ISTC-CC Presentation by Georgia Tech (8/27/15)

Karsten Schwan (site director), ***Calton Pu*** (presenter), with L. Liu, S. Yalamanchili, G. Eisenhauer, A. Gavrilovska, M. Wolf
Georgia Tech

http://www.istc-cc.cmu.edu/

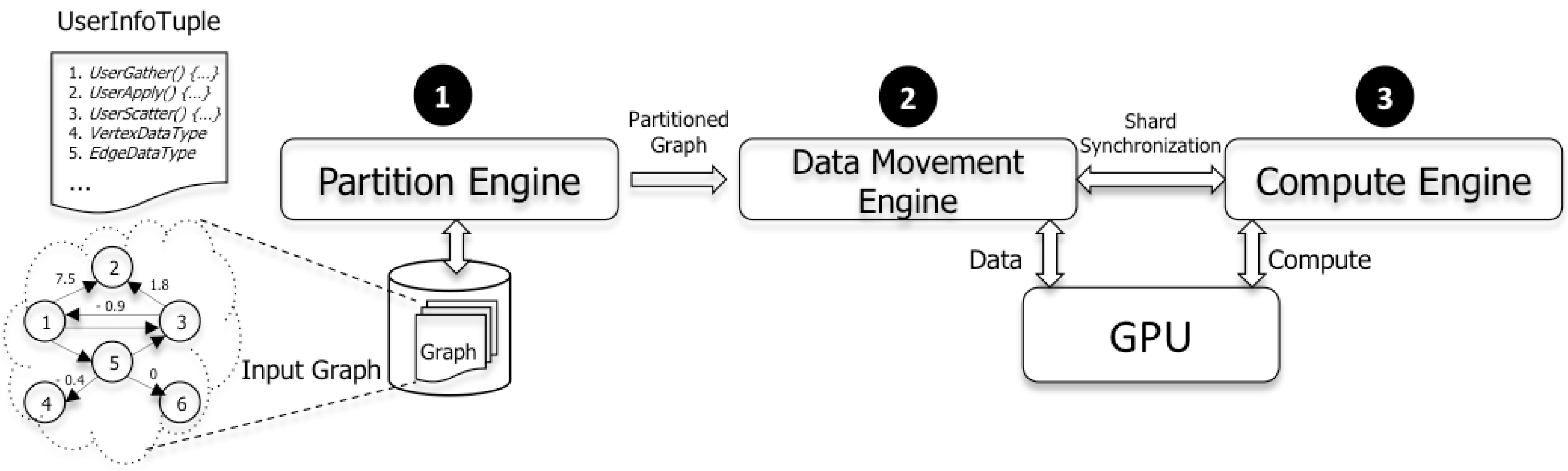
Intel Science & Technology
Center for Cloud Computing

Highlights of ISTC-CC (Georgia Tech)

- First part: yearly update on various projects
  - Karsten Schwan, Ling Liu, Calton Pu, Sudha Yalamanchili, Greg Eisenhauer, Ada Gavrilovska, Matt Wolf
  - Many students at PhD, MS, and undergraduate levels
- Second part: some details on automated management work (linking into Project Pulse)
- Highlights on work funded by ISTC-CC, or conforming to Intel open IP policy
  - Significant funding amplification from many industry collaborators and government funding agencies such as NSF and DoE

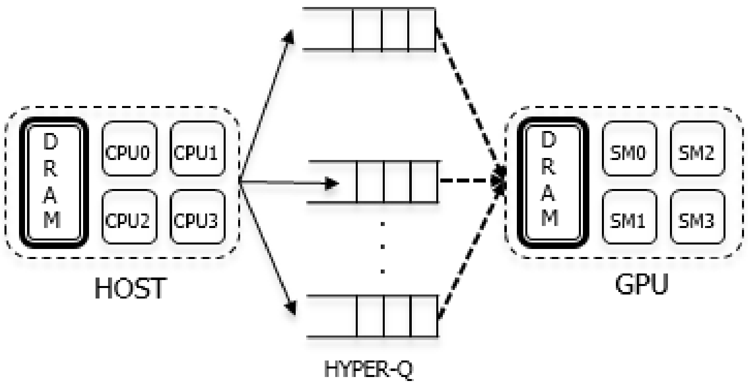
Intel Science & Technology Center for Cloud Computing

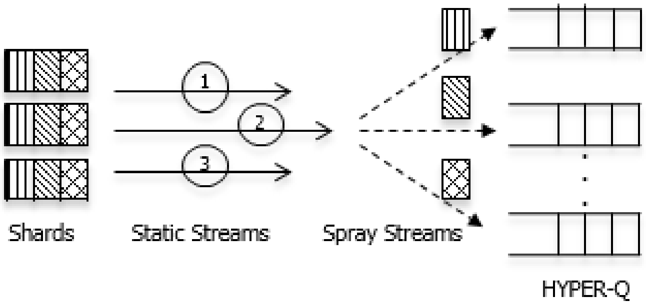
GraphReduce Architecture (Schwan)



