DOI 10.31891/2307-5732-2023-319-1-189-201
УДК 004.7:519

ЛЕМЕШКО АНДРІЙ
Державний університет телекомунікацій
https://orcid.org/0000-0001-8003-3168
АНТОНЕНКО АРТЕМ
Державний університет телекомунікацій
https://orcid.org/0000-0001-9397-1209
e-mail: artem.v.antonenko@gmail.com
ДОБРОВОЛЬСЬКИЙ ОЛЕКСІЙ
Державний університет телекомунікацій
https://orcid.org/0009-0007-9995-0629
ЗАВАДСЬКИЙ В'ЯЧЕСЛАВ
Державний університет телекомунікацій
https://orcid.org/0009-0009-5167-1051

ОПТИМІЗАЦІЯ ЧАСУ РЕАГУВАННЯ ВЕБ СЕРВЕРУ APACHE

У цій статті досліджуються підходи до онлайн-оптимізації веб-сервер Apache, зосереджуючись на параметрі MaxClients (який контролює максимальну кількість працівників). Використовуючи як емпіричні, так і аналітичні методи, ми показуємо, що MaxClients має увігнутий висхідний вплив на час відгуку, і, отже, методи підйому на пагорб можна використовувати для визначення оптимального значення MaxClients. Ми досліджуємо два оптимізатори, які використовують підйом на гору — один на основі методу Ньютона, а другий — на основі нечіткого керування. Третя техніка – це евристика, яка використовує зв'язок між використанням вузьких місць і мінімізацією часу відгуку. У всіх випадках онлайн-оптимізація скорочує час відповіді в 10 або більше разів порівняно зі статичним значенням за замовчуванням. Компроміси між онлайн-схемами такі. Метод Ньютона добре відомий, але не дає узгоджених результатів для дуже мінливих даних, таких як час відгуку. Нечітке керування більш надійне, але сходиться повільно. Евристика добре працює в нашій системі-прототипі, але її може бути важко узагальнити, оскільки вона вимагає знання про вузькі місця ресурсів і вміння вимірювати їх використання. Тематика є актуальною в контексті швидко зростаючої кількості веб-додатків та високих вимог до продуктивності та ефективності роботи веб-серверів. Дана тема досліджує можливості покращення швидкості та часу відповіді Apache веб-серверу за допомогою різноманітних технік та налаштувань, таких як оптимізація налаштувань сервера, використання кешування, стиснення даних, оптимізація маршрутизації запитів та інших. Метою даного дослідження є покращення продуктивності та швидкості відповіді Apache веб-серверу, що може бути корисним для розробників та адміністраторів веб-додатків та сервісів. Швидкість та час відповіді веб-серверів є критичними факторами для задоволення потреб користувачів та досягнення бізнес-цілей веб-додатків та сервісів. Apache є одним з найпоширеніших веб-серверів у світі, тому оптимізація часу відповіді Apache серверу є важливим завданням для багатьох розробників та адміністраторів веб-додатків та сервісів. В даному дослідженні будуть розглянуті різні підходи та техніки для оптимізації часу відповіді Apache веб-серверу, зокрема налаштування параметрів сервера, використання кешування, стиснення даних, оптимізація маршрутизації запитів та інші. Результати дослідження можуть бути корисними для розробників та адміністраторів веб-додатків та сервісів, які мають справу з використанням Apache веб-серверу. Оптимізація часу відповіді Apache серверу може значно поліпшити продуктивність та ефективність веб-додатків та сервісів, що в свою чергу може призвести до задоволення користувачів та досягнення бізнес-цілей.

LEMESHKO ANDRII, ANTONENKO ARTEM, DOBROVOLSKYI OLEKSII, ZAVADSKYI VIACHESLAV
State University of Telecommunications

APACHE WEB SERVER RESPONSE TIME OPTIMIZATION

This article explores approaches to online optimization of the Apache web server, focusing on the MaxClients parameter (which controls the maximum number of workers). Using both empirical and analytical methods, we show that MaxClients has a concave-upward effect on response time, and hence hill-climbing techniques can be used to determine the optimal value of MaxClients. We investigate two optimizers that use hill climbing—one based on Newton's method and one based on fuzzy control. A third technique is a heuristic that exploits the relationship between bottleneck utilization and response time minimization. In all cases, online optimization reduces the response time by a factor of 10 or more compared to the static default value. The trade-offs between online schemes are as follows. Newton's method is well known but does not provide consistent results for highly variable data such as response times. Fuzzy control is more reliable but converges slowly. The heuristic works well in our prototype system, but it can be difficult to generalize because it requires knowledge of resource bottlenecks and the ability to measure their use. This topic is relevant in the context of a rapidly growing number of web applications and high requirements for the performance and efficiency of web servers. This topic explores how to improve the speed and response time of the Apache web server using various techniques and settings, such as optimizing server settings, using caching, data compression, optimizing request routing, and more. The purpose of this research is to improve the performance and response speed of the Apache web server, which can be useful for developers and administrators of web applications and services. The speed and response time of web servers are critical factors in meeting user needs and achieving business goals for web applications and services. Apache is one of the most widely used web servers in the world, so optimizing the response time of the Apache server is an important task for many developers and administrators of web applications and services. This study will examine various approaches and techniques for optimizing the response time of the Apache web server, including configuring server parameters, using caching, data compression, optimizing request routing, and others. The results of the study can be useful for developers and administrators of web applications and services that deal with the use of the Apache web server. Optimizing Apache server response time can significantly improve the performance and efficiency of web applications and services, which in turn can lead to user satisfaction and business goals.

Постановка проблеми

Проблема полягає в тому, що час відповіді Apache Веб-сервера може бути досить великим в залежності від навантаження та кількості запитів, що надходять до сервера. Це може призвести до погіршення користувацького досвіду та негативно впливати на продуктивність веб-сайту. Оптимізація часу відповіді Apache Веб-сервера може включати різноманітні стратегії, такі як кешування статичних файлів, налаштування рівня журналювання, покращення обробки запитів з використанням різних алгоритмів, налаштування опцій конфігурації сервера та оптимізація бази даних, якщо вона використовується. Результатом оптимізації може бути зменшення часу відповіді сервера та збільшення швидкодії сайту, що може позитивно вплинути на користувацький досвід та рейтинг сайту в пошукових системах.

Аналіз останніх джерел

Метою оптимізації часу відгуку веб-сервера Apache є підвищення продуктивності та ефективності веб-додатків і сервісів, які використовують цей веб-сервер. Для досягнення цієї мети слід розглянути різні підходи і методи оптимізації часу відгуку веб-серверів Apache, такі як конфігурація сервера, використання кешування, стиснення даних, оптимізація маршрутизації запитів і т.д. Оптимізація часу відгуку сервера Apache може значно підвищити продуктивність і ефективність роботи веб-додатків і сервісів, що призведе до задоволення потреб користувачів і досягнення бізнес-цілей.

Об'єктами дослідження для оптимізації часу відгуку веб-сервера Apache є сам веб-сервер, його конфігураційні параметри та взаємодія між клієнтами і серверними додатками. Дослідження повинно охоплювати такі параметри веб-сервера, як налаштування мережі, конфігурацію операційної системи, налаштування серверних програм, обробку запитів та відправку відповідей.

Предметом дослідження в роботі, присвяченій оптимізації часу відгуку веб-сервера Apache, є процес, за допомогою якого сервер відповідає на запити клієнтів, та його параметри. Завданнями дослідження є швидкість обробки запитів, час відгуку, кількість запитів за одиницю часу, кількість одночасних з'єднань та інші параметри, які можуть впливати на час відгуку сервера.

