C-MART: Benchmarking the Cloud

A. Turner, A. Fox, J. Payne and H. Kim

Electrical and Computer Engineering
Carnegie Mellon University

Overview

- Introduction
- Related Works
- C-MART Overview
- Experimental Results
- Conclusions

Introduction

- Benchmark applications
  - Essential for system performance testing
  - Need to represent production environment
- Existing benchmarks
  - Not designed for Cloud Computing environments
  - May produce misleading results
- Cloud Computing Benchmarks
  - For Cloud management systems

Existing Benchmarks

- Designed for traditional dedicated hardware data centers
- Workload generators do not represent realistic user behavior
- Produce overly optimistic results
- Benchmarks fail to accurately validate Systems Under Test (SUT)

C-MART Introduction

- C-MART
  - New web application benchmark for the Cloud
- C-MART design principles
  - Scalability
  - Modern technologies
  - Flexibility
  - Client Realism

C-MART Introduction

- Scalability
  - Cloud computing allows on-demand resource provisioning
  - C-MART can horizontally scale at every tier
  - Deployment server and scaling API
- Modern Technologies
  - JavaScript, CSS, AJAX, HTML5, SQLite
- Flexibility
  - Run with different architectures, applications
- Client Realism
  - Include complex client behavior, variable typing speeds, think times, QoS expectations, page transition probabilities
  - Real-world distributions

Comparison to Existing Benchmarks

Feature	Current	C-MART	Use case	Impact
Client Caching	Low/ None	High/SQLite	Load balance by request URL	Current benchmarks: Low var in response time, ± 138 ms C-MART: High var in response time, ± 6100ms
Page access frequency	Static	Variable	Linear regression scheme to predict CPU utilization based on request rate	Current benchmark: Regressions scheme has 4.4% error C-MART: Regression scheme has 50% error Resources are underprovisioned
Session based QoS	No	Yes	Profit based on clients completing browsing sessions	Current benchmark: no QoS in decision making process and management is based on aggregation across clients C-MART: Individual client QoS decisions, causes client levels to change with open loop client arrivals Proper resource provisioning for QoS
Page Content Variability	Low	High	Consolidating VMs based on average CPU utilization	Current benchmark: Violates SLA 0% of the time C-MART: Violates SLA 22% of the time Consolidation scheme performs poorly in production system

Existing Benchmarks

- RUBiS
  - Auction website modeled after eBay
  - TCP connections shared between multiple clients
  - Think times: single exponential distribution for all pages
  - No CSS, Javascript
  - Limited multimedia content
  - Closed loop client generator
  - Non-scalable SQL database

Existing Benchmarks

- TPC-W
  - Web benchmark emulating online bookstore
  - Designed for hardware benchmarking
  - Workload generator similar to RUBiS
  - No client generated data
  - Does not represent Web 2.0 application
  - Turns off caching to make up for lack of multimedia
  - Primary metric is Web Interactions Per Second (WIPS)
    - Not relevant Cloud metric