SC 2015, GraphReduce: Processing Large-Scale Graphs on Accelerator-Based Systems

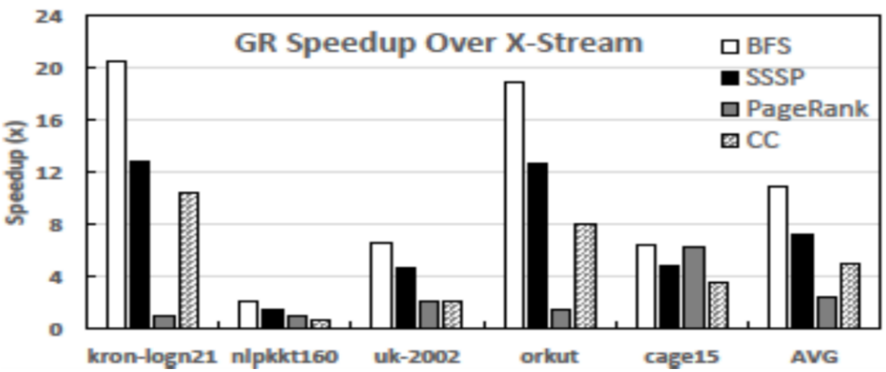
Optimizations on GraphReduce

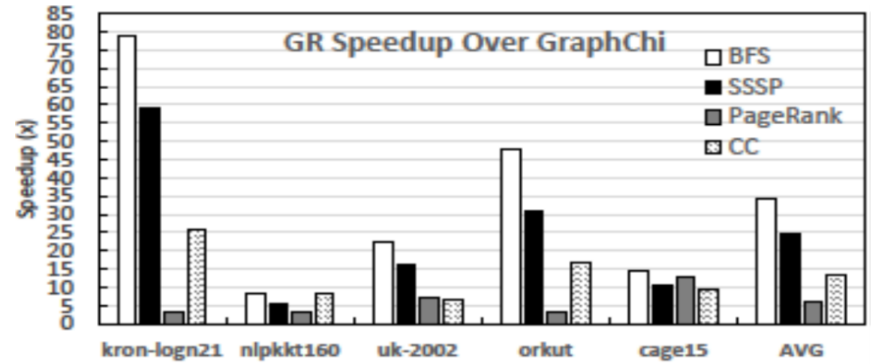




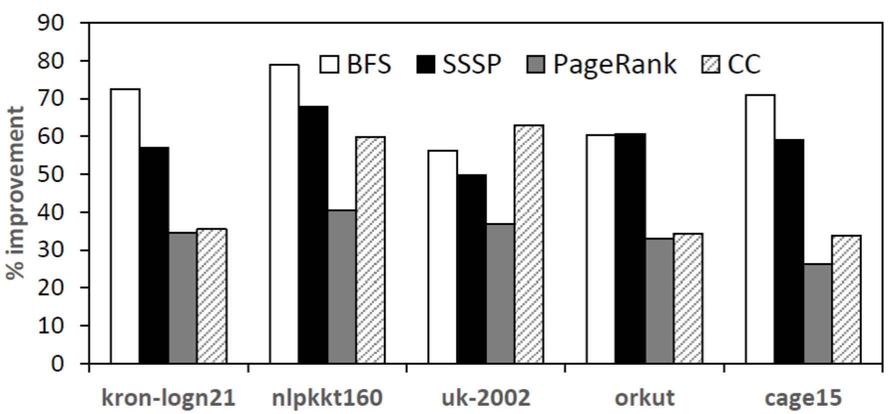
- Asynchronous execution and Spray (deep-copy) operation
- Dynamic frontier management
- Dynamic phase fusion and elimination

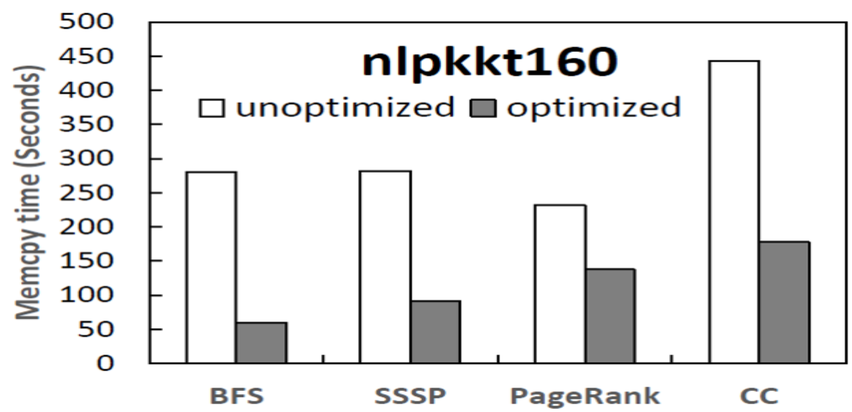
Gains by GraphReduce





GraphReduce's speedup over Graphchi and X-Stream for out-of-memory graph inputs





Benefits of GraphReduce optimizations over memcpy time

Understanding Issue Correlations: A Case Study of the Hadoop System

Scalable distributed systems are complex [Yuan et al., OSDI'14]

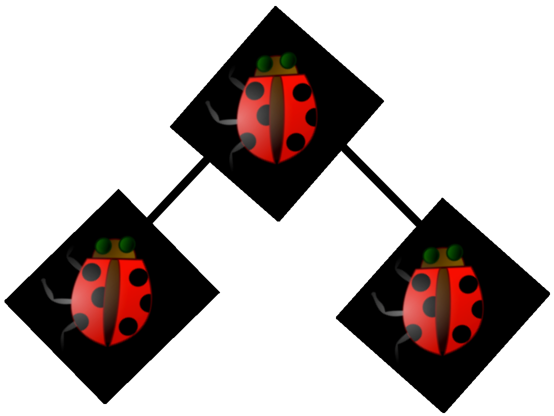
Complicated System + Error-prone + Hard to Debug

⇩ Issue Study

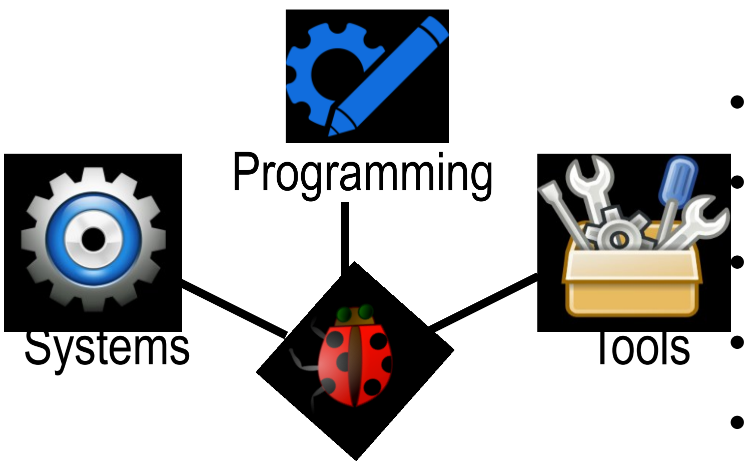
Issue Pattern → Better Software & Debugging Tools

