

## РАЗРАБОТКА СИСТЕМЫ АВТОМАТИЧЕСКОГО УПРАВЛЕНИЯ НАГРУЗКОЙ WEB-СЕРВЕРА

*Статья описывает алгоритм разработки системы автоматического управления нагрузкой web-сервера. В качестве объекта управления выбран web-сервер Apache под управлением ОС Linux.*

*В статье проведено обоснование структуры системы автоматического управления. Система управления использует многомерный линейно-квадратичный регулятор. Переходные процессы показывают высокое качество регулирования при типовых возмущениях.*

*Применение данной системы позволяет увеличить эффективность использования ресурсов корпоративных серверов. Разработанная система также может быть использована для противодействия DDoS - атак.*

*Ключевые слова: система управления, регулятор, Web-сервер, Apache.*

A.A. Stopakevych, K.U. Komarov

Odessa National Academy of Telecommunications n.a O.S. Popov, Odessa, Ukraine

## DEVELOPMENT OF AUTOMATIC LOAD CONTROL SYSTEM FOR THE WEB-SERVER

*The article describes the design algorithm of web-server load control system. The web-server Apache under OS Linux control is chosen as a control plant.*

*The structure of control system is grounded in the article. The control system uses multivariable linear-quadratic regulator. The process dynamics shows good control quality with typical disturbances.*

*The implementation of the developed system increases resource usage efficiency of corporate servers. The control system may be also used for DDoS attacks resistance.*

*Keywords: control system, controller, Web-server, Apache.*

**Вступление.** Использование систем автоматического управления с обратной связью в компьютерных системах является актуальной задачей, поскольку позволяет более эффективно использовать ресурсы компьютерной техники, подбирая требуемые настройки серверных программ исходя из текущей нагрузки на сервер, которая имеет тенденцию меняться во времени, особенно при DDoS-атаках. Такой подбор практически нереализуем вручную и, как правило, не проводится системными администраторами.

Известно, что любой сервер ограничен определенным количеством оперативной памяти (ОЗУ) и мощностью центрального процессора (ЦП), поэтому основным требованием к серверному программному обеспечению является максимально эффективное их использование. Отказ от обслуживания клиентов может приводить к экономическим потерям компании. Поэтому, кроме увеличения ресурсов серверов и их количества, современные крупные компьютерные корпорации, такие как Microsoft, IBM и Google, исследуют вопросы масштабирования и оптимизации использования ресурсов серверами с помощью математических методов. Так, например, сотрудниками корпорации IBM проводились исследования использования систем автоматического регулирования для управления СУБД, серверами каталогов, почтовыми серверами, Web-серверами, файловыми хранилищами, кластерами и т.п. Исследования показали эффективность применения теории автоматического управления и перспективность данной тематики [1].

**Постановка проблемы.** Типичный корпоративный сервер, как правило, одновременно выполняет несколько функций: Web-сервер, почтовый сервер, сервер баз данных, обеспечение работы корпоративных систем электронного документооборота, работа других приложений. Для обеспечения эффективной параллельной работы Web-сервера и других служб целесообразно балансировать нагрузку, выделив каждому программному серверу определенную часть ресурсов. Серверные операционные системы не имеют эффективных средств для такой балансировки, поэтому применение системы автоматического регулирования для данной задачи является целесообразным.

**Web-сервер как объект управления.** Наиболее распространенными web-серверами являются Apache HTTP Server, IIS (Internet Information Services), Nginx (Engine X) [2]. Apache HTTP Server – наиболее распространенный, функционально развитый кроссплатформенный Web-сервер. Он доступен для таких серверных операционных систем (ОС) как Linux, FreeBSD, ряда других UNIX-подобных ОС, а также Microsoft Windows Server. Среди поддерживаемых языков программирования для разработки Web-сайтов – PHP, Python, Ruby, Perl, также возможно использование языков программирования общего назначения: C, C++, Java [3].

Таким образом, использование Apache для решения поставленной задачи является наиболее целесообразным. Наиболее часто Apache используется в окружении ОС Linux [2]. Построим структурную модель выбранного объекта управления. Поскольку основными ресурсами сервера, как было сказано выше, являются доступная память и загрузка ЦП, то в качестве управляемых переменных системы целесообразно использовать именно их.

