации любых приложений, в работе которых имеется серверная и клиентская часть. Соответственно, под высоконагруженной системой следует понимать приложение, функционирующее с высоким уровнем нагрузки вследствие большого числа единовременно работающих с ним пользователей, больших объемов обрабатываемых данных, а также наличия большого количества расчетов и вычислений. Перечисленные факторы могут характеризовать высоконагруженное приложение как в совокупности, так и по отдельности. Однако при наличии одного из данных факторов можно будет с уверенностью утверждать, что для своей работы высоконагруженные системы требуют большого объема ресурсов [2].

Методология исследования

Целью статьи является провести анализ методов и инструментов оптимизации работы высоконагруженных систем. Объект исследования – высоконагруженные системы, выполняющие роль сайта, с которым одновременно работает большое число пользователей. Методы исследования – анализ, синтез, обобщение источников информации по проблеме исследования.

Результаты исследования и их обсуждение

С целью анализа высоконагруженных систем необходимо более подробно рассмотреть их отличительные особенности.

1. Высоконагруженные системы представляют жесткую систему, которая подразумевает наличие возможности лишь незначительных изменений для некоторых ее составляющих подсистем. Подобную систему невозможно сделать гибкой по причине сложности ее внутренней структуры, которая не может быть универсальной. Работа с данными в стабильном и устойчивом виде требует учета особенностей структуры базы данных. На основании данной структуры, а также учета объема обрабатываемых данных и частоты обращений к ним должны использоваться максимальные возможности данной системы. В случае, если реализовать работу данной системы с точки зрения гибкости, то потребуется существенное потребление ресурсов.

2. Высокая скорость отклика представляет собой одно из важнейших качеств и особенностей работы высоконагруженных приложений. Поскольку работа с данными реализуется посредством запросов, то скорость ответа на запрос должна быть высокой: чем дольше система будет отвечать на данный запрос, тем дольше пользователь будет ожидать требуемые для его работы данные [4].

3. Масштабируемость -– это важное качество для высоконагруженной системы, так как при работе с данными рост объемов информационной базы может привести к увеличению нагрузки. Именно для этого и реализуется масштабируемость, которая достигается чаще всего двумя основными путями. Первый вариант – вертикальное масштабирование, реализуемое как увеличение уровня производительности для отдельных компонентов системы с целью роста ее общей производительности, является наиболее простым вариантом, так как не требует внесения каких-либо изменений в системе. Часто это достигается посредством замены наиболее медленных элементов на более мощные и быстродействующие. Вторым вариантом является горизонтальное масштабирование, при котором система разделяется на структурные компоненты, которые разносят по различным вычислительным системам, а серверная система разносится по нескольким параллельно работающим серверным платформам. Данный метод требует внесения в систему дополнительных узлов, которые должны функционировать в формате единого целого, для чего необходимо вносить большое количество изменений в программном коде системы для оптимизации использования вычислительных ресурсов. Реализация *верти-*

кального масштабирования является простым вариантом, однако оно не может быть бесконечным, именно по этой причине горизонтальное масштабирования считается более предпочтительным. Однако необходимо учитывать потребности при выполнении процедур масштабирования системы и выбирать наиболее подходящий вариант [7].

4. Модульная структура системы достигается посредством реализации высоконагруженной системы в формате отдельных блоков, которые впоследствии распределяются по различным серверным платформам либо наиболее высоконагруженный модуль распределяется по нескольким серверам и выполняется в параллельном режиме. Второй вариант является отличным решением, однако в результате параллельной обработки начинает возрастать уровень рассинхронизации данных. По мере роста нагрузки вероятность возникновения многосистемного состояния обрабатываемых данных в результате одновременных действий пользователей также возрастает. Именно по этой причине оптимальным считается вариант, когда обработку нагруженного решения целесообразно не распараллеливать, а обособленно выносить на более производительные платформы [3].

5. Высокий уровень нагрузки на слой интеграции системы обусловлен высокой степенью разбиения системы на отдельные модули. При этом, когда в ее состав добавляется все большое количество модулей, это приводит к росту числа коммуникаций между ними, которые должны работать быстро и качественно. Получается, что по мере роста количества модулей в геометрической прогрессии растет уровень сложности взаимодействия между ними, а это приводит к нагруженности интеграционного слоя высоконагруженной системы.

6. Эксклюзивность представляет собой особенность работы высоконагруженного приложения, поскольку для их реализации отсутствуют стандартизированные универсальные решения. По этой причине каждая система будет эксклюзивной, так как при ее реализации необходимо ориентироваться на конкретные бизнес-требования.