Виклад основного матеріалу

З поширенням систем електронної комерції якість обслуговування, зокрема час відгуку, привертає все більшу увагу. Однією з проблем у цьому контексті є адаптація систем до мінливих робочих навантажень шляхом їхньої оптимізації через онлайн-конфігурацію. У цій статті розглядаються такі підходи до онлайн-оптимізації на веб-серверах Apache, з акцентом на методах, які є менш інвазивними і можуть бути застосовані до широкого спектру конфігурацій і систем.

Розглянемо параметр Apache MaxClients, який визначає кількість запитів, що паралельно обробляються веб-сервером. У таблиці 1 наведено середній час відгуку, виміряний з різними конфігураціями MaxClients для різних робочих навантажень. Тестове середовище, яке використовувалося для збору цих даних, буде описано далі в цьому розділі.

Таблиця 1

Час відгуку для різних навантажень, сек.

MaxClients	Workload	
	Dynamic	Dynamic + Static
150	50	
650	1	15
900	30	2

Часто виявляється, що оптимальне значення MaxClients змінюється залежно від типу сторінок, які відвідує сайт. Оскільки реальні робочі навантаження можуть швидко змінюватися, оптимізація цих важливих параметрів в режимі онлайн може значно їх покращити. [1,2]

У цій статті описано загальний підхід до онлайн-оптимізації часу відгуку широко використовуваного веб-сервера Apache. Суміжною областю досліджень є диференційовані сервіси, які спрямовані на досягнення цілей часу відгуку для різних класів завдань. Автори використовують аналогово-інтегральний контролер для налаштування та диференціації часу відгуку. Для регулювання використання процесора і пам'яті сервера в межах заданого значення QoS використовується схема контролера з декількома входами і декількома виходами, а також описано підхід, який поєднує теорію черг і теорію управління для регулювання часу відгуку. На жаль, проблема управління, яку вирішує цей підхід, дуже відрізняється від проблеми оптимізації. [3,4] По суті, регулювання (наприклад, забезпечення цільових значень часу відгуку для золотих і срібних послуг) визначає, як "розрізати пиріг", а оптимізація (наприклад, мінімізація часу відгуку для класу послуг) - це те, чим вона займається. Було проведено кілька досліджень в області оптимізації онлайн-ресурсів в обчислювальних системах. Це дослідження описує реалізацію Apache, яка керує ресурсами веб-сервера на основі максимізації прибутку (наприклад, відповідь протягом 8 секунд, щоб не відштовхувати користувачів). Хоча результати є цікавими, цей підхід вимагає значних змін у схемі управління ресурсами Apache. Цей підхід враховує максимізацію вигоди в угодах про рівень обслуговування для ферм веб-серверів, але таким чином, що базується на точній аналітичній моделі

керованої системи. Нещодавно був запропонований нечіткий підхід до управління, що дозволяє мінімізувати час відгуку шляхом поєднання схеми управління зі зворотним зв'язком з якісним аналізом впливу параметрів конфігурації на QoS. На жаль, цей підхід має довгий час збіжності. [6]

На рисунку 1 показано запропоновану нами архітектуру. Цільова система (наприклад, Apache) надає один або декілька параметрів конфігурації (наприклад, MaxClients), які динамічно змінюються оптимізатором для оптимізації вимірюваної змінної (наприклад, часу відгуку). По-перше, показується, що MaxClients має висхідний ефект на час відгуку, і тому для визначення оптимального значення MaxClients можна використовувати метод сходження на гору. Досліджуються два оптимізатори, що використовують метод сходження на гору, один з яких базується на методі Ньютона, а інший - на нечіткому управлінні.

Третій метод - це евристичний метод, який використовує спостережуваний взаємозв'язок між завантаженням вузьких місць і мінімізацією часу відгуку. Метод Ньютона працює краще, ніж стандартний метод Apache, але дає непослідовні результати через різний час відгуку. Нечіткий контроль є більш надійним, але збігається повільно. Евристичні методи добре працюють в нашій прототипній системі, але їх важко узагальнити, оскільки вони вимагають знання вузьких місць і вміння вимірювати використання ресурсів. [2]

Apache зазвичай структурується як сукупність робітників, що обробляють HTTP-запити. У нашому дослідженні використовується версія 1.3.19, де працівники є процесами, але ми вважаємо, що основна ідея може бути застосована в широкому сенсі.

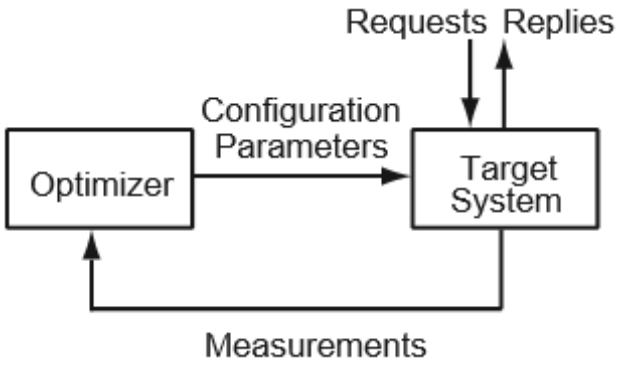

Рис.1 Загальна архітектура для онлайн оптимізації. Цільова система контролюється параметрами конфігурації, які динамічно змінюються оптимізатором у відповідь на зміну робочого навантаження

Потік запитів в Apache показано на рисунку 2. Запит потрапляє до черги прийняття TCP, де він чекає на працівника. Працівники обробляють запит до його завершення, після чого приймають новий запит. Кількість робочих процесів обмежується параметром MaxClients. [5]

Багато висновків у цій статті ґрунтуються на результатах експериментів. Всі експерименти проводилися на сервері Pentium III 600 МГц з 256 МБ оперативної пам'яті та серверним програмним забезпеченням Apache 1.3.19 з Linux 2.4.7 на тій самій машині, підключений до локальної мережі зі швидкістю 100 Мбіт/с; використовувався генератор синтетичного робочого навантаження, що працював на тій самій машині. Розподіл файлів за розмірами такий самий, як і у Webstone 2.5; використовувалися статичні та динамічні навантаження. Запити до динамічних сторінок оброблялися CGI-скриптами у Webstone 2.5. Для пояснення того, як генеруються запити, потрібні додаткові деталі.

Наша модель робочого навантаження представлена у WAGON. [9]

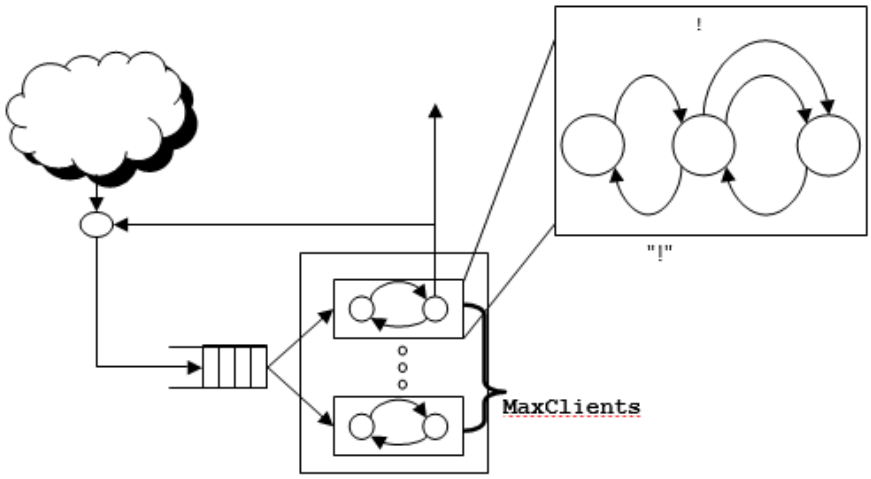

Рис. 2. Архітектура Apache та потік сеансу

Для того, щоб пояснити, як подати запит, потрібні додаткові деталі. Наша модель робочого навантаження базується на моделі WAGON, яка, як було доведено, може бути застосована до широкого спектру веб-запитів. Ця модель організовує робоче навантаження в сеанси (що представляють собою серію взаємодій користувача). Як показано на рисунку 2, сесія складається з декількох запитів до сторінок. Сторінка містить ряд вбудованих об'єктів, параметри яких визначаються довжиною пакета. Отже, параметрами навантаження є частота надходження сеансів, тривалість сеансу (кількість кліків або запитів сторінок у сеансі), довжина пакета (кількість об'єктів у пакеті) та час міркувань (час між послідовними кліками). Таблиця 2 підсумовує параметри, використані в цій роботі, і базується на даних, наданих загальнодоступним веб-сайтом, на основі синтетичного блогу, створеного з використанням моделі WAGON і з використанням httperf для здійснення HTTP/1.1 запитів.[10,11]