SOCC'15

Our Findings

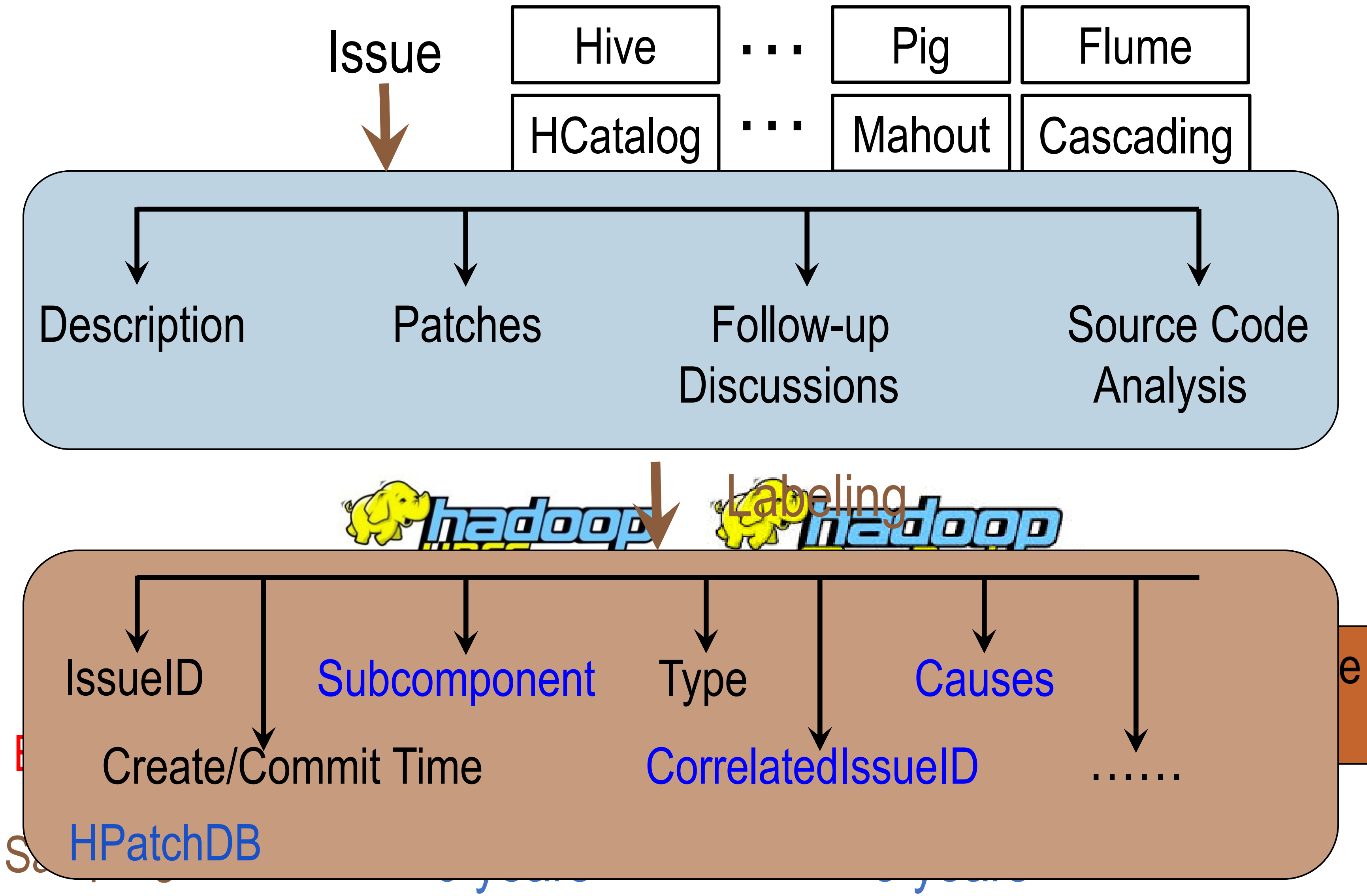


- Half of the issues are independent
- The issue correlations are not complex as we expected
- One third of the issues have similar causes
- ......

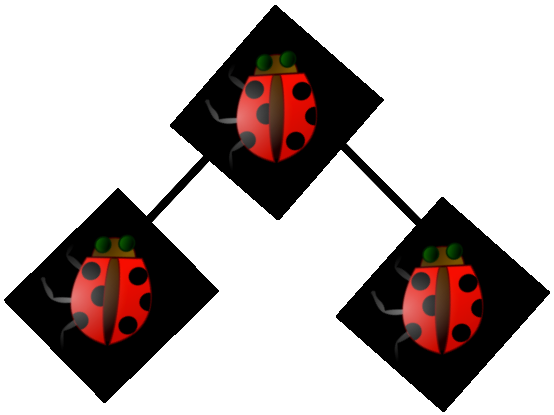


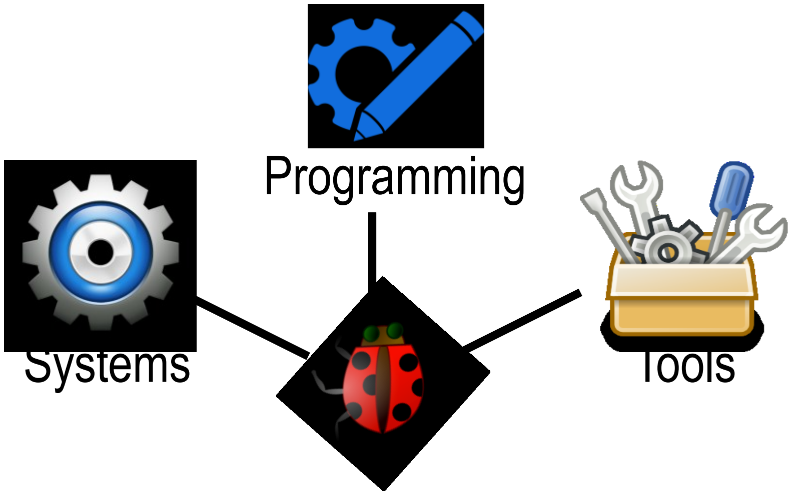
- For memory issues, GC is still the No. 1 concern
- The statement "99.99% of data reliability" is challenged
- One third of programming issues relate to interfaces
- The logging system is error-prone
- ......

Methodology Used in Our Study



Insights from Analyzing Issues





1 **Correlations Between Issues**

Issues are independent; 33% of issues have similar causes, etc.

2 **Correlations With System Characteristics**

Systems, programming, tools

Big Data Systems Research (L. Liu)

- Optimizations for Fast Iterative Graph Computations
  - GraphLego:
    - Resource Aware Graph Parallel Abstractions (Graph Cube and Slice, Strip, Dice) [ACM HPDC 2015]
  - GraphTwist:
    - Approximation with utility-aware pruning [VLDB2015]
    - Edge pruning by slices: removing some insignificant edges
    - Vertex pruning by cuts: removing some insignificant vertices
  - GraphMap:
    - Workload aware Distributed Graph Processing Framework [IEEE SC2015]

Shared Memory Optimization in Virt. Cloud

- Shared Memory Management Mechanisms
  - MemPipe:
    - Shared memory channels for improving communication efficiency between co-resident VMs
    - Incremental shared memory management
  - MemFlex
    - Shared memory based ballooning (inflate and deflate)
    - Shared memory based optimization for memory page fault.
  - MemMon
    - Memory working-set monitoring and estimate.

Poster by Qi Zhang (L. Liu)

Optimizing Performance and Productivity on Heterogeneous Processors

Sudhakar Yalamanchili
School of Electrical and Computer Engineering
Georgia Institute of Technology

Collaborators: H. Wu, M. Gupta, C. Kersey, H. Kim, I. Saeed, J. Young, H. Wu, and LogicBlox Inc.

http://www.istc-cc.cmu.edu/

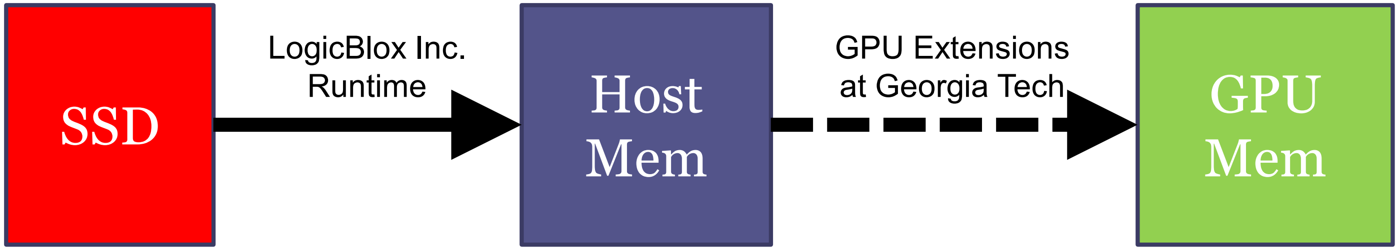
Intel Science & Technology Center for Cloud Computing

Accelerating Relational Processing

Haicheng Wu and S. Yalamanchili

- Finding cliques
  - triangle(x,y,z)<-E(x,y),E(y,z),E(x,z), x<y<z.
  - 4cl(x,y,z,w)<-E(x,y),E(x,z),E(x,w),E(y,z),E(y,w),E(z,w), x<y<z<w.