Основные параметры, которые непосредственно влияют на уровень нагрузки на аппаратное обеспечение серверного компьютера в пределах Apache HTTP Server, являются параметры MaxClients (MC)

и KeepAliveTimeout (*KAT*). Значение *MC* позволяет задать ограничение на размер пула процессов (pool of worker processes), что в свою очередь определяет количество клиентских подключений, которые сервер может обрабатывать одновременно. Чем выше это значение, тем выше нагрузка на аппаратное обеспечение (в пределах данной задачи, на ЦП и ОЗУ) и тем выше одновременное подключение к рассматриваемому серверу. Однако чрезмерно высокое значение может привести к значительному падению производительности. Параметр *KAT* (в секундах) определяет время ожидания запущенным процессом возможного запроса от того же самого клиента без установления повторного соединения. В случае, когда значение *KAT* слишком высокое, можно наблюдать спад нагрузки на ЦП и ОЗУ, однако увеличивается шанс, что клиенты вообще не смогут подключиться к серверу. Таким образом, увеличение данной настройки приводит к уменьшению нагрузки, но приводит к увеличению времени между обработками запросов [4].

Для общего представления о величинах, действующих в пределах рассматриваемой системы управления на объект, нарисую параметрическую схему объекта управления. На схеме обозначим объект в виде блока, а также величины, которые непосредственно влияют на ход выполнения процесса: управляющие воздействия и управляемые переменные. Параметрическая схема рассматриваемого объекта представлена на рис. 1.

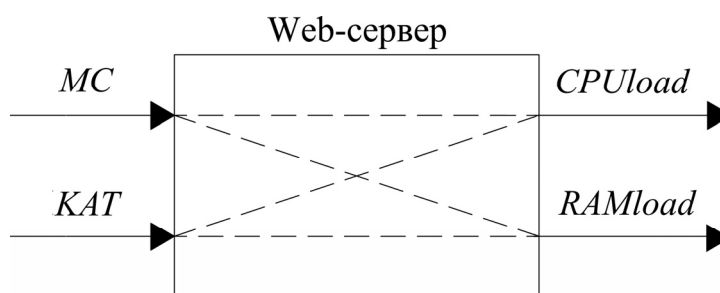


Рис. 1. Параметрическая схема рассматриваемого Web-сервера

**Исполнительная подсистема.** Для реализации рассчитанных управлений, параметры (настройки) Web-сервера должны иметь возможность изменяться динамически без перезагрузки самого сервера. Однако Apache HTTP Server не поддерживает данную возможность, поэтому в стандартной поставке для изменения параметров Web-сервера, которые записываются в файлы конфигурации, необходима его перезагрузка. Решением данной проблемы является разработка программного модуля, который обеспечит динамическое изменение входных параметров без перезагрузки Web-сервера. Структурная схема компонентов Apache HTTP Server вместе с указанным модулем представлена на рис. 2.

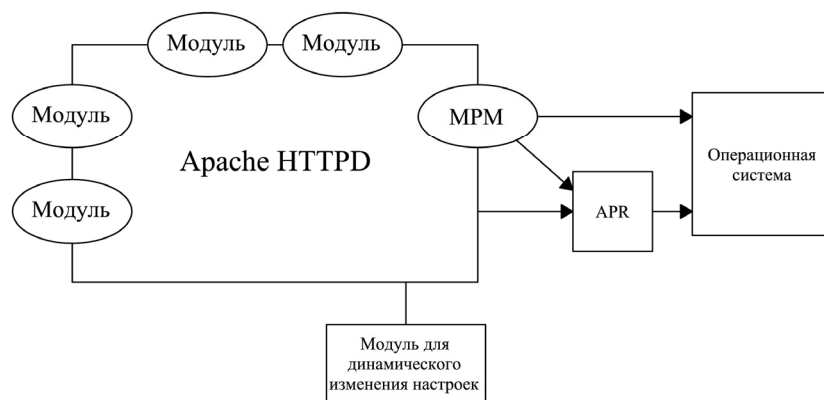


Рис. 2. Структурная схема компонентов Apache HTTP Server с подключенным модулем для динамического применения измененных параметров сервера

MPM (Multi-Processing Module) является модулем специального назначения, предназначенным для оптимизации Apache-сервера под операционную систему, на которой он установлен. В основном, данный модуль служит для доступа сервера к ОС.