Таблиця 2

Параметри робочого навантаження

Parameter Nome	Distribution	Parameters
Session Rate	Exponential	Mean = 0.1
Session Length	Log Normal	Mean = 8, Ʊ = 3
Burst Length	Gaussian	Mean = 7, Ʊ = 3
Think Time (s)	Log Normal	Mean = 30, Ʊ = 30

Показник, який ми вирішили мінімізувати, - це час відгуку на стороні сервера. Вимірювання часу відгуку на стороні клієнта є найбільш значущою метрикою для користувачів, але ця інформація зазвичай недоступна в режимі реального часу на веб-сервері. Крім того, хоча час відгуку на стороні клієнта можна приблизно визначити на основі вимірювань на стороні сервера і моделей TCP, на стороні сервера можна перевірити тільки час відгуку на стороні сервера.[12]

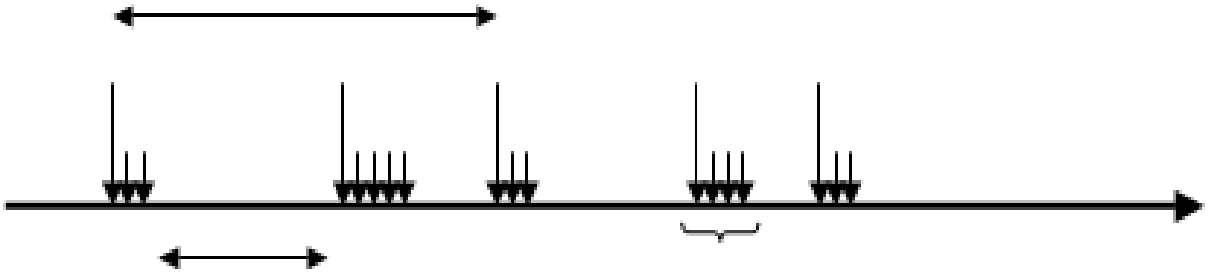

Рис. 3. Елементи моделі робочого навантаження WAGON
(Суцільні та пунктирні лінії позначають два сеанси. Довга стрілка вказує на HTML-текст веб-сторінки, тоді як коротка стрілка вказує на запит до об'єкта веб-сторінки).

У цій статті розглядається час відгуку на стороні сервера, а саме час відгуку сторінки (RT). Цей показник необхідно оцінити, оскільки доставка сторінки може включати декілька запитів. Для цього використовується наступна формула:

$$RT=AQT+BL^{x}ST \quad (1)$$

Час черги прийняття (AQT) походить з ядра Linux і додає інструмент для вимірювання середнього часу очікування з'єднань, що потрапляють в чергу прийняття протягом певного періоду часу. Час обслуговування (Service Time, ST) вимірюється шляхом вимірювання першого та останнього кроків циклу обробки запитів Apache (тобто, момент отримання запиту та відправлення відповіді). Середня кількість вбудованих запитів на сторінку, також відома як довжина пакета, позначається як BL. Її можна обчислити як кількість запитів, оброблених у всіх робочих процесах, поділену на кількість з'єднань, оброблених у черзі хоста TCP, оскільки через постійність з'єднань HTTP/1.1 існуюче TCP-з'єднання залишається відкритим між послідовними HTTP-запитами в пакеті, тільки перший запит повинен встановити TCP-з'єднання і потрапити в чергу хоста TCP. Це призводить до наведеного вище рівняння.

На рисунку 6 показано експериментальні результати Apache з різними налаштуваннями MaxClients. Колом показано середній час відгуку, а вертикальною лінією - середньоквадратичне відхилення. Зверніть увагу, що кругова лінія виглядає ввігнутою у верхній частині цієї кривої. Також ця крива показує, що MaxClients має більш ніж 10-кратний вплив на час відгуку для цього робочого навантаження.

Цю увігнутість можна пояснити з точки зору архітектури Apache: якщо значення MaxClients занадто низьке, буде велика затримка через очікування в черзі отримання TCP. Фактично, черга може переповнитися і запити можуть бути відкинуті. З іншого боку, якщо значення MaxClients занадто велике, ресурси будуть

перевантажені, що також знижує продуктивність. В екстремальних випадках перевищення ліміту кількості процесів може призвести до внутрішніх помилок сервера. Загальним результатом цих факторів є те, що час відгуку є увігнутою вгору функцією MaxClients. По суті, робочий процес можна розглядати як логічні ресурси, необхідні для обробки запиту. Однак для цього потрібні фізичні ресурси, такі як процесор, пам'ять та пропускна здатність вводу/виводу.

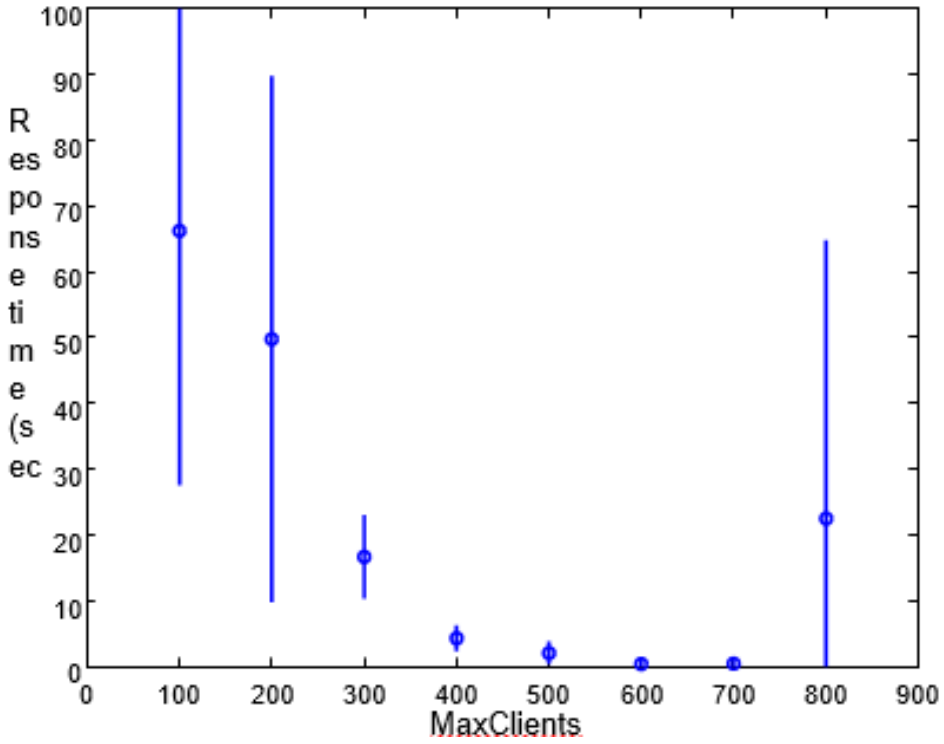

Рис. 4. Вплив MaxClients на час відгуку

Оскільки нас цікавить онлайн-оптимізація, нам потрібно динамічно змінювати MaxClients. Для цього було реалізовано механізм, схожий на Graceful Restart. Повідомляється, що цей механізм дозволяє змінювати MaxClients без зупинки Apache. Для цього потрібен агент в системі Apache. Цей агент також надає такі інструменти, як використання процесора, використання пам'яті та час відгуку на стороні сервера.

