ГОГУ. Для оценки качества прохождения тренировок предлагается использовать гибко настраиваемую систему штрафных баллов, выявляющую слабые места в подготовке того или иного специалиста.

Литература

1. Юрыгина Ю.С., Коршиков Д.Н., Носкова А.И. Адаптивный тренажер для формирования и восстановления навыков ситуационной поддержки принятия решений сменным руководителем полетов и специалистами ГОГУ // Тезисы докладов XX научно-технической конференции молодых ученых и специалистов, ОАО «Ракетно-космическая корпорация «Энергия» имени С.П. Королёва», 10 – 14 ноября 2014 года. – С. 629-631.
2. Матюшин М.М., Мишурова Н.В., Скобелев П.О., Ларюхин В.Б. Поддержка принятия решений при парировании аварийных ситуаций на борту международной космической станции с использованием интеллектуальных технологий / М.М. Матюшин // Вестник НПО им. С.А. Лавочкина. – 2015.- №2. – С. 52-57.

П.К. Попков

НАГРУЗОЧНОЕ ТЕСТИРОВАНИЕ СИСТЕМ УПРАВЛЕНИЯ САЙТОМ

(Самарский университет)

Основная цель тестирования – определить динамику потребления ресурсов при различных нагрузках на систему с последующим выводом и выявлением вектора развития разрабатываемой нами системы NapsideCMS.

Для тестирования был выбран ряд популярных систем, как коммерческих так и свободно распространяемых. Ниже приведены их названия и используемые версии: Bitrix Малый бизнес — 17.0.9, WordPress 4.9.1, Drupal 7.56, Joomla! 3.6.5, Umi.cms 16, NapsideCMS 0.1.0 R1, InstantCMS 2.8.2, KodiCMS 12.20.37(master), NetCat Standart 5.8, HostCMS 6.7.6

Время загрузки страницы состоит из двух частей: времени генерации HTML-источника страницы и времени дальнейшей подгрузки содержимого страницы (img, css, js и прочего содержимого, указаного в HTML-источнике). Отдача же физических файлов (картинок, css-файлов и пр.) не использует ресурсы сервера (память, процессор) и не использует SQL-сервер, поэтому не рассматривается в данном исследовании.

Более того, браузеры кешируют многие файлы, поэтому при повторной загрузке сайта и выводе кешированной картинки сервер почти не задействуется. Однако формирование HTML-источника — это процесс, на который сервер тратит свои ресурсы. Построение страницы производит интерпретатор PHP-скриптов. Это делается активном использовании ресурсов сервера: процессора и оперативной памяти, обращений к SQL-серверу. Самое узкое место в быстродействии таких связок — это SQL запросы. Чтобы не было

проблем, кроме правильной структуры и индексирования БД, должны быть правильно сформированы запросы. Интерпретатор PHP не всегда может корректно освободить используемые данные, поэтому обязательным является самостоятельное закрытие соединения с БД (mysql_close). Также нужно рационально использовать память во время работы скрипта с БД, используя mysql_free, иначе результирующие данные запроса так и останутся в памяти до конца работы интерпретатора.

Для проведения экспериментов на хостинге были установлены веб сервер, база данных, дополнительное ПО, необходимое для корректной работы каждой CMS системы. В качестве тестового стенда использовались: Apache 2.2.34, PHP 5.4, MySQL 5.7.20, Zend Engine 2.6.0, Apache benchmark 2.4.

Каждая CMS была установлена в отдельную директорию, после чего в индексируемую страницу встраивался сторонний код, который позволял делать замеры и сохранять результат о времени выполнения скрипта и объеме затраченной памяти.

На выходе был получен файл лог-файл, в котором содержались необходимые данные о времени загрузки скрипта и потреблении оперативной памяти при его выполнении. Для создания запросов на сайт был использован ab (Apache benchmark), без создания параллельных запросов. Стоит отметить, что целью не было тестирование самого сервера apache, лишь получение средних результатов обработки запросов для каждой CMS:

ab -n 1000 http://dir_sX.altmedia.su/ > ~/bench/ab/dir_sX.altmedia.log.

Директории «dir_sX» соответствуют своей CMS и указаны в следующем порядке:s1 - Bitrix, s2 – Wordpress, s3 – Drupal, s4 - Joomla, s5 - Umi.CMS, s6 - Napside CMS, s7 – InstantCMS, s8 – KodiCMS, s9 – NetCat, s10 HostCMS. После сбора данных имеется подробная картина структуры каждой системы по отдельности. А именно, ее полный размер и наполнение базы данных каждой CMS. Эти цифры варьируются от 3мб до 200мб.