Existing Benchmarks

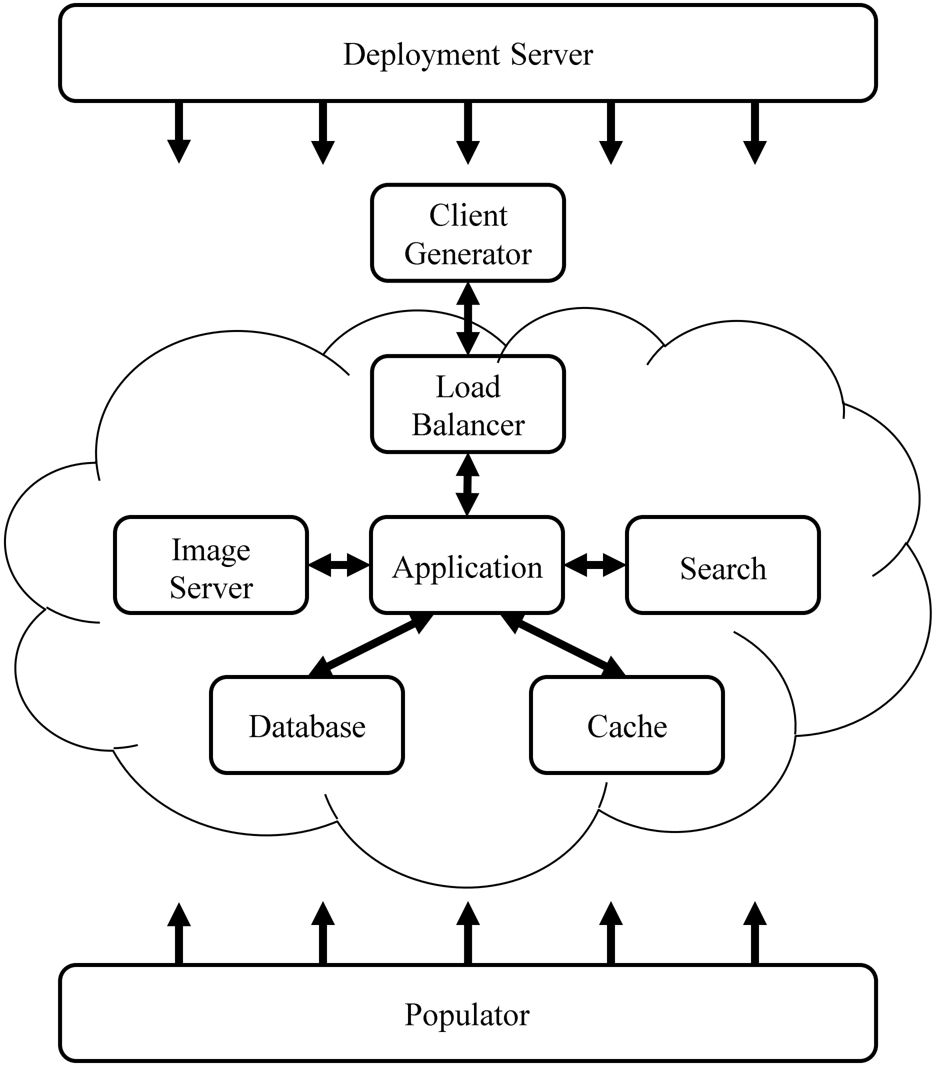
- Olio
  - Social event web calendar
  - Single think time distributions
  - Static probability matrix for page transitions
  - Non-scalable SQL database
- SPECweb2009
  - Benchmark for measuring power efficiency
  - Simulates backend database
  - Closed-loop workload generator

Existing Benchmarks

- CloudSim
  - Simulation only
  - No SQL in application model
  - Simplistic workload model
- YCSB 2010
  - Does not emulate application, benchmarks different databases
  - Does not account for interactions between application tiers
  - Can not extrapolate application level performance from this data

C-MART

- Online auction and shopping
  - Future extension to video-on-demand
- Multi-tier
- Deployment, scaling API
- Custom workload generator



Scalability

- Cloud allows for on-demand resource provisioning
- Horizontally elastically scale at every tier
- Stateless application tier

Tier	Description
Client	Run on multiple hosts, consolidates statistics
Load Balancer	Clients can be directed to multiple load balancers, similar to DNS balancing system
Application	Stateless, does not lock shared resources
Cache	Memcache Distributed Hash Table
Search	Solr Multiple read-only copies
Image	MongoDB replicas
Database	Cassandra replicas

Dynamic Scalability and Deployment

- Automatic scaling
- API defines VMs
  - Application tier
  - IP addresses
  - Hot/cold backups
  - CPU, RAM allocations
- Allows users to create custom data center management algorithms
- Easy to define additional servers and change resource allocations

```
<Host hostOS="Fedora" port="4444‶ IP="10.66.1.3">
   <VirtualMachine IP="10.1.4.3">
     <OSType>Fedora</OSType>
     <VMType Type="Application" Backup="Hot"/>
     <UserName>root</UserName>
     <Password>cmart</Password>
   </VirtualMachine>
   <VirtualMachine IP="10.1.1.7">
     <OSType>Fedora</OSType>
     <VMType Type="SQL" RAMSIZE="512"/>
   </VirtualMachine>
</Host>
```

Modern Technologies

- HTML5, AJAX, CSS, SQLite, Multimedia
  - Impact resource utilizations
    - Server side
    - Client side
  - Changes caching requirements
  - Increases variability
    - Requesting same page at different times results in different resource utilizations
  - Perform functions at client side that traditionally required request to server
  - Periodic or non-user initiated AJAX requests
    - Only return modified data in response