Хоча ця стаття зосереджена на MaxClients, ми вважаємо, що використаний підхід має більш широке застосування. Наприклад, зараз ми досліджуємо KeepAliveTimeOut, ще один параметр Apache, який визначає час, протягом якого сеанс може залишатися неактивним, коли використовується постійне з'єднання. Інші системи мають конфігураційні параметри, подібні до MaxClients, такі як кількість потоків сервлетів або потоків EJB на сервері додатків. [13]

Онлайн-оптимізація описує, як можна мінімізувати час відгуку шляхом динамічної адаптації MaxClients, виходячи з того, що час відгуку є увігнутим MaxClients. Розглянуто кілька методів, включаючи метод Ньютона, нечіткий контроль та евристичні методи. Ці методи порівнюються з точки зору мінімізації часу відгуку та швидкості збіжності до фіксованого значення. Перше є бажаним з точки зору покращення якості обслуговування. Другий важливий для адаптації до змін навантаження. Обидва вищезгадані підходи передбачають зворотній зв'язок. Тому MaxClients коригується на основі спостережуваного впливу на час відгуку або інші показники. Альтернативно можна використовувати метод прямого зворотного зв'язку, коли оптимальне значення MaxClients розраховується на основі аналітичної моделі. Цей метод прямого зворотного зв'язку є привабливим, оскільки він дозволяє уникнути проблем, пов'язаних зі стабільністю та швидкістю збіжності. Однак цей метод вимагає аналітичної моделі з вхідними даними, які а) відстежують виміряний час відгуку і б) можуть бути легко оцінені. Модель, розроблен, задовольняє пункт (а), але не (б). Наприклад, швидкість обслуговування μm залежить від кількості серверів, тобто μm = f(m), але ми не знаємо цієї функції. Можна розробити модель, яка задовольняє (а) і (b), але за відсутності такої моделі ми звернемося до підходу зі зворотним зв'язком. [16,17]

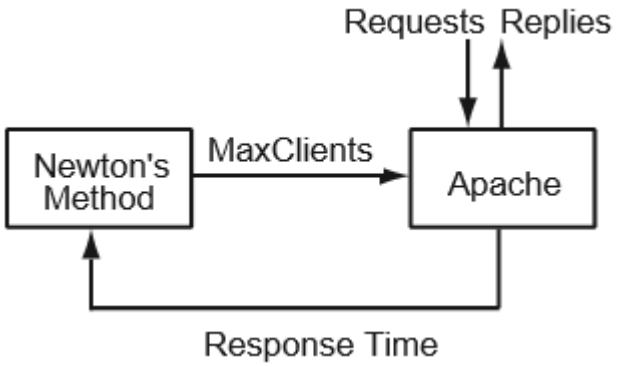

Рис. 5. Архітектура онлайн-оптимізації Apache з використанням методу Ньютона для динамічного налаштування MaxClients на основі вимірювань часу відповіді

На рисунку 5 показано архітектуру, де метод Ньютона використовується для оптимізації під час перегляду Apache в режимі онлайн за допомогою динамічної адаптації MaxClients.[14] Метод Ньютона - це широко використовувана техніка оптимізації, яка використовує нахил функцій, що мінімізуються (наприклад, рисунок 4), для оцінки значення MaxClients, яке мінімізує час відгуку. Наприклад, якщо y - час відгуку, *a x* - MaxClients, можна використати апроксимацію *y = f(x) ≈ a(x-x*)2+b*. Метод Ньютона представлений наступним рівнем.

$$x_{k+1} = x_k - (\bigtriangledown^2 f(x_k))^{-1} \bigtriangledown f(x_k) \tag{2}$$

де *xk* - значення x у дискретний момент часу *k*. Рівняння (3) починається з початкового значення *x0* при *k= 0*.

Його нахил (друга часткова похідна *f(x)*) обчислюється в точці *xk*; його від'ємне значення вказує на більш крутий напрямок спуску). Значення (другої частинної похідної *f(x)*) вказує на розмір кроку оновлення; введення частинних похідних другого порядку усуває локальний лінійний пошук і може призвести до швидшої збіжності. Однак цей алгоритм більш чутливий до шумів вимірювань.

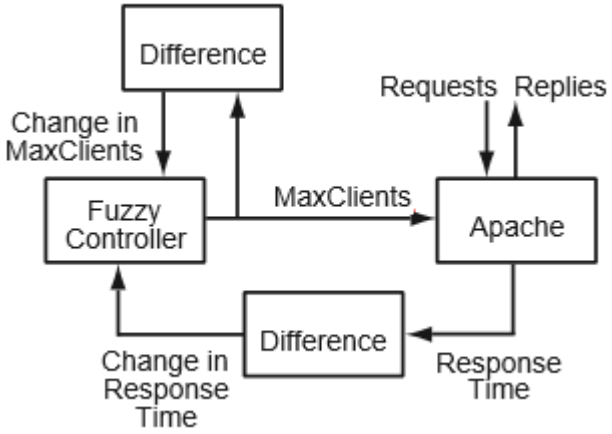

Рис. 6. Архітектура онлайн-оптимізації Apache з використанням Fuzzy Control для динамічного налаштування MaxClients на основі змін MaxClients і часу відповіді

Нечітке керування - це ще один підхід до онлайн-оптимізації. У цьому дослідженні розглядається саме цей підхід. На рисунку 6 показано архітектуру, яка використовує нечіткий контроль для оптимізації під час огляду Apache онлайн шляхом динамічної адаптації MaxClients. Нечіткий контролер використовує зміни в MaxClients і час перегляду для динамічної оптимізації MaxClients.

Поведінка нечіткого контролера керується набором правил IF-THEN. Наприклад, "ЯКЩО зміна MaxClients невелика і зміна часу відгуку незначна, ТО наступна зміна MaxClients незначна". Терміни changein-MaxClients і changein-response-time є лінгвістичними змінними, а neglarge - лінгвістичними значеннями. Лінгвістичні зміни - це природний спосіб впоратися з невизначеністю, яка створює теорію ймовірності в умовах комп'ютерних систем. Лінгвістичні змінні країни у формі, яка відповідає кількості змінних. Системи нечіткого керування забезпечують метод узгодження між числовими та лінгвістичними змінами (так звана нечітка фазифікація та розгортання). Для отримання додаткової інформації про нечітке керування див. Нечітке керування. [15,18]

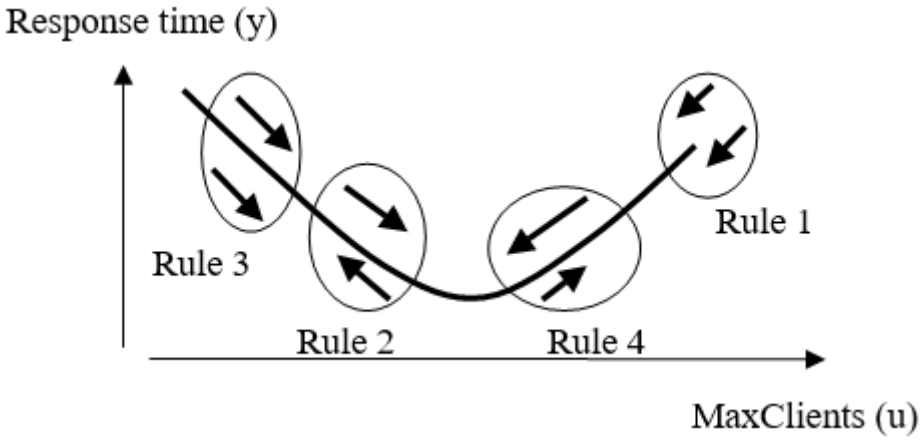

Рис. 7. Ілюстрація нечітких правил

Таблиця 3

Нечіткі правила

Rule	**IF**	**THEN**
	change in Max Clients AND change in Response Time	change in next Max Cleints
1	neglarge AND neglarge	neglarge
2	neglarge AND poslarge	poslarge
3	poslarge AND neglarge	poslarge
4	poslarge AND poslarge	poslarge