Multi-predicate Join

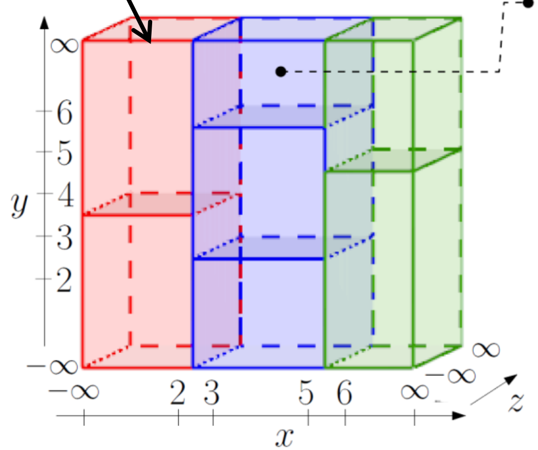


- Relational computations over out of core data sets
- Implementation of multi-predicate join for graph processing using GPUs
  - 3-clique and 4 clique problems

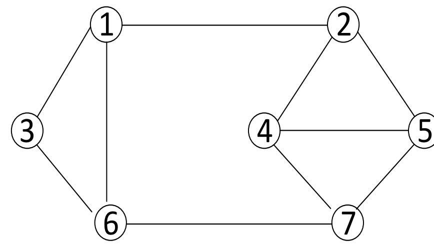
H. Wu, D. Zinn, M. Aref, and S. Yalamanchili, "Multipredicate Join Algorithms for Accelerating Relational Graph Processing on GPUs," *Proceedings of ADMS*, September 2014

Out-of-Core Data Management

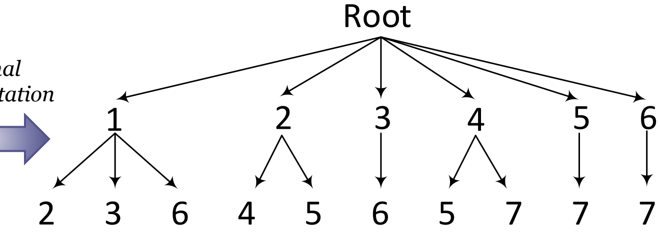
Data partitions (boxes) to fit in memory (worst-case optimal)



Box [3⩽x⩽5,6⩽y<∞,-∞<z<∞]

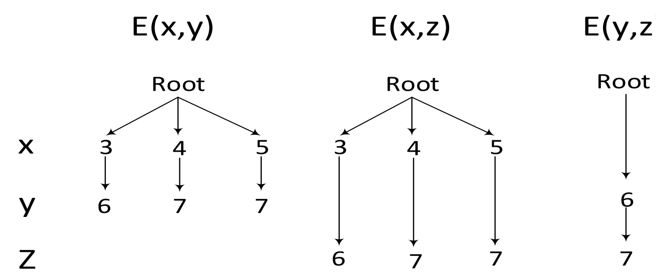


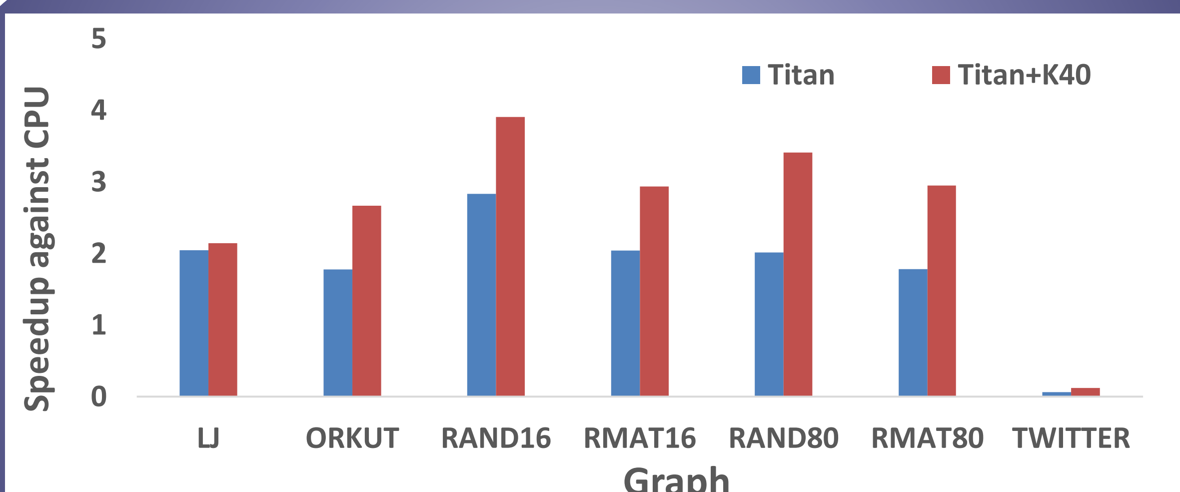
Internal Representation



Trie representation of edges

triangle(x,y,z)<-E(x,y),E(y,z),E(x,z), x<y<z



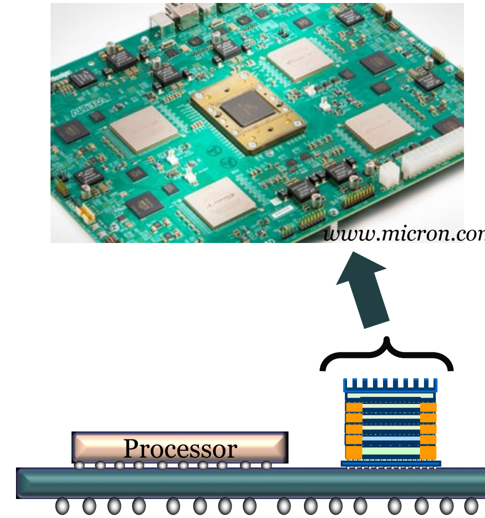


- Large, out of core graphs
- Baseline is CPU boxed multi-predicate join
- SSD and PCIe are not the bottlenecks

Near Memory Data Intensive Computing

Kim (CS), Mukhopadhyay (ECE), Yalamanchili (ECE)
Collaborative Discussions with Intel Labs (N. Carter)

- Move Analytics Primitives (RA) into the memory system
  - Custom low power GPU(Harmonica)
  - Progress on Base compiler for in-memory GPU



A. Gavrilovska (CS), K. Schwan (CS), Yalamanchili (ECE)

- Technology Assessment
  - Collaboration with Lexis Nexis
  - Assess the impact of In-memory acceleration for HPCC