Modern Technologies

- Real World Distributions
  - Sampled 100,000 eBay auctions to create empirical distributions
    - Number and frequency of words in products titles and descriptions
    - Number of items in each product category
    - Number of images on product page
    - Seller ratings
    - Product bid and buy now prices
  - Distributions used to differentiate between items
  - Create natural hot spots

Flexibility

- Tier configuration
  - Two-tier to six-tier architecture
  - Different architecture options for different tiers
  - Database populators provided

Flag	Effect
Solr	On/Off to use Solr as the sites search engine
Cache	0, 1, 2 use Memcache to cache either none, database heavy query results or all results
Database	MyISAM, InnoDB for MySQL storage, Cassandra for NoSQL
Image	'img' for local storage, 'netimg' for NFS, 'mongoDB' for GridFS/MongoDB
Web	Web 1.0 or Web 2.0 enable HTML5, SQLite, AJAX, JavaScript

Flexibility

- Web Design Technologies
  - Two implementations
  - Implementation 1
    - SQLite, AJAX, JavaScript, render most pages at client side
  - Implementation 2
    - More traditional client-server model
- Multiple Application types
  - Auction
  - Shopping
  - Video-on-demand

Flexibility

- Client Flexibility
  - Different operating modes, variable complexity
    - Open-loop/closed-loop
    - Random client bursts
    - Read/write heavy
    - Markov chain page transitions/content, history-based decisions
    - User-defined client satisfaction levels

Client Realism

- Clients are unique
  - Variability in resource utilizations, performance expectations
- Content and history based client decisions
  - Content on item page defines its appeal
    - Description length
    - Number/quality of pictures
    - Seller rating
    - Time to auction expiry
  - Clients consider their personal history
    - Users are more likely to bid on items they have already bid on
  - Creates natural hot spots in website

Client Realism

- QoS-based user decisions
  - Amazon loses 1% of revenue for every 100 ms of additional delay in response time
  - Response Times
    - View more pages on fast websites
    - Likelihood of obtaining information to make purchase
    - Clients leave slow websites
  - QoS focus of Cloud management
    - Client QoS reactions must be modeled in workload generator
- Modern Browsers
  - Tabbed browsing
  - Form fillers, typing speed, typing errors
  - Caching

Performance Metrics

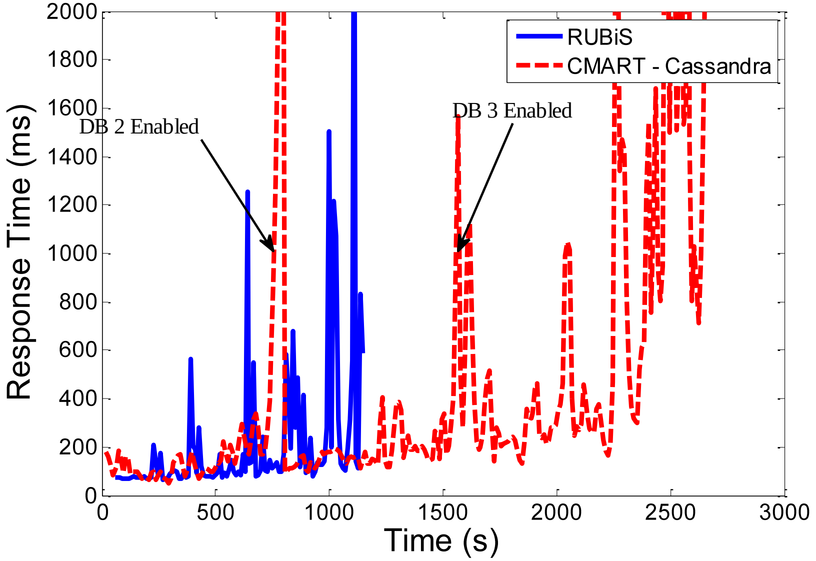
- Metrics must be relevant to system evaluation
- Web Interactions Per Second
  - Better suited for evaluating hardware than Cloud Management
- C-MART metrics
  - Distributions of response times per page type
  - Client session data
  - Server/VM resource utilization levels
  - Webpage statistics

