Empirical Examination of a Collaborative Web Application

Christopher Stewart (stewart@cs.rochester.edu), Matthew Leventi (mlventi@gmail.com), and Kai Shen (kshen@cs.rochester.edu) University of Rochester
The collaborative web application (WeBWoRK), trace, and dataset described in these slides are publicly available.

http://www.cs.rochester.edu/u/stewart/collaborative.html
Motivation

- “Benchmarks shape a field for better or worse; they are how we determine the value of change.”
  -- David Patterson, 1994

- Web applications benefit from systems research

- Systems research benefits from good benchmarks
  - SPECweb for static web content
  - TPC benchmarks for databases
  - RUBiS for multi-tier services
Benchmarks for Emerging Applications

- Benchmarks must evolve
  - Previous work: Static content benchmarks did capture dynamic content workloads
  - Open question: do dynamic content workloads capture web 2.0 workloads?
    [Nagpurkar-IISWC-2008, Lim-ISCA-2008]

- This study investigates an emerging web application
  - Do existing benchmarks capture its workload properties?
  - Are systems research solutions affected by this new workload?
Collaborative Web Applications

- Multi-tier web applications comprise a handful of scripts supplied by their developers

- Facebook, Google Docs, and SalesForce are different: 
  *They allow users to contribute content and scripts*

- The term *collaborative web application* reflects joint development between application designers and users
  - Benefit from the creativity of a large user base
  - Synonymous terms: web platform, utility
**Traditional Web Application**

1. Developers run code on a server
2. Users access developer-supplied functions
Collaborative Web Applications

1. Developers create a platform
2. *Users contribute scripts*
3. Users access user-supplied functions

Platform
- execution engine
- libraries
- data

Developer

Contributor

Scripts

End user

Execute "A"
Evaluation of a Collaborative Web Application

- We deployed a real collaborative web application with a real dataset and real trace
  - Do existing benchmarks represent collaborative web applications?
    Compare with traditional benchmarks
  - Are research solutions affected by this emerging application?
    Reevaluate previous research with a collaborative web application

Study the characteristics of collaborative web applications

Public release of real application, dataset, and trace
Outline

1. Motivation

2. WeBWoRK: A Real Collaborative Web Application
   - Introduction and Design
   - Real Trace and Dataset
   - Similar Emerging Applications

3. Comparison with Existing Benchmarks

4. Reevaluation of Past Research

5. Conclusion
Introducing WeBWoRK

- WeBWoRK is a web-based homework checker
  - Developed at the University of Rochester
  - Services 50,000 students at 80 universities worldwide
  - Teachers contribute problems
  - Students access problems and check answers

- Design Goals
  - Support a broad range of problem sets
  - Reduce the burden of teachers in grading
  - Ensure each student does their own work
WeBWoRK Design

- A problem is encoded in script
  - Written in Problem Generation (PG) Language
    - a variant of PERL
  - WeBWoRK is a platform that executes teacher-supplied scripts

- PG Scripts include:
  - A function that displays the problem
  - A function that checks answers
  - Number randomization to prevent copying
    - Same core problem (e.g., algebraic equations), but each student receives a unique version
      - *Student A receives:* \( 5x = 35 \)
      - *Student B receives:* \( 4x = 24 \)
Real Dataset and Trace

- Real traces of end-user behavior are especially important in collaborative web applications
  - Trace collected over 3 years at the University of Rochester
  - Dataset of 3,000 teacher-supplied problems

- Redeployed on local machines in the CS department
  - 2GHz Intel XEON processor
  - 2GB memory
  - Linux 2.6.10 with request context tracking [ASPLOS-2008]
  - All benchmarks run on this platform
Outline

1. Motivation

2. WeBWoRK: A Real Collaborative Web Application

3. Comparison with Existing Benchmarks
   - Traditional benchmarks
   - Clustering and Regularity
   - Inter-property Correlations

4. Reevaluation of Past Research

5. Conclusion
Traditional Benchmarks

RUBiS
- Implements core functions of an auction website
- J2EE-based multi-component
- Realistic nonstationary workload [USENIX-2008]