APR (Apache Portable Runtime) является набором библиотек, обеспечивающим единый интерфейс разработки для всех платформ. Целью такой библиотеки является необходимость предотвращения ошибок при работе разработанного модуля, связанных с особенностями той или иной ОС. Так как программное обеспечение должно разрабатываться в среде определенной операционной системы, то наличие такой библиотеки является необходимым.

**Измерительная подсистема.** Кроме разработки программного модуля существует и дополнительная задача – определение величин загруженности ресурсов аппаратного обеспечения с помощью измерительной системы. Поскольку алгоритмы и способы получения данных параметров зависят от операционной системы, а также её настроек, целесообразно разработать измерительную систему с реконфигурируемой структурой [5]. Для решения этой задачи в ОС Linux разработана программа-скрипт,

созданный с использованием языка программирования высокого уровня Python [6]. Программа позволяет считывать записи операционной системы о количестве и характеристиках запущенных процессов и измерять выходные параметры объекта управления – загрузку ЦП и ОЗУ.

**Среда исполнения.** Для сборки исходного кода и запуска необходимого программного обеспечения был выбран такой дистрибутив ОС Linux как Ubuntu Server. Данный дистрибутив является свободным для загрузки и использования, а также поставляется со всеми необходимыми программными пакетами для работы с сетью и компиляции исходного кода как на языке C (на котором написан исходный код Apache), так и на множестве других языков.

**Разработка экспериментальной математической модели динамики сервера.** Построение экспериментальной математической модели будет проводиться методом активного эксперимента [7]. Для этого задействованы два персональных компьютера, соединенные сетью. Один компьютер играет роль серверного компьютера, второй – клиентского. Так как отсылка всего лишь одного пользовательского запроса на объект управления не позволит оценить его динамические характеристики, то на клиентскую машину необходимо установить специальное сетевое программное обеспечение для тестирования надежности серверов. Суть данного ПО – эмулировать одновременную посылку множества запросов на указанный сетевой адрес. Его использование помогает имитировать различные, приближенные к реальным, ситуации нагрузки на сервер.

Конфигурация клиентского и серверного компьютера приведена в табл. 1. В таблице перечислены параметры оборудования, существенные для поставленной задачи.

Таблица 1

**Конфигурация основных комплектующих серверного и клиентского персональных компьютеров**

	Сервер	Клиент
ЦП	Intel Core i3, 2 ядра по 2,53 ГГц	Intel Core2Duo, 2 ядра по 2,0 ГГц
ОЗУ	4 Гб	2 Гб
ОС	Linux Ubuntu Server версия 14.04	Windows 7 Professional

Среди большого количества программного обеспечения для тестирования надежности Web-серверов, наиболее функционально развитой и подходящей для поставленной задачи является программа JMeter [8].

Активный эксперимент проходит следующим образом. После запуска созданной задачи, к сетевому адресу, указанному в компоненте “HTTP-запрос”, подключается заданное количество пользователей и проводит загрузку страницы. Затем, когда процент загрузки CPU и RAM –  $CPUload$  и  $RAMload$  становится почти постоянным, проводится изменение количества пользователей сначала в сторону увеличения, а затем в сторону уменьшения. Все параметры записываются на сервере и после усреднения разгонных характеристик, а также их аппроксимации, была получена математическая модель в форме пространства состояний вида:

$$y_{k+1} = A \cdot y_k + B \cdot u_k, A = \begin{bmatrix} 0.72 & -0.147 \\ -0.0305 & 0.75 \end{bmatrix}, B = \begin{bmatrix} -84.5 & 4.39 \\ -2.48 & 2.81 \end{bmatrix} \cdot 10^{-4} \quad (1)$$

где  $u_k$  – текущая величина вектора  $[MC, KAT]^T$ , стандартные единицы настройки сервера Apache;

$y_k$  и  $y_{k+1}$  – текущая и последующая выходные величины, вектора  $[CPUload, KATload]^T$ , в долях единицы.

**Обоснование структуры системы автоматического управления и её технической реализации.** Для регулирования многомерных и многосвязных объектов управления с известной математической моделью рекомендуется использовать многомерные регуляторы [9].