7. Активное использование принципов дублирования критических узлов. Так как высоконагруженные системы важны в работе бизнеса, то обеспечение их работоспособности является важнейшим вопросом. В связи с этим наиболее важные структурные элементы данных систем в обязательном порядке дублируются как по программной, так и по аппаратной части. При этом дублируемые решения не обязательно должны функционировать параллельно, достаточно лишь, чтобы в случае возникновения высокой нагрузки они могли принять на себя ее часть и разгрузить другие компоненты системы [1].

Высокая нагрузка в большинстве случаев не является особенностью реализации конкретной системы, чаще всего это непосредственные условия, в которых данная система функционирует, представляя собой результаты работы в той или иной категории бизнеса. К числу наиболее распространенных причин роста нагрузки на систему в местах их эксплуатации относят:

– рост обращений к системе в рамках функционирования системы класса CRM и ERP, работающей с большим количеством пользователей, или в центрах помощи клиентам и саП-центрах, в которых обрабатывается большое количество звонков и обращений пользователей;

– объем обрабатываемой информации, наблюдаемый в системах мониторинга с большим количеством подключаемого к ним оборудования, системы бизнес-аналитики, а также системы класса CRM и ERP, в которых обрабатываются значительные объемы информации;

– некорректная настройка системы выявляется, как правило, в результате ошибок при написании программного кода либо отсутствия оптимизации, в результате чего возникает рост уровня нагрузки на серверную платформу [12].

В любом случае для того, чтобы устранить возникающую нагрузку на систему, используют несколько подходов к их оптимизации:

– методы, основанные на оптимизации обращений на основании сетевых протоколов и используемые для различного рода сторонних библиотек, процедур кэширования обращений к информационной базе и кэширования ответных данных от сервера;

– методы, направленные на оптимизацию процессов получения информации из базы данных и включающие в себя индексирование базы данных, ее репликацию и секционирование [10].

Рассмотрим более подробно способы оптимизации обращения и получения информации от базы данных.

Работа серверного приложения подразумевает организацию взаимодействия с клиентом, в процессе которого осуществляется информационный обмен. Для этого обязательно должна быть произведена процедура сериализации данных для обеспечения их передачи, а после завершения передачи данные должны быть десериализированы с целью их обработки уже непосредственно внутри системы. Выполнение каждой из указанных двух процедур требует определенных временных затрат, что является одним из оптимальных вариантов [6].

Вторым направлением оптимизации является кэширование обращений к информационной базе. При организации обращений клиента к серверу, на котором размещена информационная база, рано или поздно возникнет ситуация возникновения данных, которые либо изменяются очень редко, либо не изменяются вовсе. С целью минимизации числа обращений к информационной базе следует прибегнуть к использованию кэша информационной системы, который будет выступать в роли хранилища результатов запросов к базе. Для этого чаще всего используют специальные хэш-таблицы, период

обращения к которым чаще всего равен константе. Принцип организации кэширования ответов от веб-сервера является аналогичным.

Одним из наиболее распространенных методов по оптимизации работы с базой данных является ее индексирование – использование специальных механизмов (индексов), которые позволяют ускорить доступ к данным. При этом, помимо ускорения процедур поиска необходимых значений индексы служат для обеспечения поддержки целостности. К издержкам использования индексов необходимо отнести тот факт, что в результате выполнения операций по удалению данных необходимо обязательно произвести перестроение индекса в рамках этой же транзакции. Использование индексов является оправданным в тех случаях, когда в базе данных хранится очень большое число записей, в противном случае обязательная операция перестроения индекса приведет к тому, что общее время выполнения операции будет таким же, как и до добавления индекса.

Следующий вариант оптимизации работы с базой данных – проведение репликации базы данных, который также является распространенным решением, направленным на повышение уровня пропускной способности в совокупности с повышением уровня доступности системы в целом. Репликация реализуется посредством создания нескольких экземпляров одной информационной базы, в которых роли экземпляров разделяются на ведущие и ведомые узлы. Цель работы данных узлов заключается в выполнении операций чтения и возвращения результатов, а также своевременного обновления данных, поступающих от ведущего узла или набора ведущих узлов. Непосредственно ведущий узел выполняет все типы операций, будь то операции чтения либо операции записи, обеспечивая при этом своевременное обновление состава ведомых узлов. На текущий момент времени выделяют несколько наиболее распространенных видов репликации:

1. Синхронная репликация, при реализации которой система ожидает завершение записи информации не только в ведущем узле, но и во всех ведомых узлах, данный вид репликации гарантирует завершение операций, но требует значительного времени ожидания.