Cloud Management Systems

- VM placement, migration, energy-optimization, SLA and QoS targets
- Basic management systems
  - Model application response time as function of resource utilizations (CPU, memory, I/O, etc.)
    - Linear regression, Neural networks
    - Autoregression
  - Model response time as function of request types
    - Requests do not necessarily use the same amount of resources on each access

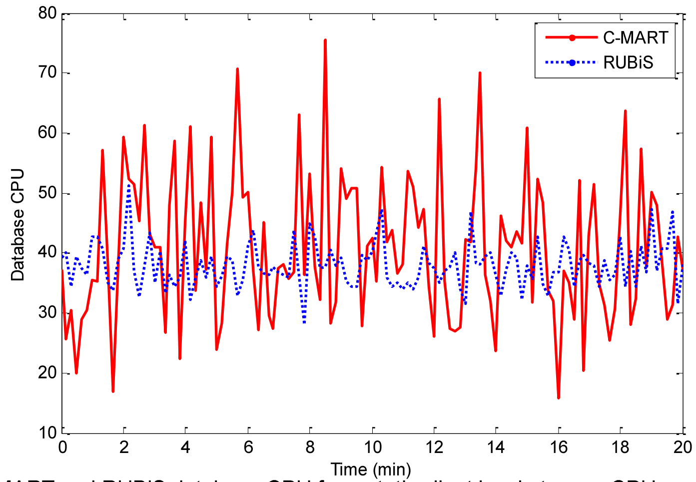
Application Scalability

- Cloud allows for elastic scaling and on-demand resource provisioning
- Scaling prevents any one tier from becoming performance bottleneck
- 95th percentile response time for C-MART and RUBiS as workload is increased
  - In C-MART new Cassandra instances can be activated to keep system from crashing

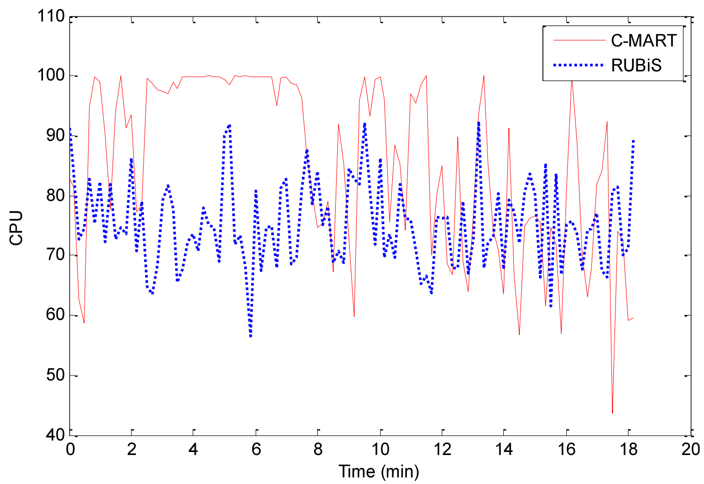


VM Consolidation

- When consolidating VMs, all VMs must still receive sufficient resources
- Easier when VM uses static amount of resources
  - Unrealistic for real application
- C-MART and RUBiS CPU
  - Both average 40%
- Two VM consolidation
  - C-MART has many more CPU violations



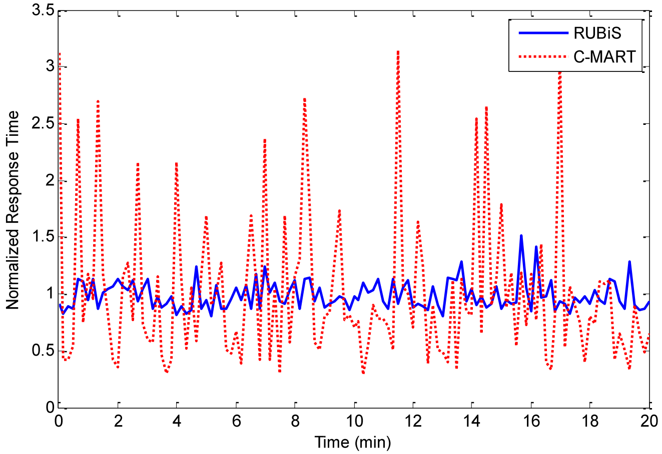
C-MART and RUBiS database CPU for a static client level at same CPU average