Структурная схема многомерной автоматической системы управления с отрицательной обратной связью представлена на рис. 3.

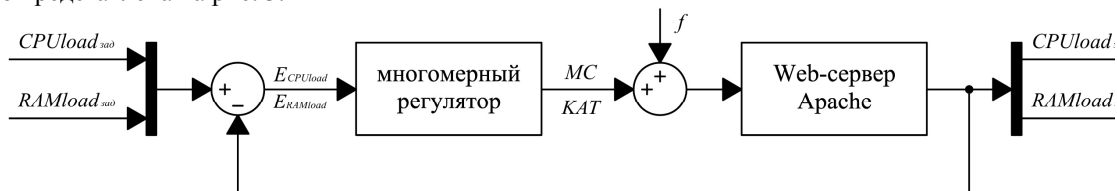


Рис. 3. Структурная схема многомерной САУ с Web-сервером Apache

Заданием системы управления является требуемая загрузка ЦП ( $CPUload_{зад}$ ) и требуемая загрузка оперативной памяти ( $RAMload_{зад}$ ), задаваемые в долях единицы, которые передаются в регулятор. Регулятор реализован как фоновая служба, которая с заданным периодом получает текущие параметры загрузки с помощью вызова программы-скрипта на Python. Затем, программа вычисляет необходимые управляющие

воздействия (требуемые значения *MaxClients* и *KeepAliveTime*) и отправляет их с помощью механизма сокетного соединения, подключенного к Apache-модулю, в таблицы настройки Apache. Поскольку заданная модель, по которой рассчитан регулятор, не может точно отражать динамику объекта управления, то в системе присутствуют неконтролируемые возмущения (содержание трафика, работа других служб, работа операционной системы), для эффективного устранения которых необходимо использовать специальные методики расчета регуляторов.

**Обоснование выбора типа многомерного регулятора.** Для обеспечения регулирования процесса был выполнен синтез системы управления с дискретным линейно-квадратичным регулятором.

Структурная схема регулятора представлена на рис. 4. Регулятор включает матрицы пропорциональной ( $K_1$ ) и интегральной ( $K_2$ ) составляющих.

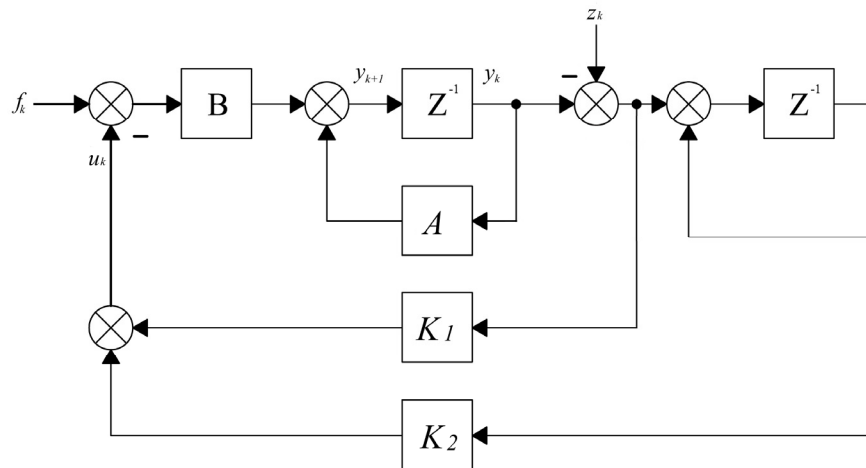


Рис. 4. Структурная схема линейно-квадратичного ПИ-регулятора, подключенного к системе управления

Для расчета цифрового линейно-квадратичного ПИ-регулятора используется уравнение Риккати. Алгоритм расчета регулятора выполняется следующим образом. В начале, формируется расширенная модель системы, включающая последовательно соединенные модели исходной системы и возмущений:

$$Ad = \begin{bmatrix} A & 0 \\ I & I \end{bmatrix}, \quad Bd = \begin{bmatrix} B \\ 0 \end{bmatrix} \quad (2)$$

где  $I$  – единичная матрица.