2. Асинхронная репликация не требует ожидания завершения процедуры записи на всех ведомых узлах и возвращает системе управление сразу после завершения записи на ведущем узле; преимущество данного метода заключается в более высокой скорости работы, однако он не гарантирует выполнение операции на всех ведомых узлах.

3. Репликация вида «Ведущий-Ведомые» требует обеспечения условия, при котором в системе существует только один ведущий узел, все остальные являются ведомыми. Данный вид репликации является наиболее простым в плане внутренней организации, так как не требует обеспечения синхронизации ведущих узлов: ведущий узел реализует процедуры чтения и записи данных, а ведомые – только чтения. Обновление ведомых узлов происходит через заданные временные промежутки. К числу недостатков данного метода репликации является время для обеспечения синхронизации данных ведущим узлом после синхронизации ведомых. Помимо этого, в случае отказа ведущего узла может произойти потеря данных, которые поступают на него в момент возникновения отказа. Вариантом решения данной проблемы является переназначение роли ведущего узла одному из подчиненных, однако при этом необходимо обеспечить сообщение об изменении статуса бывшему ведущему узлу, так как в противном случае после его восстановления произойдет рассинхронизация данных.

4. Репликация вида «Ведущие-Ведомые», представляющая собой развитие предыдущего вида репликации, подразумевает добавление нескольких ведущих узлов и требует, в первую очередь, решения проблемы синхронизации именно ведущих узлов.

Еще одним вариантом оптимизации работы с базами данных является их секционирование – процедура, в результате выполнения которой база данных разбивается на несколько малых частей с целью повышения пропускной способности, что позволяет добиться равномерного распределения данных между различными секциями во избежание перекоса в работе одного из узлов. Это способствует достижению более высокой скорости обработки данных, а также росту пропускной способности системы. Реализация процедуры секционирования может происходить на основании:

– диапазонов ключа, что является наиболее простым и доступным вариантом, когда разбиение данных происходит на основании определенного ключа, однако основной проблемой в данном случае является выбор оптимального для разбиения ключа;

– с использованием хэш-функции – более эффективный способ, при котором секционирование реализуется специальной хэш-функцией, которая разбивает данные практически в равных пропорциях.

При организации процедуры секционирования требуется периодическое проведение ребалансировки, так как процедура записи для каждой секции выполняется неравномерно, что впоследствии может привести к снижению скорости функционирования системы [11].

Заключение

Целесообразно сделать вывод о том, что для обеспечения процедур оптимизации и достижения более высокого уровня производительности системы возможным является прибегнуть к нескольким различным методам. Выбор наиболее подходящего метода зависит от конкретной ситуации, возможностей разработчиков, а также требований заказчика.

Литература

1. Бабичев С. Л. Распределенные системы: учебное пособие для вузов / С. Л. Бабичев, К. А. Коньков. – М.: Издательство Юрайт, 2023. – 507 с.
2. Гниденко И. Г. Технологии и методы программирования: учебное пособие для вузов / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. – М.: Издательство Юрайт, 2022. – 235 с.
3. Дубовик Е. В. Web на практике. CSS, HTML, JavaScript, MySQL, PHP для funstack-разработчиков / Е.В. Дубовик, А.П. Никольский. – М.: Наука и техника, 2021. – 432 с.
4. Зыков С. В. Программирование: учебник и практикум для вузов / С. В. Зыков. – М.: Издательство Юрайт, 2022. – 320 с.
5. Казарин О. В. Надежность и безопасность программного обеспечения: учебное пособие для вузов / О. В. Казарин, И. Б. Шубинский. – М.: Издательство Юрайт, 2023. – 342 с.
6. Костюк Ю.А. Лекции по основам программирования / Ю.А. Костюк. – Томск: Томский государственный университет, 2022. – 260 с.
7. Лаврищева Е. М. Программная инженерия и технологии программирования сложных систем: учебник для вузов / Е. М. Лаврищева. – 2-е изд., испр. и доп. – М.: Издательство Юрайт, 2022. – 432 с.
8. Лащевски Т. Облачные архитектуры: разработка устойчивых и экономичных облачных приложений / Т. Лащевски, К. Арора, Э. Фарр, П. Зонуз. – СПб: ИД Питер, 2022. – 320 с.
9. Лукьянов П.Б. Разработка и реализация портальных решений / П.Б. Лукьянов. – М.: Прометей, 2020. – 166 с.
10. Роберт М. Чистая архитектура. Искусство разработки программного обеспечения / М. Роберт. – СПб: ИД Питер, 2022. – 352 с.
11. Чернышев С. А. Принципы, паттерны и методологии разработки программного обеспечения: учебное пособие для вузов / С. А. Чернышев. – М.: Издательство Юрайт, 2022. – 176 с.
12. Черпаков И. В. Основы программирования: учебник и практикум для вузов / И. В. Черпаков. – М.: Издательство Юрайт, 2022. – 219 с.