Оптимізаційну задачу, що розглядається, можна легко виразити у вигляді нечітких правил. Правила в Таблиці 3 організовані наступним чином (також показано на Рисунку 7, де пунктирна стрілка вказує на передумову IF, а суцільна стрілка вказує на наступну частину THEN): частина IF змінює позицію на кривий час огляду для оптимального значення MaxClients. Наприклад, правило 4 розглядає випадок, коли MaxClients збільшується, що збільшує час перегляду. Це означає, що ви знаходитесь праворуч від оптимального значення MaxClients; частина THEN показує, як ви змінюєте MaxClients, де neglarge - це зменшення, а poslarge - збільшення. Правила 1 і 3 вказують на позицію, коли час відгуку зменшується в результаті останньої зміни MaxClients у правильному напрямку. На противагу цьому, правила 2 і 4 стосуються ситуацій, коли час відгуку збільшився в результаті останньої операції, тобто "неправильної операції": кількість змін MaxClients змінюється швидкістю збіжності та ступеня коливання стаціонарного стану.

Очевидно, що якщо крива крутіша, то невеликі зміни в MaxClients є оптимальними. Для більш поступових градієнтів краще використовувати значні зміни. Евристичні методи оптимізації на основі насичення є стійкими до шуму та специфічні функції, що оптимізуються, і керуються прагненням до швидкої збіжності. Зі збільшенням MaxClients і завантаженням процесора на 100% час відгуку мінімізується. Це видно зі статичних і динамічних вимірювань навантаження на рисунках 10 і 11, де середній час очікування в черзі і час обслуговування виміряно для різних значень MaxClients, а час перегляду розраховано за допомогою рівня (1). Для моніторингу використання системних ресурсів також було виміряно використання процесора та пам'яті. на рис. 10 годин перегляду зменшується при збільшенні MaxClients з 200 до 480, після чого завантаження процесора становить приблизно 100%. на рис. 11 цей стан насичення настає, коли MaxClients досягає приблизно 800. [19,20]

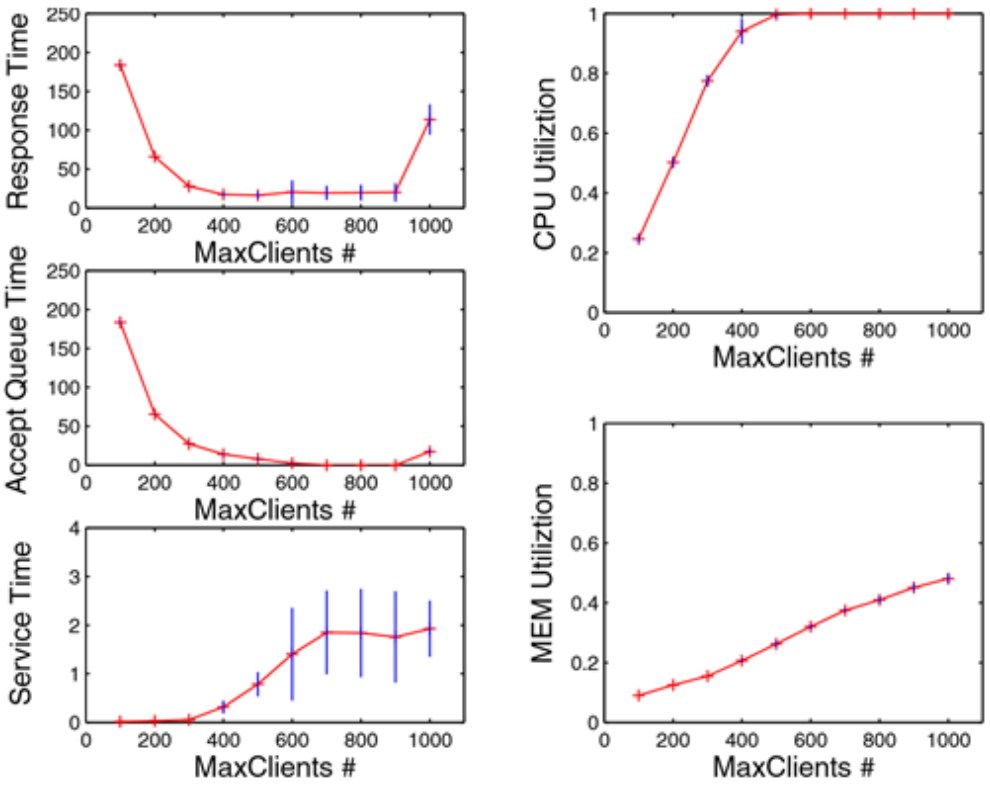

Рис. 8. Вимірювання Apache для динамічного навантаження

Наша інтуїція щодо того, чому це відбувається, полягає в наступному: MaxClients шукає набір логічних ресурсів, які називаються працівниками Apache. Ці логічні ресурси поділяються на фізичні ресурси, такі як процесор, пам'ять і пропускну здатність уводу/виводу. Збільшуючи MaxClients до тих пір, поки фізичні ресурси не будуть насичені, більше логічних ресурсів можуть працювати паралельно.

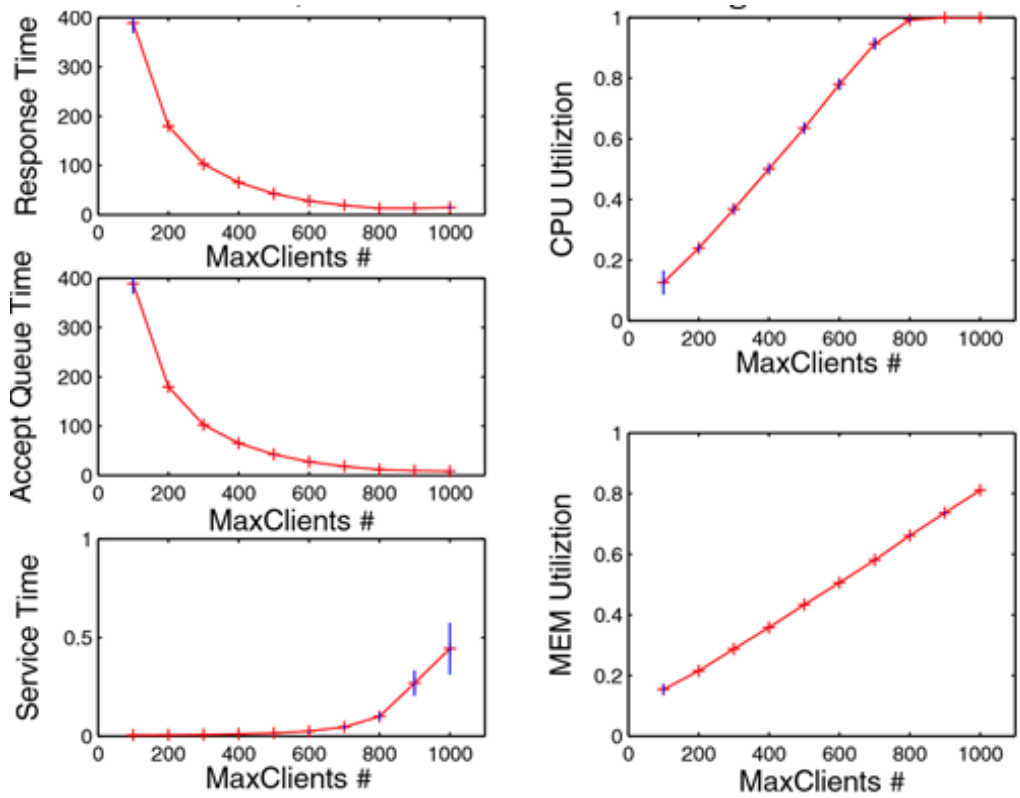

Рис. 9. Вимірювання Apache для статичного навантаження

Однак, як тільки фізичний ресурс насичується, подальше збільшення логічного ресурсу не відбувається збільшення паралелізму. Натомість таке збільшення додає накладні витрати (наприклад, через перемикання процесів).