Для оценку скорости загрузки скрипта проведено несколько прогонов подряд и собираны все данные по критериям «Минимальное время», «Максимальное время», «Среднее время». На основе этих данных был проведен следующий тест: «Оценка Apache benchmarks», где были собраны полезные данные касающиеся взаимодействия каждой CMS с конечным пользователем.

Так как каждая CMS-система управления сайтом поддерживает работу с каталогами файлов или в качестве интернет магазина, то в данном случае был подготовлен CSV-файл с наполнением каталога, который содержит в себе чуть более 10 000 наименований. Некоторые CMS, такие как wordpress и kodiCMS не поддерживают «из коробки» импорт .csv файлов, поэтому файлы были импортированы напрямую в базу данных. Также стоит отметить, что некоторые CMS требуют определенное представление импортируемых файлов. MySQL – это сервер, принимающий запросы и после обработки отдающий результат. Очевидно, что для увеличения быстродействия требуется снизить количество SQL запросов.

Также проблемой многих CMS являются циклы. Рассмотрим запрос ***SELECT id FROM base WHERE per=1 WHILE { SELECT image FROM base_image WHERE base_id = id }***. Если имеется 200 записей, и 200 пользователей сделавших запрос то будет 201*200=40200 запросов к БД. MySQL является отдельным сервером и запросы поступают по протоколу TCP в бинарном виде. Таким образом PHP отправит 40200 запросов TCP на порт 3306, забивая сетевую карту сервера их обработкой. TCP-соединение в UNIX – это файловый дескриптор, являющийся объектом ядра. 40200 объектов ядра содержат минимум 4 байта каждый только для хранения id. То есть 4 * 40200 = 40 кб данных, которые стоят в очереди. Это расход памяти только на один запрос выборки id позиций с картинками. В итоге, если давать рекомендации, рассматривая наш простой запрос выборки позиций с картинками, лучше сначала запросить необходимые позиции ***SELECT id FROM base WHERE per=1***, сохранить результат и затем создать отдельный запрос картинок ***SELECT image FROM base_image WHILE base_id IN (1,2,3...100)*** — это 2 запроса к базе mySQL от каждого из пользователей и всего 2*200=400 запросов к БД, вместо 40200.

Также важным является использование в CMS внутрисистемного кеширования как результатов запросов, так и внутренних данных скриптов. Это можно достичь разными способами, и желательно использовать несколько одновременно. Начиная от статических функциональных переменных (как локальные хранилища), до файлового кеша, когда результаты большинства SQL-запросов сохраняются CMS во временные файлы, затем используются вместо дубликата запросов. Если циклы, неоптимизированные запросы к SQL и отсутствие алгоритмов кеширования не особо заметны при нескольких одновременных посетителях, то когда посетителей онлайн на сайте 100, 250 и больше, временная разница становится заметно ощутимой. Для этих целей и проводится нагрузочное тестирование. Есть несколько вариантов проведения нагрузочного тестирования. Наряду с профессиональным программным обеспечением такими как Apache Jmeter и Яндекс.Танк, существуют условно бесплатные онлайн решения для быстрого и легкого проведения нагрузочного тестирования сценарным методом сразу для всех CMS. Для этого был использован сервис loaddy.com. Тестирование проводилось по очереди, на каждую CMS по 10 минут, интервалы между проверками — 1 минута. Сначала была дана нагрузка в 100 человек. Результаты нагрузочного тестирования представлены в таблице 1.

Таблица 1 –Результаты нагрузочного тестирования 100чел./10мин.

Test/CMS	Bitrix	Wordpress	Drupal	Joomla	Umi.CMS	NapsideCMS	InstamtCMS	KodiCMS	NetCat	HostCMS
Доступность	2.91 %	100 %	100 %	100 %	56,37 %	100 %	100 %	76,25 %	92.18 %	100 %
Скорость нагрузки	0.38 с.	0.25 с.	0.22 с.	0.39 с.	0.30 с.	0.19 с.	0.23 с.	0.32 с.	0.29 с.	0.21 с.
Время ответа	0.23 с.	0.13 с.	0.11 с.	0.14 с.	0.21 с.	0.10 с.	0.11 с.	0.19 с.	0.15 с.	0.10 с.
Получено данных	7.22 мб.	5.21 мб.	4.13 мб.	5.18 мб.	4.12 мб.	5.39 мб.	6.65 мб.	5.72 мб.	4.29 мб.	5.30 мб.