Следующим шагом является решение дискретного уравнения Риккати (в программе Matlab):

$$K = [K_1, K_2] = dlqr(Ad, Bd, Q, R) \quad (3)$$

где  $Q, R$  – диагональные настроечные матрицы регулятора.

После этого, формируются матрицы регулятора  $A_r, B_r$  и  $C_r$ :

$$A_r = I, B_r = I, C_r = -K_2, D_r = -K_1 \quad (4)$$

для расчета управляющего воздействия по закону:

$$x_{k+1} = A_r \cdot x_r + B_r(y_k - z), \quad u_{k+1} = C_r \cdot x_r + D_r(y_k - z) \quad (5)$$

Собирается система управления и моделируется переходной процесс при подаче максимальных возмущений и/или отклонении задания.

Настройка регулятора проводится следующим образом. Если выходы системы отклоняются больше регламентных значений, то увеличивается диагональный элемент матрицы  $Q$ , соответствующий номеру отклоненного выхода. Если управляющее воздействие превышает ограничение, то увеличивают диагональный элемент матрицы  $R$ , соответствующий номеру превысившего управления и снова решают уравнение Риккати. Если не удастся подобрать матрицы  $Q$  и  $R$ , то уменьшают максимальную величину отклонения задания и возмущения так, чтобы выполнялись требования. После этого делают вывод о том, что система способна парировать только найденные максимальные возмущения и отклонения задания.

В результате функционирования замкнутой системы управления с объектом управления были получены графики переходных процессов при возмущении максимальным изменением количества клиентов, изображенные на рис. 5. Анализ графиков показывает, что система управления справилась с возмущением и вернулась к установленным значениям заданий.

### Выводы

Синтез регуляторов для систем автоматического управления компьютерными системами является перспективной темой исследований. В работе показано, что разработанная система автоматического управления нагрузкой web-сервера Apache под управлением ОС Linux способна справиться с поставленной задачей, что позволяет использовать её для балансирования нагрузки на многофункциональных корпоративных серверах. Использование предложенного подхода к построению системы управления позволяет уменьшить количество незапланированных отказов в обслуживании. Он может быть применен