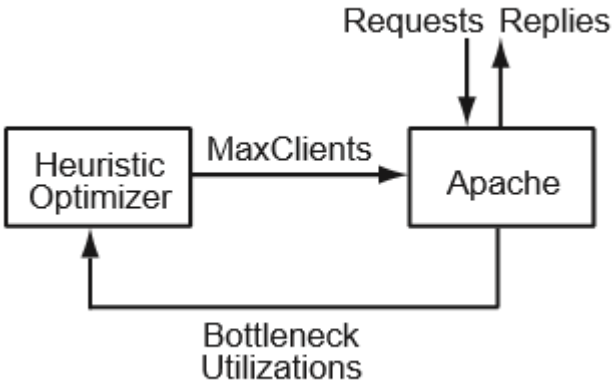

Рис. 10. Архітектура онлайн-оптимізації Apache з використанням евристики на основі насиченості для динамічного коригування MaxClients та змін у використанні вузьких місць

Експериментальні результати порівнює методи онлайн-оптимізації з точки зору мінімального значення досягнутого часу відповіді, швидкості конвергенції та стійкості. На рисунку 11 порівнюється продуктивність методу Ньютона зі схемою Apache за замовчуванням. Фігура містить три підфігури, кожна з яких має два сюжети. На кожному рисунку верхній графік показує траєкторію MaxClients, а нижній — відповідний час відповіді. Зверніть увагу, що метод Ньютона справді покращує час відповіді порівняно зі стандартною схемою Apache. Однак через мінливість часу відгуку різні прогони методу Ньютона можуть дати дуже різні результати. Це пояснюється тим, що отримання матриці Гессе вимагає трьох зразків для обчислення другої похідної на кожному кроці алгоритму. Це збільшує час конвергенції, а також алгоритм є більш чутливим до шуму при вимірюванні часу відгуку. На жаль, час відгуку зазвичай досить шумний, якщо він не усереднений для багатьох зразків (те, що зменшує швидкість, з якою контролер може реагувати на зміни робочого навантаження). Через таку чутливість до шуму ми не розглядаємо метод Ньютона в інших порівняннях.

Далі ми порівнюємо типову схему Apache з нечітким керуванням і евристичним методом, представленим раніше. На рисунку 13 показано результати для динамічного навантаження. (Результати структуровані так само, як на рисунку 11, 12, 13.) Ми бачимо, що евристика збігається зі своїм значенням MaxClients через 2 хвилини. Для нечіткого керування конвергенція займає приблизно 14 хвилин. З іншого боку, нечітке керування досягає меншого часу відгуку. На рисунку 14 показано результати для статичного навантаження. Знову ж таки, евристика сходиться швидше, ніж нечітке керування.