HPCC SYSTEMS®

LexisNexis®

HPCC Engineering Summit

Intel Science & Technology Center for Cloud Computing

Leveraging eBoxes and Compilers

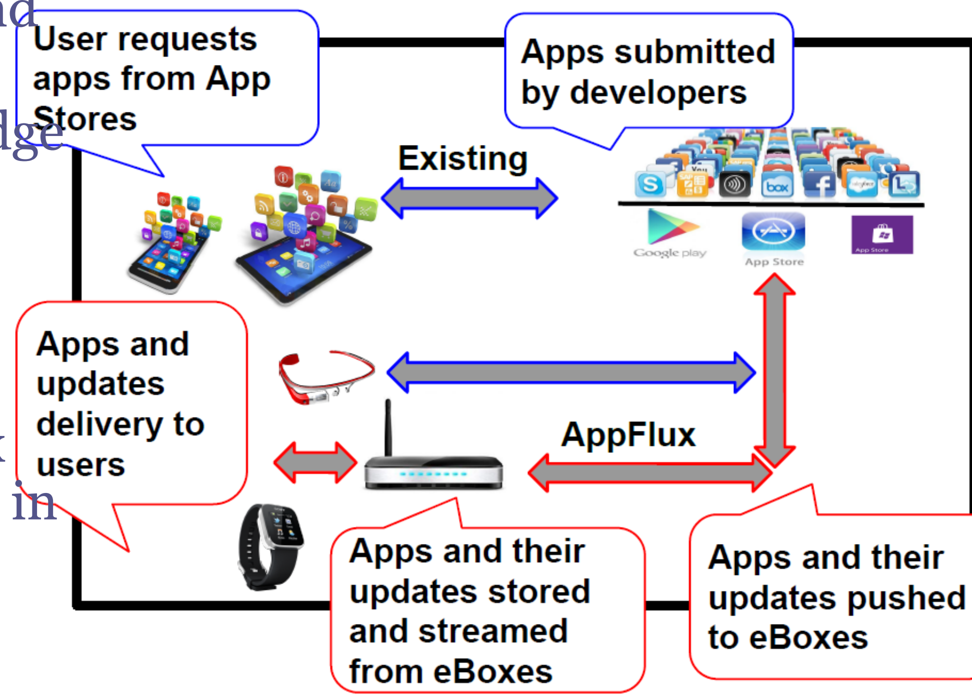
Ada Gavrilovska
Georgia Tech

http://www.istc-cc.cmu.edu/

Intel Science & Technology
Center for Cloud Computing

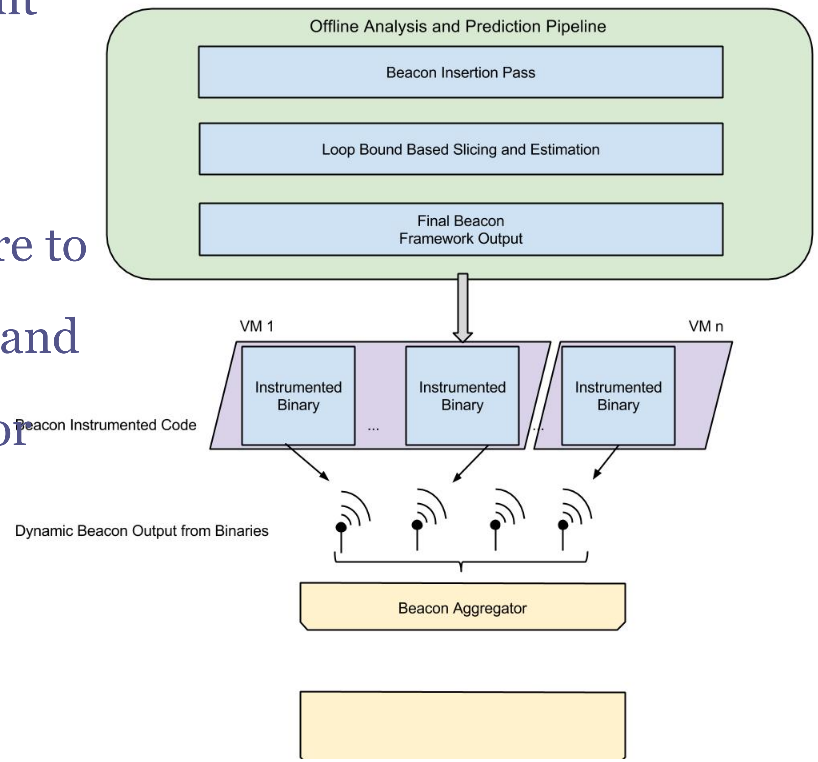
Enhancing the Edge via eBox-based Services

- Some results presented last year, also supported by ISTC-EC in the past and VMware
- Leverage high-density/low-power edge boxes – eBoxes
- Infrastructure for app streaming, caching, ephereral app delivery;
- Fully integrated in Android stack
- Outcome: 2x faster app delivery, 10x faster app descovery, 70% reduction in traffic; no performance impact
- AppFlux: Taming App Delivery @TRIOS'15 (Bhardwaj, Agarwal, Gavrilovska, Schwan); others in submission/preparation



Compiler-Assisted Resource Management

- Goal: dynamic resource allocation to concurrent workloads/workload components
- Problem: profile-based techniques limited effectiveness (input-dependent requirements, irregular applications…)
- Approach: LLVM-based compiler infrastructure to instrument binary with "**beacons**". Beacons generate information based on dynamic input and actual execution path taken. Intercepted by resource managers (e.g., VM manager, VCPU or thread scheduler, runtime-level scheduler…)
- Outcome: improved workload performance, reduced performance variability, improved resource use and management efficiency
- Compiler-assisted Load Balancing on Large Clusters @PACT'15 (Deodhar, Parikh, Gavrilovska, Pande); others in submission/preparation



Intel Science & Technology Center for Cloud Computing

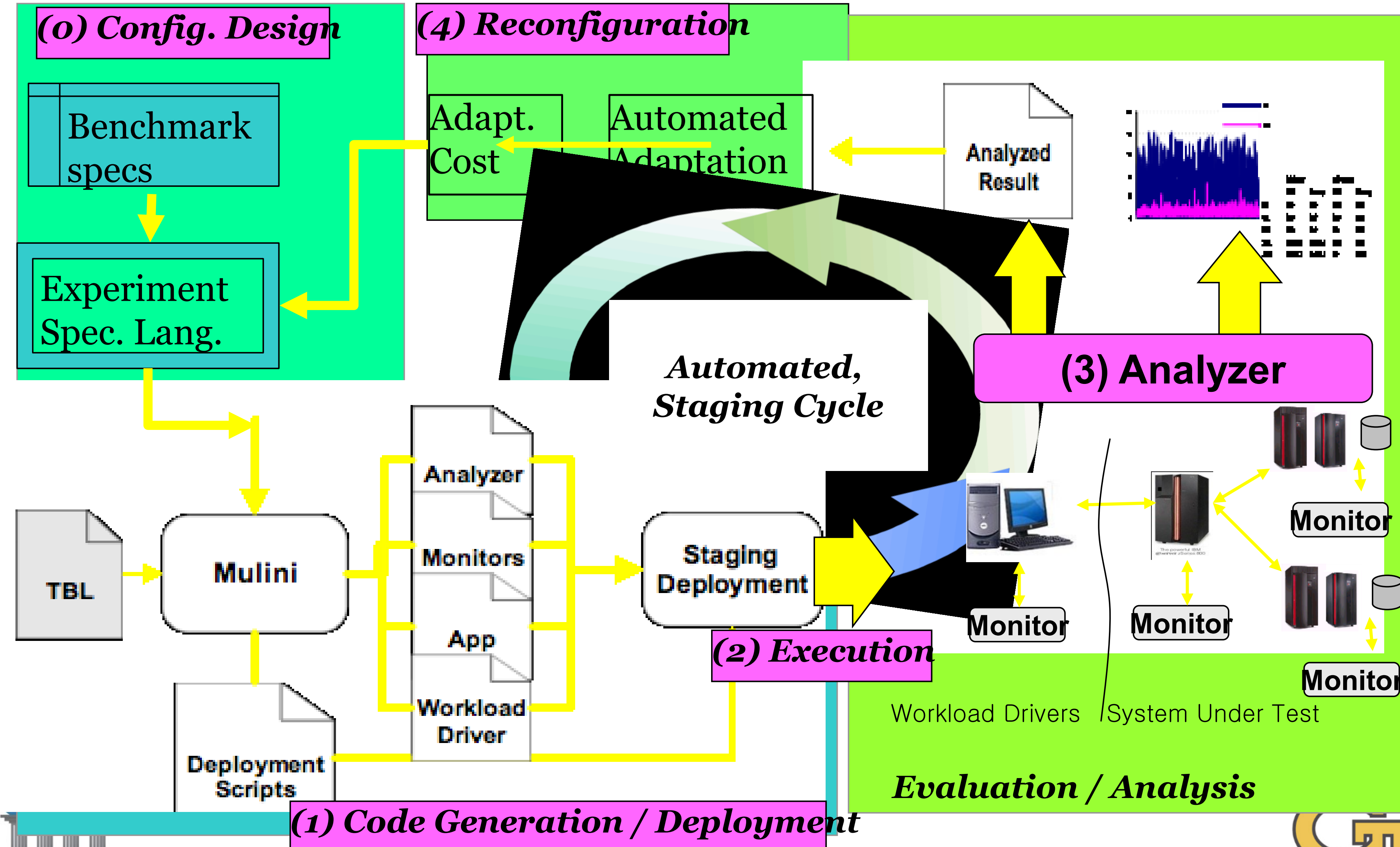
Automated Cloud Management through Experimental Measurements