References in Cyrillics

1. Babichev S. L. Raspredelennye sistemy: uchebnoe posobie dlja vuzov / S. L. Babichev, K. A. Kon'kov. – M.: Izdatel'stvo Jurajt, 2023. – 507 s.
2. Gnidenko I. G. Tehnologii i metody programmirovanija: uchebnoe posobie dlja vuzov / I. G. Gnidenko, F. F. Pavlov, D. Ju. Fedorov. – M.: Izdatel'stvo Jurajt, 2022. – 235 s.
3. Dubovik E. V. Web na praktike. CSS, HTML, JavaScript, MySQL, PHP dlja fullstack-razrabotchikov / E.V. Dubovik, A.P. Nikol'skij. – M.: Nauka i tehnika, 2021. – 432 s.
4. Zykov S. V. Programmirovanie: uchebnik i praktikum dlja vuzov / S. V. Zykov. – M.: Izdatel'stvo Jurajt, 2022. – 320 s.
5. Kazarin O. V. Nadezhnost' i bezopasnost' programmnogo obespechenija: uchebnoe posobie dlja vuzov / O. V. Kazarin, I. B. Shubinskij. – M.: Izdatel'stvo Jurajt, 2023. – 342 s.
6. Kostjuk Ju.A. Lekcii po osnovam programmirovanija / Ju.A. Kostjuk. – Tomsk: Tomskij gosudarstvennyj universitet, 2022. – 260 s.
7. Lavrishheva E. M. Programmnaja inzhenerija i tehnologii programmirovanija slozhnyh sistem: uchebnik dlja vuzov / E. M. Lavrishheva. – 2- e izd., ispr. i dop. – M.: Izdatel'stvo Jurajt, 2022. – 432 s.
8. Lashhevski T.Oblachnye arhitektury: razrabotka ustojchivyh i jekonomichnyh oblachnyh prilozhenij / T. Lashhevski, K. Arora, Je. Farr, P. Zonuz. – SPb: ID Piter, 2022. – 320 s.
9. Luk'janov P.B. Razrabotka i realizacija portal'nyh reshenij / P.B. Luk'janov. – M.: Prometej, 2020. – 166 s.
10. Robert M. Chistaja arhitektura. Iskusstvo razrabotki programmnogo obespechenija / M. Robert. – SPb: ID Piter, 2022. – 352 s.
11. Chernyshev S. A. Principy, patterny i metodologiirazrabotki programmnogo obespechenija: uchebnoe posobie dlja vuzov / S. A. Chernyshev. – M.: Izdatel'stvo Jurajt, 2022. – 176 s.
12. Cherpakov I. V. Osnovy programmirovanija: uchebnik i praktikum dlja vuzov / I. V. Cherpakov. – M.: Izdatel'stvo Jurajt, 2022. – 219 s.L

Харазян Айк Арменович, ведущий разработчик,
Высшая школа экономики, Москва, Россия
haykking@gmail.com

Ключевые слова

высоконагруженная система, распределенные приложения, оптимизация высоконагруженных систем, высокая нагрузка, оптимизация программного обеспечения.

Hayk Kharazyan ANALYSIS OF METHODS AND TOOLS FOR OPTIMIZING THE FUNCTIONING OF HIGHLY LOADED SYSTEMS

Keywords
high-load system, distributed applications, optimization of highload systems, high load, software optimization.

DOI: 10.34706/DE-2023-02-11
JEL classification: C61 – методы оптимизации, модели программирования, динамический анализ

Abstract
In this article the methods and tools for optimizing the operation of highly loaded systems are considered. The description of the concept of a highly loaded system, the main reasons for the increase in load are given. In particular, the need to implement an individual implementation of the system was presented depending on the requirements and functional tasks for it, as well as the dependence of the load on the system on the level of its scaling, the number of users working with it, and the amount of data being processed. In addition, the most common ways to optimize the operation of highly loaded systems are presented and conclusions are given.
.