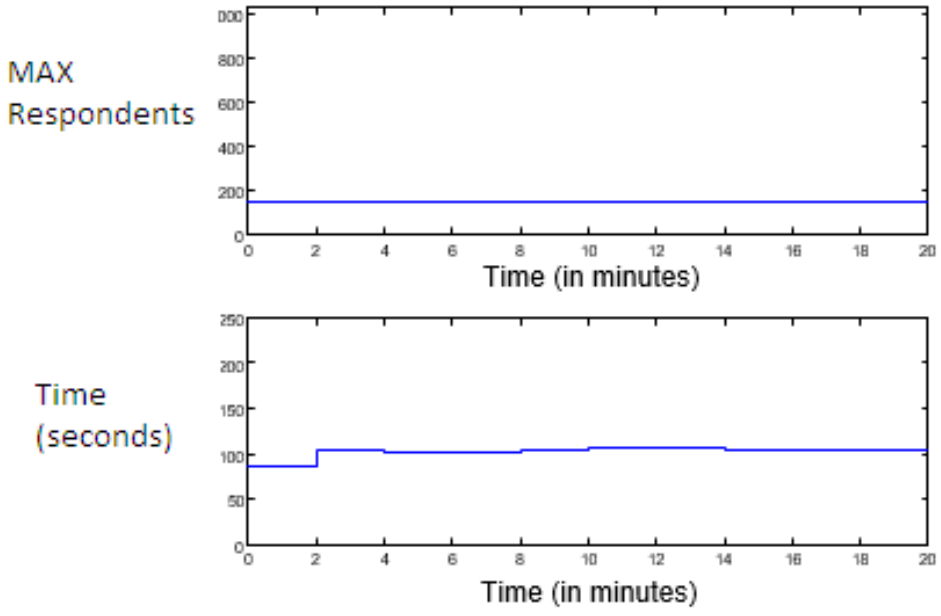

Рис. 11. Типова схема керування Apache

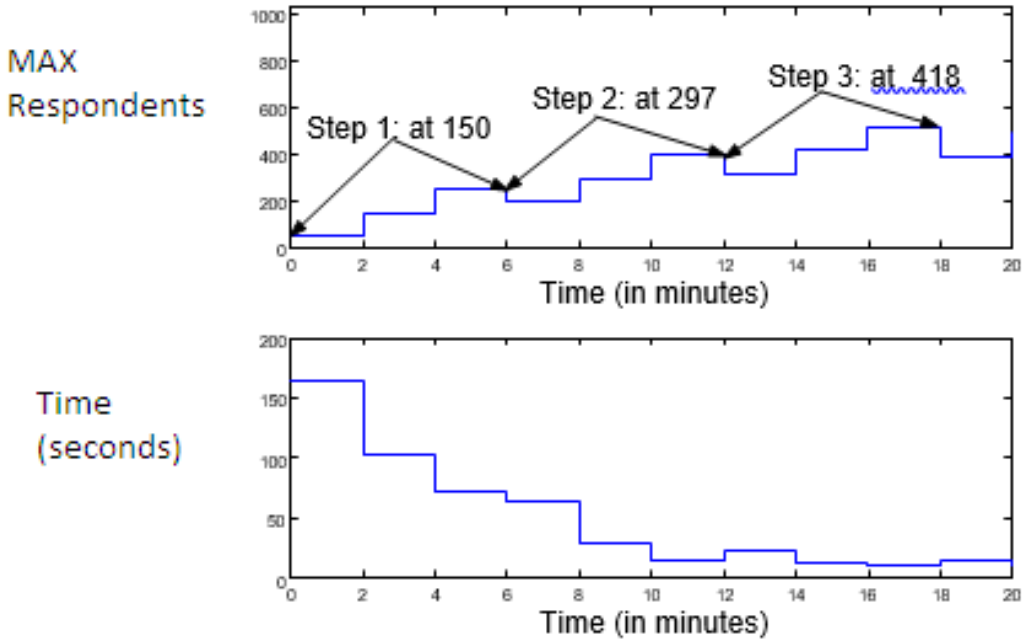

Рис. 12. Перший запуск

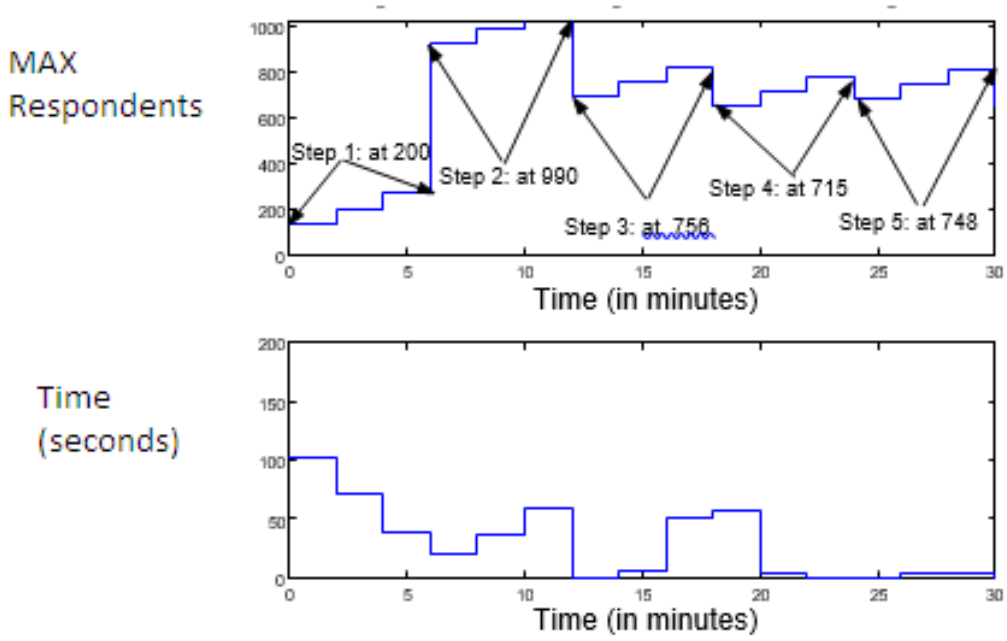

Рис. 13. Другий запуск

Порівняння стандартного методу Apache та методу Ньютона при динамічному навантаженні; метод Ньютона, безумовно, досягає меншого часу відгуку, але його поведінка є непослідовною через мінливість часу відгуку. Тут, однак, стаціонарний час відгуку, досягнутий евристичним методом, майже дорівнює часу відгуку, досягнутому не чітким керуванням.

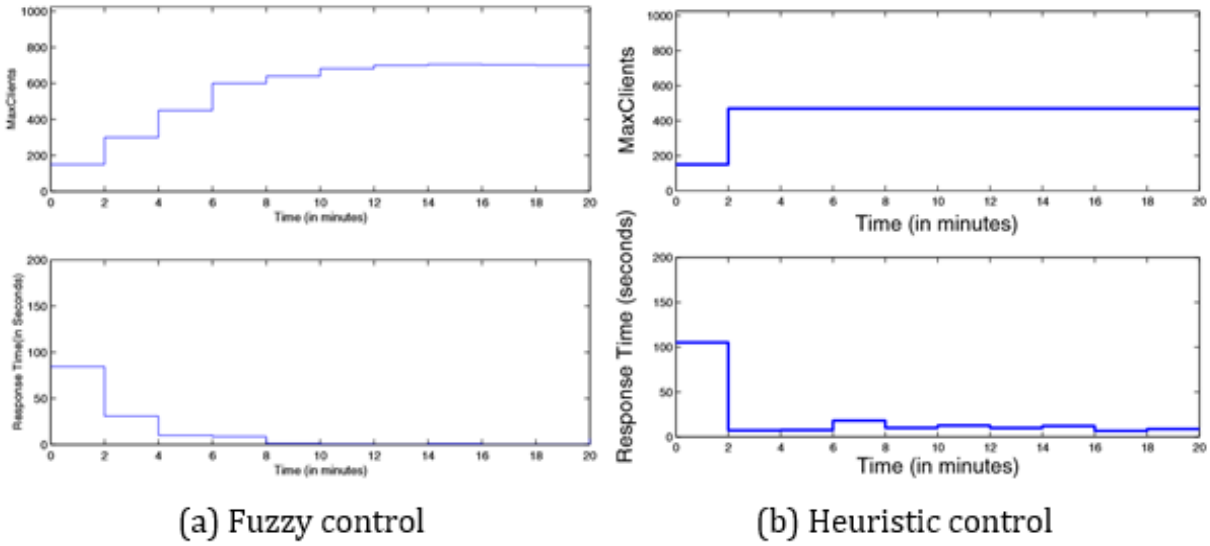

Рис. 14. Порівняння продуктивності схем онлайн-оптимізації при динамічному навантаженні

У таблиці 4 наведено якісне порівняння чотирьох сформованих систем. Система Apache обробляє точки мінімізації зору за час відгуку головним чином тому, що вона не була розроблена для цієї мети. Ньютонівський метод є кращим у цьому відношенні, але збігається повно і має погану завадостійкість. Нечіткий контроль є дуже стійким, після чого він мало припущений, але збігається повно. Наші евристичні методи забезпечують хорошу оптимізацію та швидку збіжність, але припущення про перевантаження ресурсів не завжди справджуються.

Таблиця 4

Якісні порівняння методик

	Optimization	**Speed**	**Robustness**
Default Apache	Poor	Fast	Good
Newton's Method	Fair	Slow	Poor
Fuzzy Control	Good	Slow	Good
Heuristic	Good	Fast	Fair

Висновки

У цій статті досліджено підхід до веб-оптимізації параметрів конфігурації веб-сервера Apache, зосереджуючись на методах, які є менш інвазивними і можуть бути застосовані до широкого спектру параметрів і систем. Ми зосереджуємося на параметрі MaxClients, який контролює максимальну кількість клієнтів. По-перше, ми показуємо, що MaxClients має висхідний ефект на час відгуку і що для визначення оптимального значення MaxClients можна використовувати методи сходження. Це було продемонстровано за допомогою вимірювань та аналітичного моделювання. Основна інтуїція полягає в тому, що MaxClients контролює компроміс між затримками в черзі отримання TCP і затримками через боротьбу за ресурси операційної системи. Нами досліджено два оптимізатори, що використовують метод сходження на гору (один заснований на методі Ньютона, а інший - на нечіткому управлінні); третій метод є евристичним, який використовує взаємозв'язок між завантаженням вузьких місць і мінімізацією часу відгуку. В обох випадках веб-оптимізація показала, що скорочення часу відгуку було більш ніж у 10 разів порівняно з використанням статичних значень за замовчуванням. Компроміси між онлайн-системами полягають у наступному. Метод Ньютона добре відомий, але він не забезпечує послідовних результатів з дуже мінливими даними, такими як час відгуку. Нечіткий контроль є більш надійним, але збігається повільно. Евристичний метод добре працює в нашій прототипній системі, але його може бути важко узагальнити, оскільки він вимагає знання вузьких місць і здатності вимірювати використання ресурсів. Майбутні виклики включають наступне. По-перше, оптимізація декількох параметрів одночасно. Це може включати інші параметри, які можна динамічно налаштовувати в Apache, такі як KeepAliveTimeOut (який визначає, як довго TCP-з'єднання повинно залишатися активним для клієнта, перш ніж воно розірветься). По-друге, хоча ми вивчали налаштування продуктивності веб-серверів Apache, існують більш складні системи, такі як сервери баз даних і сервери додатків, де онлайн-оптимізація може мати більш драматичний вплив на час відгуку кінцевого користувача.

Література

1. Y. Diao, J. L. Hellerstein, and S. Parekh, “Optimizing quality of service using fuzzy control,” in Proceedings of Distributed Systems Operations and Management, 2002.