Calton Pu
Professor and J.P. Imlay Chair in Software
Georgia Institute of Technology
Many PhD, MS, Undergraduate students and industry collaborators

http://www.istc-cc.cmu.edu/

Intel Science & Technology Center for Cloud Computing

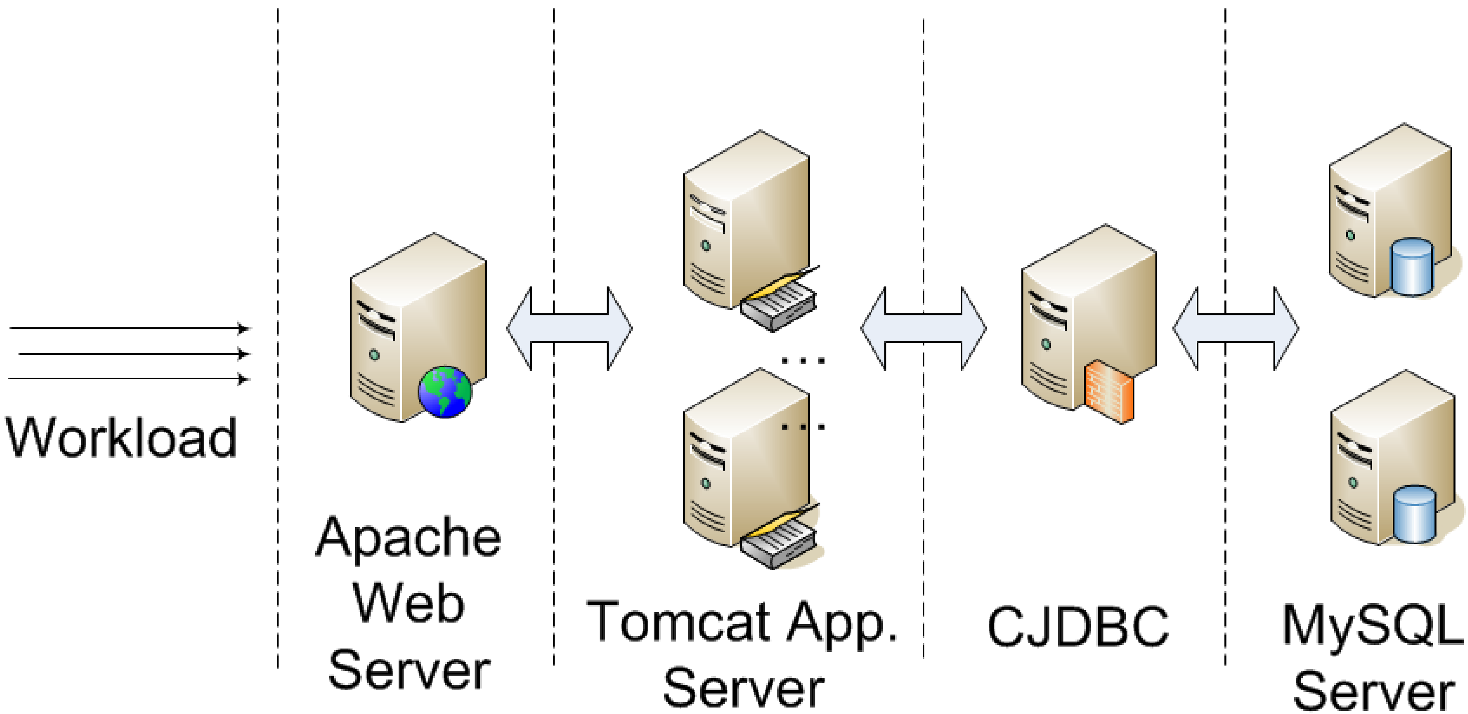
Elba: Automated Measurements



N-tier Benchmarks

Example Experiment: RUBBoS benchmark based on Slashdot

- Sample configuration (1/2/1/2)



Elba Experience

- Experimental studies analyzing performance data
  - Production-scale experiments on “real data centers”
  - Collaboration with many industry partners
  - Funding amplification from NSF
- Between 2013 and 2014: 13 papers
  - IEEE CLOUD, SCC, ICDCS, IRI, Big Data Congress, BigData, ACM TRIOS
  - More than 40 papers (2005 – 2014)

Intel Science & Technology Center for Cloud Computing

Automating Experiments

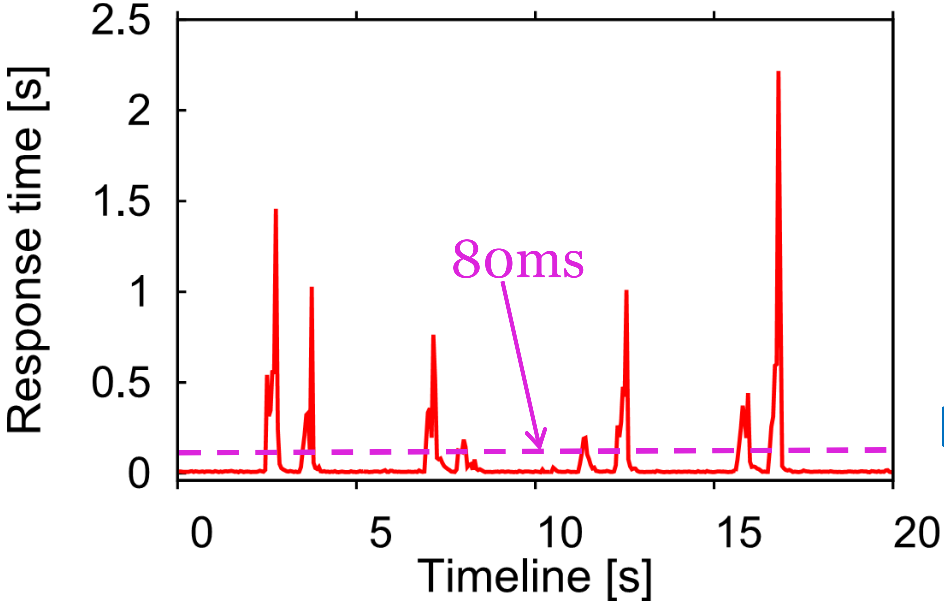
- Transform and generate scripts to automatically create, manage and analyze experiments from user-friendly specification files
- Develop open tools for automated experiments
  - Support a wide variety of ***evolving*** clouds, benchmarks and performance monitors
  - Support flexible customization for many configuration parameters
- High resolution monitoring at low cost