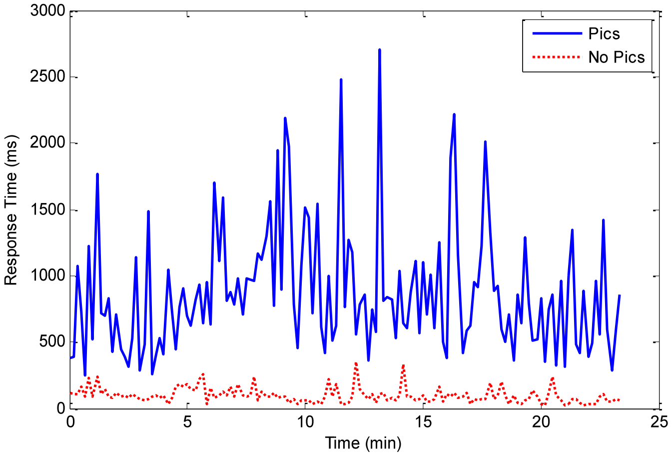
CPU of two consolidated instances of C-MART and RUBiS

Response Time

- Response times difficult to control
  - Much higher variability
- Response times for static client
  - Variance 20x higher in C-MART than RUBiS
- Response time for Multimedia content, modern technologies
  - 10 to 30 times higher



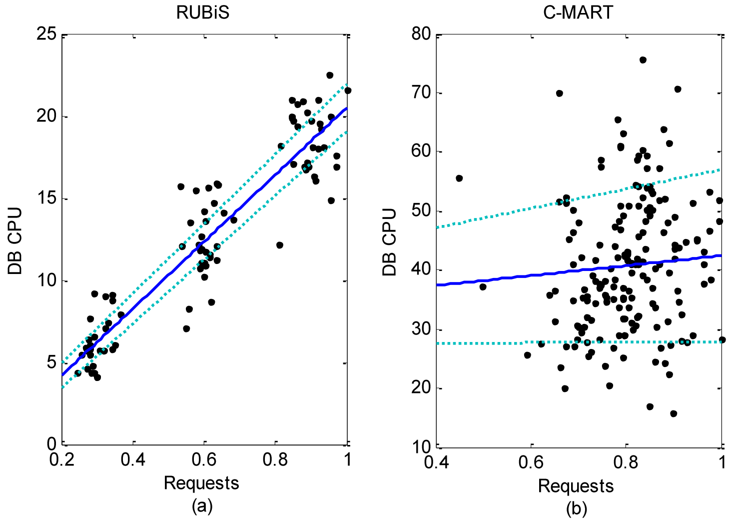
C-MART and RUBiS response times for a static client level. For comparison purposes, each response time was normalized to the average response time from the respective experiments



C-MART Response Times when Pictures, CSS, JavaScript are and aren't downloaded

Performance Prediction

- Use request levels to predict response time and resource utilizations
  - Effective only if
    - ratio of incoming request types is predictable and
    - resource utilization for request types is constant
- Resource utilization levels easier to predict in RUBiS than C-MART