2. Apache Software Foundation. http://www.apache.org.

3. Y. Diao, J. L. Hellerstein, and S. Parekh, “A business-oriented approach to the design of feedback loops for performance management,” in Proceedings of Distributed Systems Operations and Management, 2001.

4. C. Lu, T. Abdelzaher, J. Stankovic, and S. Son, “A feedback control approach forguaranteeing relative delays in web servers,” in Proceedings of the IEEE Real-Time Technology and Applications Symposium, 2001.

5. Y. Diao, N. Gandhi, J. L. Hellerstein, S. Parekh, and D. M. Tilbury, “Using MIMOfeedback control to enforce policies for interrelated metrics with application to the Apache web server,” in Proceedings of Network Operations and Management, 2002.

6. L. Sha, X. Liu, Y. Lu, and T. Abdelzaher, “Queuing model based network serverperformance control,” in Proceedings of the IEEE Real-Time Systems Symposium, 2002.

7. D. Menasce, V. Almeida, R. Fonsece, and M. Mendes, “Busines oriented resourcemanagement policies for e-commerce servers,” Performance Evaluation, vol. 42, pp. 223–239, Oct. 2000.

8. Z. Liu, M. S. Squillante, and J. L. Wolf, “On maximizing service-level-agreementprofits,” in Proceedings of the ACM Conference on Electronic Commerce (EC’01), 2001.

9. Mindcraft, “Webstone 2.5 web server benchmark,” 1998. http://www.mindcraft.com/ webstone/.

10. Z. Liu, N. Niclausse, C. Jalpa-Villanueva, and S. Barbier, “Traffic model and performance evaluation of web servers,” Tech. Rep. 3840, INRIA, Dec. 1999.

11. D. Mosberger and T. Jin, “httperf: A tool for measuring web server performance,”in First Workshop on Internet Server Performance (WISP 98), pp. 59—67, ACM, June 1998.

12. D. P. Olshefski, J. Nieh, and D. Agrawal, “Inferring client response time at the webserver,” in Proceedings of the ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, 2002.

13. S. S. Lavenberg, ed., Computer performance modeling handbook. Orlando, FL: Academic Press, INC, 1983.

14. L. Perssini, The Mathematics of Nonlinear Programming. Springer-Verlag, 1988.

15. K. M. Passino and S. Yurkovich, Fuzzy Control. Menlo Park, CA: Addison Wesley Longman, 1998.

16. Зайцев Є.О., Антоненко А.В., Березниченко В.О., Закусило С.А. Smart засоби визначення аварійних станів у розподільних електричних мережах міст. Таврійський науковий вісник. Серія: Технічні науки, 2022, (5).

17. Лемешко А.В., Антоненко А.В., Цвик О.С. Аналіз і особливості програмного забезпечення для контролю трафіку. Вісник Хмельницького національного університету. Серія: Технічні науки, 2023, (1)

18. Лемешко А.В., Антоненко А.В., Балвак А.А., Новіченко Є.О. Актуальні засади створення алгоритмів обробки інформації для логістичних центрів. Таврійський науковий вісник. Серія: Технічні науки, 2023 (1)

19. Твердохліб А.О., Коротін Д.С., Антоненко А.В. Ефективність функціонування комп’ютерних систем при використанні технології блокчейн і баз данних. Таврійський науковий вісник. Серія: Технічні науки, 2022, (6)

20. Lei Song, Biswanath Mukherjee.On the Study of Multiple Backups and Primary-Backup Link Sharing for Dynamic Service Provisioning in Survivable WDM Mesh Networks / IEEE Journal on selected areas in Telecommunication, 2008. Vol. 26, No 6. pp. 84–91

References

1. Y. Diao, J. L. Hellerstein, and S. Parekh, “Optimizing quality of service using fuzzy control,” in Proceedings of Distributed Systems Operations and Management, 2002.

2. Apache Software Foundation. http://www.apache.org.

3. Y. Diao, J. L. Hellerstein, and S. Parekh, “A business-oriented approach to the design of feedback loops for performance management,” in Proceedings of Distributed Systems Operations and Management, 2001.

4. C. Lu, T. Abdelzaher, J. Stankovic, and S. Son, “A feedback control approach forguaranteeing relative delays in web servers,” in Proceedings of the IEEE Real-Time Technology and Applications Symposium, 2001.

5. Y. Diao, N. Gandhi, J. L. Hellerstein, S. Parekh, and D. M. Tilbury, “Using MIMOfeedback control to enforce policies for interrelated metrics with application to the Apache web server,” in Proceedings of Network Operations and Management, 2002.

6. L. Sha, X. Liu, Y. Lu, and T. Abdelzaher, “Queuing model based network serverperformance control,” in Proceedings of the IEEE Real-Time Systems Symposium, 2002.

7. D. Menasce, V. Almeida, R. Fonsece, and M. Mendes, “Busines oriented resourcemanagement policies for e-commerce servers,” Performance Evaluation, vol. 42, pp. 223–239, Oct. 2000.

8. Z. Liu, M. S. Squillante, and J. L. Wolf, “On maximizing service-level-agreementprofits,” in Proceedings of the ACM Conference on Electronic Commerce (EC’01), 2001.

9. Mindcraft, “Webstone 2.5 web server benchmark,” 1998. http://www.mindcraft.com/ webstone/.

10. Z. Liu, N. Niclausse, C. Jalpa-Villanueva, and S. Barbier, “Traffic model and performance evaluation of web servers,” Tech. Rep. 3840, INRIA, Dec. 1999.

11. D. Mosberger and T. Jin, “httperf: A tool for measuring web server performance,”in First Workshop on Internet Server Performance (WISP 98), pp. 59—67, ACM, June 1998.

12. D. P. Olshefski, J. Nieh, and D. Agrawal, “Inferring client response time at the webserver,” in Proceedings of the ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, 2002.

13. S. S. Lavenberg, ed., Computer performance modeling handbook. Orlando, FL: Academic Press, INC, 1983.

14. L. Perssini, The Mathematics of Nonlinear Programming. Springer-Verlag, 1988.

15. K. M. Passino and S. Yurkovich, Fuzzy Control. Menlo Park, CA: Addison Wesley Longman, 1998.

16. Zaitsev Ye.O., Antonenko A.V., Bereznychenko V.O., Zakusylo S.A. Smart zasoby vyznachennia avariinykh staniv u

rozpodilnykh elektrychnykh merezhakh mist. Tavriiskyi naukovyi visnyk. Seriia: Tekhnichni nauky, 2022, (5).

17. Lemeshko A.V., Antonenko A.V., Tsvyk O.S. Analiz i osoblyvosti prohramnoho zabezpechennia dlia kontroliu trafiku. Visnyk Khmelnytskoho natsionalnoho universytetu. Ceriia: Tekhnichni nauky, 2023, (1)

18. Lemeshko A.V., Antonenko A.V., Balvak A.A., Novichenko Ye.O. Aktualni zasady stvorennia alhorytmiv obrobky informatsii dlia lohistychnykh tsentriv. Tavriiskyi naukovyi visnyk. Seriia: Tekhnichni nauky, 2023 (1)

19 Tverdokhlib A.O., Korotin D.S., Antonenko A.V. Efektyvnist funktsionuvannia kompiuternykh system pry vykorystanni tekhnolohii blokchein i baz dannykh. Tavriiskyi naukovyi visnyk. Seriia: Tekhnichni nauky, 2022, (6)

20. Lei Song (2008) Biswanath Mukherjee. On the Study of Multiple Backups and Primary-Backup Link Sharing for Dynamic Service Provisioning in Survivable WDM Mesh Networks / IEEE Journal on selected areas in Telecommunication. Vol. 26, No 6. pp. 84–91.