Intel Science & Technology Center for Cloud Computing

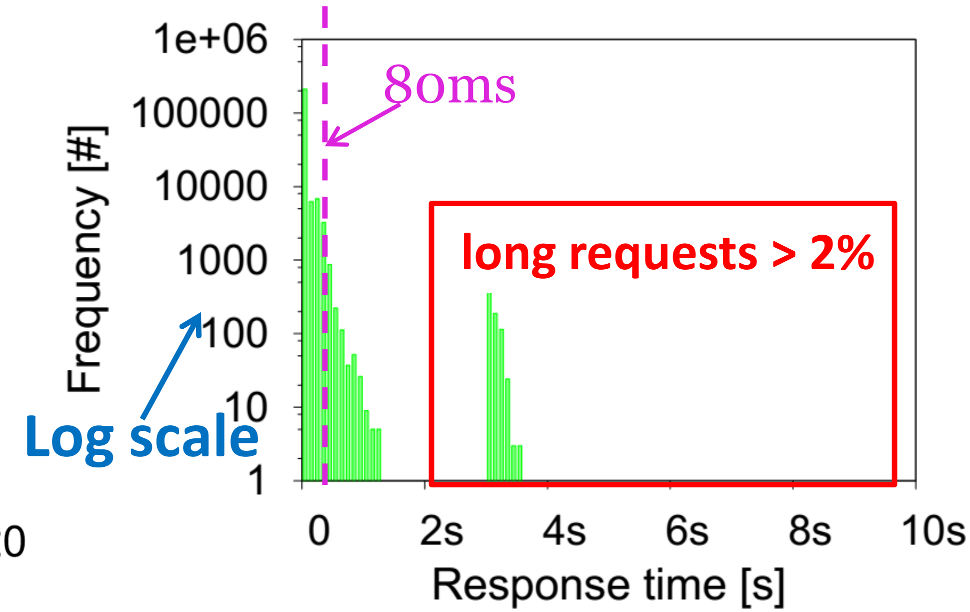
Example: Very Short Bottlenecks (TRIOS'13)

- High resolution monitoring at low cost:
  - See VSBs at tens of milliseconds
  - A few percent monitoring overhead

P-I-T Response time at 50ms resolution



Cumulative request response time distribution



Five Steps of Experimental Process

1. Input Experiment Metadata (XML)

User-provided configuration
Scripts and generator execution

2. Generate Experiment Scripts

Provision environment
Run benchmark
Setup, tear down infrastructure

3. Execute experiment on various clouds

Performance data extraction & load into database

4. Collect, extract, load experimental Data

5. Analyze Results (Excel and statistical tools)

Scale of Experiments

Figures for Fall 2014 and Spring 2015; taking into account diversity of work, including large-scale experimentation and infrastructure development activities

	Emulab	**PRObE**	**Local Cluster**
Experiments (mins)*	***91,728***	***4,641***	***2,925***
Nodes Used (#)	***6,048***	***1,092***	***4,516***
Experiments (#)	***14,112***	***714***	***450***

*Experimental workloads range from 3 - 7 mins, each lasting about 20 - 30 min

Step 2: Script Transformation Example

Code Template

```
source set_elba_env.sh
mkdir -p <xsl:value-of
select="//params[@name='RUBBOS_
TOP']/@value"/>
```

1 template line

XML Input

```
<xtbl name="Rubbos" version="0.1">
<params name="OUTPUT_HOME"
value="/opt/rubbos/output"/>
<params name="RUBBOS_TOP"
value="/mnt/rubbos"/>
</xtbl>
```

Intermediate Representations

```
<file>
<name id="Tomcat_deploy.sh" loc="/home/scripts"/>
source set_elba_env.sh
mkdir -p /mnt/rubbos
</file>
```

Occurs in 5 templates

Experiment-specific Scripts

```
#/home/scripts/TOMCAT_DEPLOY.sh
source set_elba_env.sh
mkdir -p /mnt/rubbos
```

And 3 script files

Intel Science & Technology Center for Cloud Computing

Script Size of Experiment Runs

The following figures correspond to deploying a 16-node, (4 clients; 2W\4A\1M\4D), RUBBoS application benchmark in the Emulab cluster. Generated lines are an intermediate representation that enable application, DBMS, OS and user-specific customizations to be applied.

	Templates (XSLT Lines)	**Intermediate** (XML/XLST Lines)	**Final Scripts** (Shell Script Lines)
Core	**900**	**400**	**1500**
Deployment	**3300**	**2000**	**2200**
Benchmark	**1400**	**500**	**500**

Step 4: Extract/Load Monitor Results

- Challenge: many performance monitors, many configuration parameters, many output formats
- Automated Approach:
  - use script transformation techniques to annotate monitor output
  - generalize parser to consume schema (from the annotations) and parse the encapsulated data accordingly

Intel Science & Technology Center for Cloud Computing

Example 1: dstat

- **Some monitors can output simple, CSV-formatted data files**

```
"Dstat 0.6.9 CSV output"
"Author:","Dag Wieers <dag@wieers.com>",,,,"URL:","http://dag.wieers.com/home-made/dstat/"
"Host:","169",,,,"User:","root"
"Cmdline:","dstat -c -d -i -m -n -r -y --vm --no --output /tmp/169.254.100.3.csv 1",,,,"Date:","25 Feb 2012 19:14:49 EST"

"total cpu usage",,,,,,"dsk/total",,"interrupts",,,"memory usage",,,,"net/total",,"io/total",,"system",,"virtual memory",,,
"usr","sys","idl","wai","hiq","siq","read","writ","15","17","18","used","buff","cach","free","recv","send","read","writ","int"
0.731,0.794,97.731,0.536,0.066,0.144,370910.050,14976.328,1.031,18.487,1.806,219025408.0,23076864.0,249458688.0,3556507648.0,
0.0,0.0,100.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,4.0,219045888.0,23076864.0,249470976.0,3556474880.0,468.0,386.0,0.0,0.0,28.0,16.0,0
0.990,0.0,99.010,0.0,0.0,0.0,0.0,0.0,2.0,0.0,2.0,219045888.0,23076864.0,249470976.0,3556474880.0,60.0,0.0,0.0,0.0,26.0,16.0,0
10.309,59.794,23.711,5.155,1.031,0.0,2867200.0,139264.0,0.0,23.0,37.0,655368192.0,23093248.0,250908672.0,3118698496.0,3651.0,
5.0,44.0,50.0,0.0,1.0,0.0,245760.0,0.0,2.0,1.0,13.0,219578368.0,23093248.0,251027456.0,3554369536.0,618.0,570.0,2.0,0.0,532.0
0.0,0.0,100.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,219578368.0,23093248.0,251027456.0,3554369536.0,60.0,0.0,0.0,0.0,22.0,20.0,0.0,
0.0,0.0,100.0,0.0,0.0,0.0,0.0,0.0,2.0,0.0,1.0,219578368.0,23093248.0,251027456.0,3554369536.0,0.0,0.0,0.0,0.0,23.0,14.0,0.0,0
0.0,0.0,100.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,2.0,219578368.0,23093248.0,251027456.0,3554369536.0,120.0,0.0,0.0,0.0,19.0,14.0,0.0
0.990,0.0,98.020,0.0,0.0,0.990,0.0,90112.0,0.0,5.0,4.0,219578368.0,23101440.0,251019264.0,3554369536.0,120.0,42.0,0.0,10.0,34
0.0,0.0,100.0,0.0,0.0,0.0,0.0,0.0,2.0,0.0,0.0,219570176.0,23101440.0,251027456.0,3554369536.0,0.0,0.0,0.0,0.0,18.0,14.0,0.0,0
7.0,14.0,79.0,0.0,0.0,0.0,0.0,81920.0,0.0,4.0,55.0,220344320.0,23109632.0,251031552.0,3553583104.0,5386.0,6105.0,0.0,8.0,294.
2.020,12.121,84.848,0.0,1.010,0.0,8192.0,196608.0,2.0,11.0,26.0,219537408.0,23130112.0,251039744.0,3554361344.0,2536.0,1989.0
2.0,5.0,91.0,0.0,0.0,2.0,0.0,0.0,0.0,0.0,433.0,228413440.0,23130112.0,251039744.0,3545485312.0,84721.0,84632.0,0.0,0.0,666.0,
0.0,0.0,99.0,0.0,1.0,0.0,0.0,0.0,2.0,0.0,31.0,228540416.0,23130112.0,251039744.0,3545358336.0,2680.0,8347.0,0.0,0.0,80.0,66.0
0.990,0.0,99.010,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,228540416.0,23130112.0,251039744.0,3545358336.0,0.0,0.0,0.0,0.0,18.0,11.0,0.
0.0,1.0,99.0,0.0,0.0,0.0,0.0,0.0,2.0,0.0,1.0,228540416.0,23130112.0,251039744.0,3545358336.0,60.0,0.0,0.0,0.0,23.0,16.0,0.0,0
```