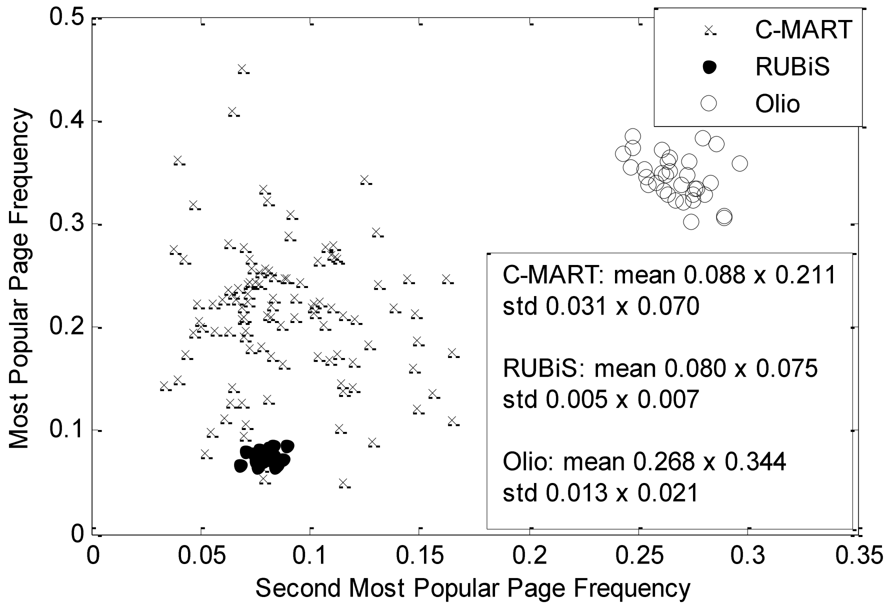


CPU Prediction based on workload for (a) RUBiS and (b) C-MART

$$CPU_{\text{RUBiS}} = (20.4 \pm 0.9)\lambda + (0.1 \pm 0.6)$$
$$CPU_{\text{C-MART}} = (18 \pm 9)\lambda + (26 \pm 8)$$

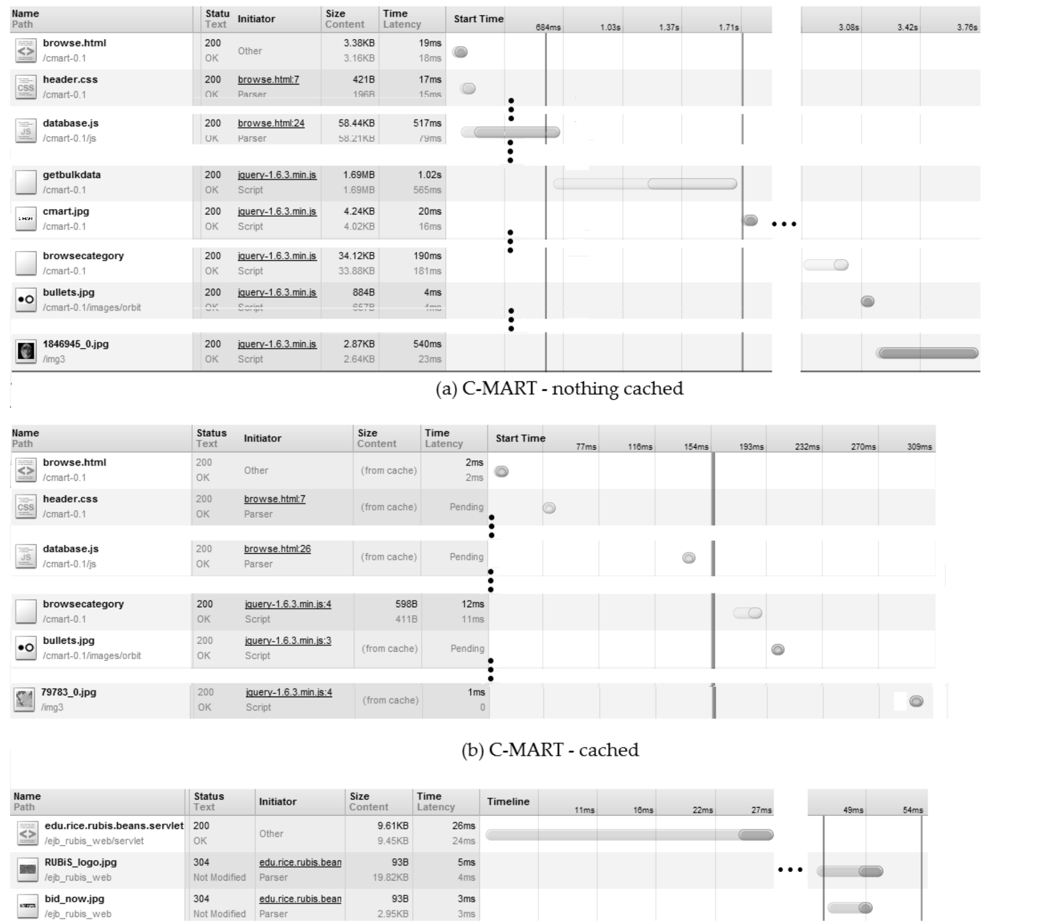
Client Behavior

- Non-stationarity
  - Real web applications do not have constant ratio of request types over time
- Result of complex client decision making process and unique clients
- Website may appear much more predictable in RUBiS