Устойчивость сайта при внезапных нагрузках — важный момент. Конечно, постоянная посещаемость в 500 одновременных он-лайн посетителей встречается, мягко говоря, не повсеместно. Это 3.000 посетителей в час или 72.000 уникальных посетителей в сутки, а при действии каждого посетителя раз в 5 секунд — более 8.000.000 хитов. Владельцы сайтов с такой постоянной посещаемостью выбирают выделенные сервера, а не VPS. Однако и владелец небольшого интернет-магазина может давать рекламу, привлекать посетителей, и при кажущейся надежности сайта, в пики рекламной кампании обнаружить значительное снижение доступности сайта и несколько процентов от ожидаемых продаж. Поэтому для нагрузочного тестирования в качестве платформы был использован недорогой виртуальный сервер, рыночная стоимость которого находится в пределах 1000 рублей в месяц. Возможно, на выделенном физическом сервере с индивидуальной настройкой каждой системы под ресурсы, результаты у выбывших CMS были бы немного другими.

Литература

1. Криспин Л., Грегори Д. Гибкое тестирование: практическое руководство для тестировщиков ПО и гибких команд М.: «Вильямс», 2010. — 464 с.
2. Нирав Метха. CMS с открытым исходным кодом. Пособие для начаинающих., 2009. — 340 р.
3. Брамптон М.. PHP5 CMS Framework Development. Июнь 2008. — 328 р.
4. Денис Колисниченко. Движок для вашего сайта. CMS Joomla!, Slaed, PHP-Nuke. — Петербург: БХВ, 2008. — 352 с.
5. Савельева Н. Системы управления контентом (рус.) // Открытые системы. — 2004. — № 4.

О.В. Порубай, А.А. Горовик

ИСПОЛЬЗОВАНИЕ МЕТОДА ВИОЛЫ-ДЖОНСА ДЛЯ РАЗРАБОТКИ СИСТЕМЫ РАСПОЗНАВАНИЯ ЛИЦА ЧЕЛОВЕКА НА ФОТО И ВИДЕОИЗОБРАЖЕНИЯХ

(Ферганский филиал ТУИТ им.Мухаммада ал-Хоразмий)

В связи с быстрым ростом и развитием информационных технологий, большинство направлений техники и науки используют системы, в которых информация носит характер поля (изображения, видеоизображения). Начиная обрабатывать такую информацию возникает множество сложных проблем. Определив ряд этих проблем, можно выделить одну из самых сложных – обработка и распознавание образа на изображении (видеоизображении) [1].

Процессы узнавания и распознавания образов всегда был интересны и значительны, особенно в связи с возрастающими потребностями защиты: охранные системы, верификация пластиковых и кредитных карт, экспертиза в области криминалистики, биометрические системы идентификации, телеконференции и прочее. Смотря на то, что человек хорошо распознает лица людей, многие разработчики программного обеспечения пытаются «научить компьютер» проводить эту процедуру, разрабатывая все большее количество различного рода систем распознавания лица человека.

Описание алгоритма метода Виолы-Джонса. Для решения задачи распознавания лиц используются различные методы. Одним из них является метод Виолы-Джонса, который был взят за основу при написании программы распознавания лица человека на фото и видеоизображениях в реальном времени.

Метод был предложен Полом Виолой и Майклом Джонсом в 2001 году [2,3]. Благодаря своей высокой точности и теоретической основе, данный метод стал приобретать большую популярность среди программистов. Однако, сложные вычисления, используемые в данном методе, по-прежнему нуждались в мощной аппаратной платформе. Данный метод в общем виде использует принцип сканирующего окна. То есть рамка, которая значительно меньше, чем отображаемое в камере изображение, двигается с некоторым заданным шагом по изображению, и с помощью каскада слабых классификаторов ищет черты лица и помечает их.

Обобщенную схему распознавания в алгоритме Виолы-Джонса можно увидеть на рисунке 1.

Метод основывается на следующих основных принципах:

- изображения используются в интегральном представлении –это позволяет быстро вычислять нужные объекты и детали;
- для поиска нужного объекта используются признаки Хаара;
- для выбора наиболее подходящих признаков для искомого объекта на данной части изображения, используется бустинг (от англ. boost – улучшение, усиление);