Example 2: sar

- Other monitors can produce highly variable and difficult-to-parse output (syntax & semantics)

```
Linux 2.6.32-358.18.1.el6.x86_64 (elba2)    09/18/2013  _x86_64_    (4 CPU)

08:11:18 AM     CPU     %user     %nice   %system   %iowait    %steal     %idle
08:11:19 AM     all      1.01      0.00      0.25      0.50      0.00     98.24
08:11:19 AM       0      4.08      0.00      1.02      2.04      0.00     92.86
08:11:19 AM       1      0.00      0.00      0.00      0.00      0.00    100.00
08:11:19 AM       2      0.00      0.00      0.00      0.00      0.00    100.00
08:11:19 AM       3      0.00      0.00      0.99      0.00      0.00     99.01

08:11:18 AM    proc/s   cswch/s
08:11:19 AM      2.04   2068.37

08:11:18 AM  pswpin/s pswpout/s
08:11:19 AM      0.00      0.00

08:11:18 AM  pgpgin/s pgpgout/s   fault/s  majflt/s  pgfree/s pgscank/s pgscand/s pgsteal/s    %vmeff
08:11:19 AM      0.00     28.57    646.94      0.00    541.84      0.00      0.00      0.00      0.00

08:11:18 AM       tps      rtps      wtps   bread/s   bwrtn/s
08:11:19 AM     12.24      0.00     12.24      0.00    114.29

08:11:18 AM   frmpg/s   bufpg/s   campg/s
08:11:19 AM    -61.22      2.04      3.06

08:11:18 AM kbmemfree kbmemused  %memused kbbuffers  kbcached  kbcommit   %commit
08:11:19 AM   7179524    711684      9.02     36444    456152    254532      1.60

08:11:18 AM       DEV       tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz     await     svctm     %util
08:11:19 AM    dev8-0      5.10      0.00     57.14     11.20      0.03      5.40      5.20      2.65
08:11:19 AM  dev253-0      7.14      0.00     57.14      8.00      0.03      4.14      3.71      2.65
08:11:19 AM  dev253-1      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00
08:11:19 AM  dev253-2      0.00      0.00      0.00      0.00      0.00      0.00      0.00      0.00

08:11:18 AM     IFACE   rxpck/s   txpck/s    rxkB/s    txkB/s   rxcmp/s   txcmp/s  rxmcst/s
08:11:19 AM        lo      3.06      3.06      0.16      0.16      0.00      0.00      0.00
08:11:19 AM      eth0   1271.43   1173.47    679.75    854.51      0.00      0.00      0.00

08:11:18 AM     IFACE   rxerr/s   txerr/s    coll/s  rxdrop/s  txdrop/s  txcarr/s  rxfram/s  rxfifo/s  txfifo/s
```

Intel Science & Technology Center for Cloud Computing

Transforming Output of sar

XML Tree Parser

```
tree = XMLTree.parse(input_file)
root = tree.getroot()
node_list = list(root.iter())
for i in node_list:
        #controls when to start the next record
        if i.tag == record_ind:
            if ctr > 0:
                diff = value_str[len(last_str):len(new_str)-1]
                row_list.append(diff)
                value_str = value_str + '\n'
                last_str = value_str
            ctr = ctr + 1
        if len(i.attrib) ==0:
            #check for invisibile characters like tabs and line feeds
            test_list = [ord(s) for s in i.text if ord(s) == 9 or ord(s) == 10]
            if len(test_list)==0:
                    key_str = key_str + i.tag + ","
                    value_str = value_str + i.text + ","
        for k,v in i.attrib.iteritems():
            key_str = key_str + k + ","
            value_str = value_str + v + ","
        new_str = value_str
#output row_list to a file for database import
```

Parsing the following version of SAR output is reduced to parsing a XML tree

SAR Annotated Output


```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE sysstat PUBLIC "DTD v2.13 sysstat //EN"
"http://pagesperso-orange.fr/sebastien.godard/sysstat.dtd">
<sysstat>
    <sysdata-version>2.13</sysdata-version>
    <host nodename="node-5.111base.elba.marmot.pdl.cmu.local">
        <sysname>Linux</sysname>
        <release>2.6.43.8-1.fc15.x86_64</release>
        <machine>x86_64</machine>
        <number-of-cpus>2</number-of-cpus>
        <file-date>2015-07-18</file-date>
        <statistics>
            <timestamp date="2015-07-19" time="01:16:04" utc="1" interval="1">
                <cpu-load>
                    <cpu number="all" user="6.63" nice="0.00" system="3.06" iowait="0.00" steal="0.00" idle="90.31"/>
                    <cpu number="0" user="13.40" nice="0.00" system="5.15" iowait="0.00" steal="0.00" idle="81.44"/>
                    <cpu number="1" user="0.00" nice="0.00" system="1.01" iowait="0.00" steal="0.00" idle="98.99"/>
                </cpu-load>
                <process-and-context-switch per="second" proc="2.00" cswch="1588.00"/>
                <swap-pages per="second" pswpin="0.00" pswpout="0.00"/>
```

Georgia Tech Highlights for ISTC-CC

- Research on big data graph algorithm optimization
  - K. Schwan, L. Liu
- Research on program optimization for heterogeneous processors and memories
  - S. Yalamanchili, A. Gavrilovska
- Research on automating experiments on large scale benchmarks
  - C. Pu
- Many publications, some tool releases, more planned