Combining Structured and Unstructured Data for Web Application Performance Analysis
Performance Analysis Goals

- Determine CPU utilization of executing applications
- Determine Memory utilization,
- Network,
- Hard Disk, etc.
Performance Analysis Goals

- Determine CPU utilization of executing applications
- Determine Memory utilization,
- Network,
- Hard Disk, etc.
- Focus on CPU - assume other analyses can be performed with similar tools
Performance Analysis

- Typical Tools include:
  - Task Manager
Performance Analysis

- Typical Tools include:
  - Task Manager
  - Top
Performance Analysis

- Typical Tools include:
  - Task Manager
  - Top
  - Visual Log Parser

![Image of performance analysis tools](image.png)
Desktop Applications

Applications

OpenOffice.org 2.0

Mozilla Firefox

Apache

User

Works well with typical performance analysis tools!
Web Applications

Web Application Server

Web Applications

Users
Revised Performance Analysis Goals

- In a web application environment, we want to:
  - Determine the CPU utilization of each application over a specified period of time.
Web Applications

- How typical performance analysis tools can help us
  - Top can tell us the CPU utilization of the apache process, httpd.
  - Log parsing can give us an indication of the "Activity" of a particular application.
Web Applications

- Shortcomings of performance analysis tools
  - Top/Task Manager only show process-level statistics
  - Log Parsers can provide more detailed information, but are disconnected from CPU utilization information.
Problem Approach

- Examine structured data from Top to determine periods of high overall CPU utilization.
- Examine unstructured data from logs to determine activity level of applications.
- Combine the two to establish a relationship between CPU utilization and application activity.
Challenges

- How should data from Top be examined?
- Which logs should be examined?
- How should data from logs be examined?
- How can the two be combined?
How should data from Top be examined?

- Use Top to sample the CPU utilization of the apache process over a period of time.
- For each sample, insert the CPU utilization and the time of the sample into a database.
- Query the database to determine times when the CPU utilization is over a given threshold.
Which logs should be examined?

- Logs are mostly unstructured data.
  - Assume that each log is a sequence of lines of text where each line is an event.
  - Define an "interesting" event to be one that has a timestamp and refers to a web application.
- Examine logs that have at least one interesting event.
How should data from logs be examined?

- For a given period of time, determine the activity of each application.
- Count how many times each application is referenced in the log in that period.
- Scale counts against average events-per-request statistics.
App A contributed 4 events to the log!

But App B only contributed 2 events to the log.

App A generates more events per request on average than App B.
Application Activity Scaling

How can the average events per request be measured?

- Run a representative sample of requests for an application.
- Count the number of interesting events.
- Divide that count by the number of requests.
- Repeat for each application.
How can the two be combined?

- Time is the key.
- Determine what times the CPU was high.
- Query the logs in that time period to determine which application is most active.
- For this to work, we need to make a few assumptions.
How can the two be combined?

Assumptions:

- The number requests to an application and overall CPU utilization are highly correlated.
- CPU impact of each application per request is equivalent.
Experimental Design

- Create an apache web server and install 4 PHP applications.
  - Wiki
  - Image Gallery
  - Calendar
  - Forums
Experimental Design

- Use Selenium to simulate user traffic to these applications with varying degrees of activity.
  - Scale of 0 to 4 with 4 being the most active.

- Run 4 Experiments with different activity parameters
- For each experiment, chose an application. Set it's activity to 4.
- For all other applications, use activity level 1.
Experimental Results – Wiki

Activity of Web Applications over Time

vertical axis: Requests
horizontal axis: Time in seconds

(There isn't a very clear winner here)
Averaged Activity of Web Applications over Time

vertical axis: Requests
horizontal axis: Time in seconds
Experimental Results – Wiki

- Average each series down to a single value.
- Provides objective criterion for CPU utilization attribution.

Percent of CPU Attributed to Applications

- Calendar
- Wiki
- Gallery
- Forums

vertical axis: % CPU Utilization
Experimental Results - Calendar

Averaged Activity of Web Applications over Time

Percent of CPU Attributed to Applications

- Calendar
- Wiki
- Gallery
- Forums

vertical axis: % CPU Utilization

---

vertical axis: Requests
horizontal axis: Time in seconds
Experimental Results - Gallery

Averaged Activity of Web Applications over Time

vertical axis: Requests
horizontal axis: Time in seconds

Percent of CPU Attributed to Applications

vertical axis: % CPU Utilization
Experimental Results - Forums

Averaged Activity of Web Applications over Time

Percent of CPU Attributed to Applications

vertical axis: % CPU Utilization

vertical axis: Requests

horizontal axis: Time in seconds
Conclusions

- Experimentally, the methods described have worked relatively well.
- Each experiment was conducted with similar parameters, but the results were not very similar.
- This suggests our assumptions may not be totally valid.
Questions?