Caching and SQLite

- Changes amount of data requested from server
- C-MART two identical requests
  - Nothing cached
    - 3.76 s response time
    - 51 elements in request with 1.69 MB file to initially populate SQLite
  - Data already cached
    - 309 ms response time
    - 47 elements in request
    - Due to cache, list of items on page reduced from 34 kB to 598 B
- RUBiS
  - Only 3 elements in request
    - Identical on every request



(a) C-MART - nothing cached

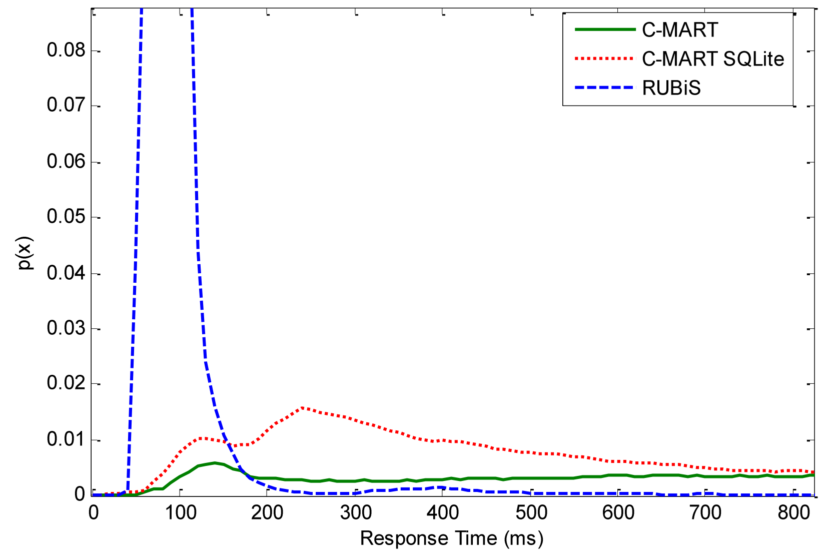
(b) C-MART - cached

(c) RUBiS

Fig. 11: Network view of loading the Browse page of (a) C-MART when nothing has been cached, (b) C-MART after files have been cached and SQLite database has been populated, (c) RUBiS

Caching and SQLite (cont.)

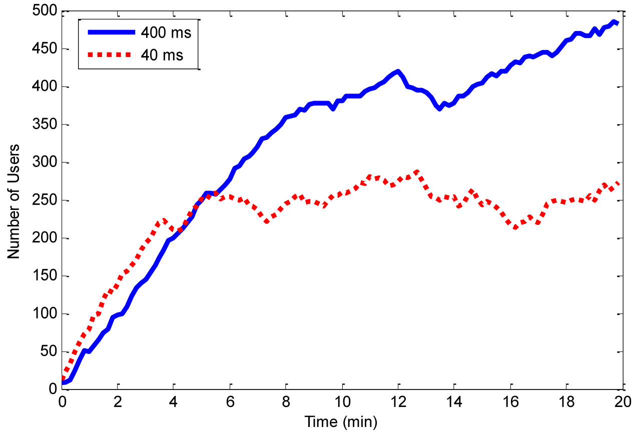
- Creates response time distribution with heavy tail
- Multiple peaks for SQLite due to cache
- Predictive large peak for RUBiS



Item page response time distributions for C-MART with and without SQLite, and RUBiS

QoS Measurement

- QoS measured per-user, not aggregate basis
- Open-loop client generator accounts for QoS effects much better than closed-loop
- Different response time expectations changes length of client session and final workload levels



User load for different Response Time expectations with an open-loop client

Situation	Percent of Clients that leave due to poor QoS	Average Client Session Length (s)
400 ms Response Time Threshold, Per-client QoS, Open Loop	19.5%	333
40 ms Response Time Threshold, Per-client QoS, Open Loop	48.4%	201
40 ms Response Time Threshold, Per-client QoS, Closed Loop	57.7%	171
40 ms Response Time Threshold, Aggregate QoS, Open Loop	19.8%	239

Conclusions

- Existing benchmarks
  - Inappropriate for benchmarking Cloud systems
  - Misleading results (usually results in under-provisioning)
- C-MART
  - Benchmark application for the Cloud
  - Scalable
  - Modern Web Design Technologies
  - Flexible Architecture
  - Realistic Client/Workload Generator
- theone.ece.cmu.edu/cmart