TPC-C
- Terminal operators issuing order-entry transactions
- Database centric, several transaction types

SPECweb
- Included in the paper
Experimental Setup

- We analyze request-level characteristics
  - A request is a common unit of work
  - Request-level properties are important in research solutions
    - [pai-asplos-1998]
    - [urgoankanar-osdi-2002]
    - [barham-osdi-2004]
    - [elinkety-eurosys-2007]
    - [stewart-eurosys-2007]
    - [lim-isca-2008]
    - [soundararajan-usenix-2008]
    - [stewart-usenix-2008]
    - and many others

- Patterns in the per-request CPU usage

- Correlations between CPU usage and system calls for each request
In RUBiS, clusters with similar CPU usage are obvious.

In WeBWoRK, no clear cluster boundaries.
In TPC-C, we observe a regular pattern in the CPU usage:
- Caused by a request type that depends on a random integer

In WeBWoRK, there is no clear pattern in the CPU usage.
In RUBiS, there is a strong correlation between system calls and CPU usage.

In WeBWoRK, there is no correlation at all.
Summary of Comparison with Traditional Benchmarks

- Request-level characteristics in WeBWoRK are different
  - Less clustered; more diverse CPU usage
  - Do not follow easily identifiable patterns
  - There is no correlation between properties

- These results makes sense:
  - Resource consumption depends heavily on user contributions
  - Large number of independent users injects randomness

- Do these results matter?
Outline

1. Motivation
2. WeBWoRK: A Real Collaborative Web Application
3. Comparison with Existing Benchmarks
4. Reevaluation of Past Research
   - Magpie-style Request Classification
     [OSDI-2004, ASPLOS-2008]
   - Request Mix Performance Models
     [Eurosys-2007, USENIX-2008]
5. Conclusion
Motivation: System management--- i.e., server consolidation and platform selection--- affects the bottom line of almost every firm in every industry.

Goal: Build performance models that can guide management for production web applications.

Insight: Requests of the same “type” have similar resource requirements.

Performance models parameterized by the mix and volume of request types.
Application to WeBWoRK

- Request mix model of CPU utilization

\[ U = B_0 + \sum B_j \cdot N_j \]

- Relative frequency of request types varies (i.e., the mix is nonstationary), which allows calibration from logs of request arrivals and CPU utilization

WeBWoRK comprises three request types

- Submit problem, access problem, and submit solution
- Calibrated with nonstationary 10-hour trace
- Evaluated on the next 10-hours (prediction)

Intuition: Aggregate CPU usage is a linear combination of the average usage per type
Results

- CPU utilization over time--- Each interval is 5 minutes

- Actual CPU utilization differs significantly from model based prediction

- Request mix models describe the variation in utilization for RUBiS---not for WeBWoRK

![Graph showing CPU utilization over time with RUBiS and WeBWoRK data points and an R^2 table]

<table>
<thead>
<tr>
<th></th>
<th>RUBiS</th>
<th>WeBWoRK</th>
</tr>
</thead>
<tbody>
<tr>
<td>R^2</td>
<td>0.90</td>
<td>0.25</td>
</tr>
</tbody>
</table>
Outline

1. Motivation
2. WeBWoRK: A Real Collaborative Web Application
3. Comparison with Existing Benchmarks
4. Reevaluation of Past Research

5. Conclusion
   - Future Work
   - Take aways
Open Problems

- How do we deploy collaborative web applications?
  - Maximize overall performance
  - Differentiated services

- How do we deploy collaborative web applications on top of collaborative web applications?
  - Facebook on Amazon EC2?

- Challenges for system management
  - Performance modeling is more difficult
  - Dynamic control in constant flux
  - Integration with traditional applications--- fall back?
Take Away Points

- Collaborative web applications are not well represented by existing benchmarks
  - Request-level characteristics are more diverse and less regular

- Previous research should be revisited in the context of collaborative web applications

- Need for benchmark innovation
  - As a first cut, our WeBWoRK setup is available
    http://www.cs.rochester.edu/u/stewart/collaborative.html