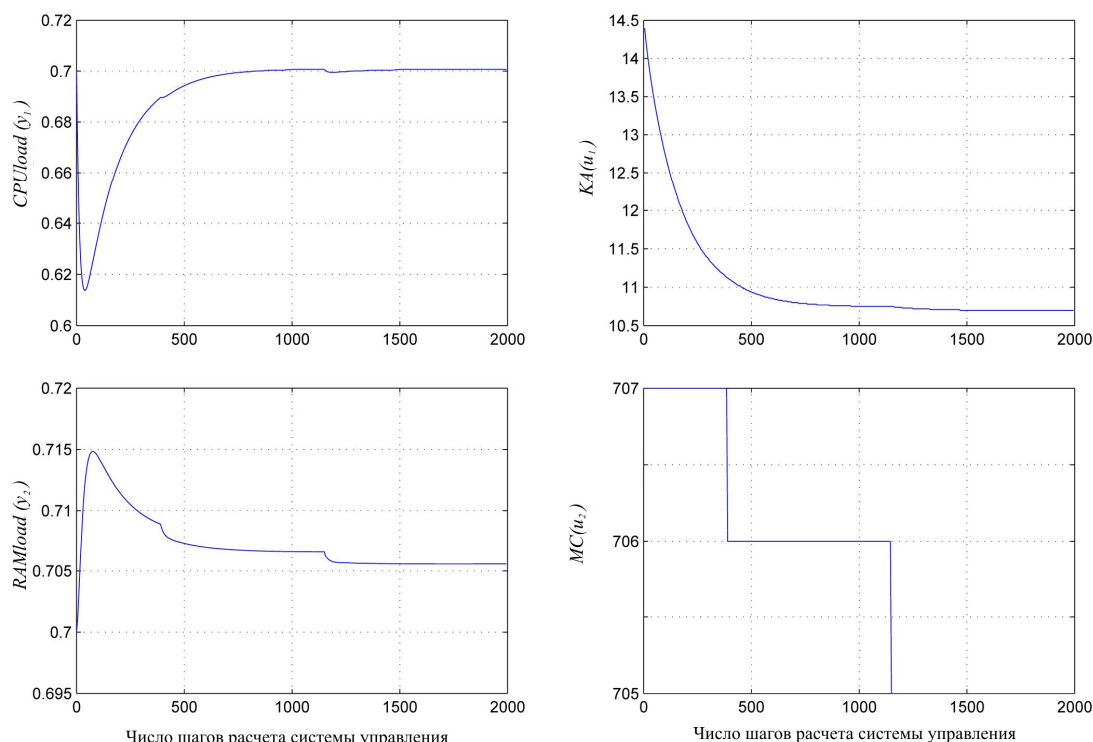


Рис. 5. Результаты работы линейно-квадратичного ПИ-регулятора в исследуемой системе управления

## Литература

1. Yixin Diao. Control of Large Scale Computing Systems / D. Yixin, J. Hellerstein, S. Parekh, T. Watson, ACM SIGBED Review.–2006.–V.3.–P.17-22.
2. Netcraft. August 2015 Web Server Survey [Электронный ресурс] – Режим доступа.– URL: <http://news.netcraft.com/archives/2015/08/13/august-2015-web-server-survey.html>
3. Kew N. The Apache Modules Book .– NJ, USA: Prentice Hall PTR Upper Saddle River, 2007.
4. Gandhi N. MIMO control of an apache web server: Modeling and controller design / N. Gandhi, D. Tilbury, Y. Diao, J. Hellerstein, S. Parekh//Proc. of 6-th American Control Conference.- 2002.-P. 4922-4927
5. Кондратов В.Т. Определение и базовая классификация измерительных систем/В.Т.Кондратов //Вимірювальна та обчислювальна техніка в технологічних процесах.– 2014. – № 3. – С. 85-100.
6. Alvarez T. How Can Apache Help to Teach And Learn Automatic Control / T.Alvarez, H. D. Herrero, J. Francisco, D. Madrigal//IEEE Education Engineering (EDUCON).–2010.–C.1539-1546.
7. Дилигенская А. Н. Идентификация объектов управления .– Самара: Самарский государственный технический университет, 2009.
8. Apache JMeter Manual [Электронный ресурс] – Режим доступа.– URL: <http://jmeter.apache.org/usermanual/get-started.html>
9. Стопакевич А.А. Системный анализ и теория сложных систем управления.– Одесса: Астропринт, 2013.

## References

1. Yixin Diao. Control of Large Scale Computing Systems / D. Yixin, J. Hellerstein, S. Parekh, T. Watson, ACM SIGBED Review.– 2006.–V.3.–P.17-22.
2. Netcraft. August 2015 Web Server Survey [E'lektronnyj resurs] – Rezhim dostupa.– URL: <http://news.netcraft.com/archives/2015/08/13/august-2015-web-server-survey.html>
3. Kew N. The Apache Modules Book .– NJ, USA: Prentice Hall PTR Upper Saddle River, 2007.
4. Gandhi N. MIMO control of an apache web server: Modeling and controller design / N. Gandhi, D. Tilbury, Y. Diao, J. Hellerstein, S. Parekh//Proc. of 6-th American Control Conference.- 2002.-P. 4922-4927
5. Кондратов В.Т. Определение и базовая классификация измерительных систем/В.Т.Кондратов // Vymiriuvalna ta obchysliuvalna tekhnika v tekhnolohichnykh protsesakh.– 2014. – № 3. – С. 85-100.
6. Alvarez T. How Can Apache Help to Teach And Learn Automatic Control / T.Alvarez, H. D. Herrero, J. Francisco, D. Madrigal//IEEE Education Engineering (EDUCON).–2010.–C.1539-1546.
7. Diligenskaya A. N. Identifikaciya ob'ektov upravleniya .– Samara: Samarskij gosudarstvennyj texnicheskij universitet, 2009.
8. Apache JMeter Manual [E'lektronnyj resurs] – Rezhim dostupa.– URL: <http://jmeter.apache.org/usermanual/get-started.html>
9. Stopakevich A.A. Sistemyj analiz i teoriya slozhnyx sistem upravleniya.– Odessa: Astroprint, 2013.

Рецензія/Peer review : 7.11.2014 р.

Надрукована/Printed :20.10.2015 р.

Рецензент: Загребнюк В.І.