



ORACLE®




ORACLE®

DB Time Oracle Performance Tuning: Theory and Practice

Graham Wood, Uri Shaft, John Beresiewicz
Oracle Corporation

April 2008



The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.

The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.



Agenda

- Oracle Performance Tuning Methods: A brief history
- Database Time and Average Active Sessions
- DB Time and System Performance
- Active Session History
- The DB Time Method

Oracle Performance Tuning Methods:

A brief history





Oracle Tuning Methods

- Prehistory (v5)
 - Debug code
- Dark Ages (v6)
 - Counters/Ratios
 - BSTAT/ESTAT
 - SQL*Trace
- Renaissance (v7)
 - Introduction of Wait Event instrumentation
 - Move from counters to timers
 - STATSPACK



The Modern Age of time-based tuning

- YAPP (8i) - Instance tuning using instance statistics
 - Non intrusive, always on
 - Broadly scoped
- Method R (9i) - Session tuning using 10046 SQL traces
 - Tightly scoped
 - Must be highly selective
- DB Time Tuning (10g) – Comprehensive tuning using fundamental notion of time in database
 - Multiple scoping levels
 - Always on, non-intrusive
 - Built into infrastructure: instrumentation, ASH, AWR, ADDM, EM

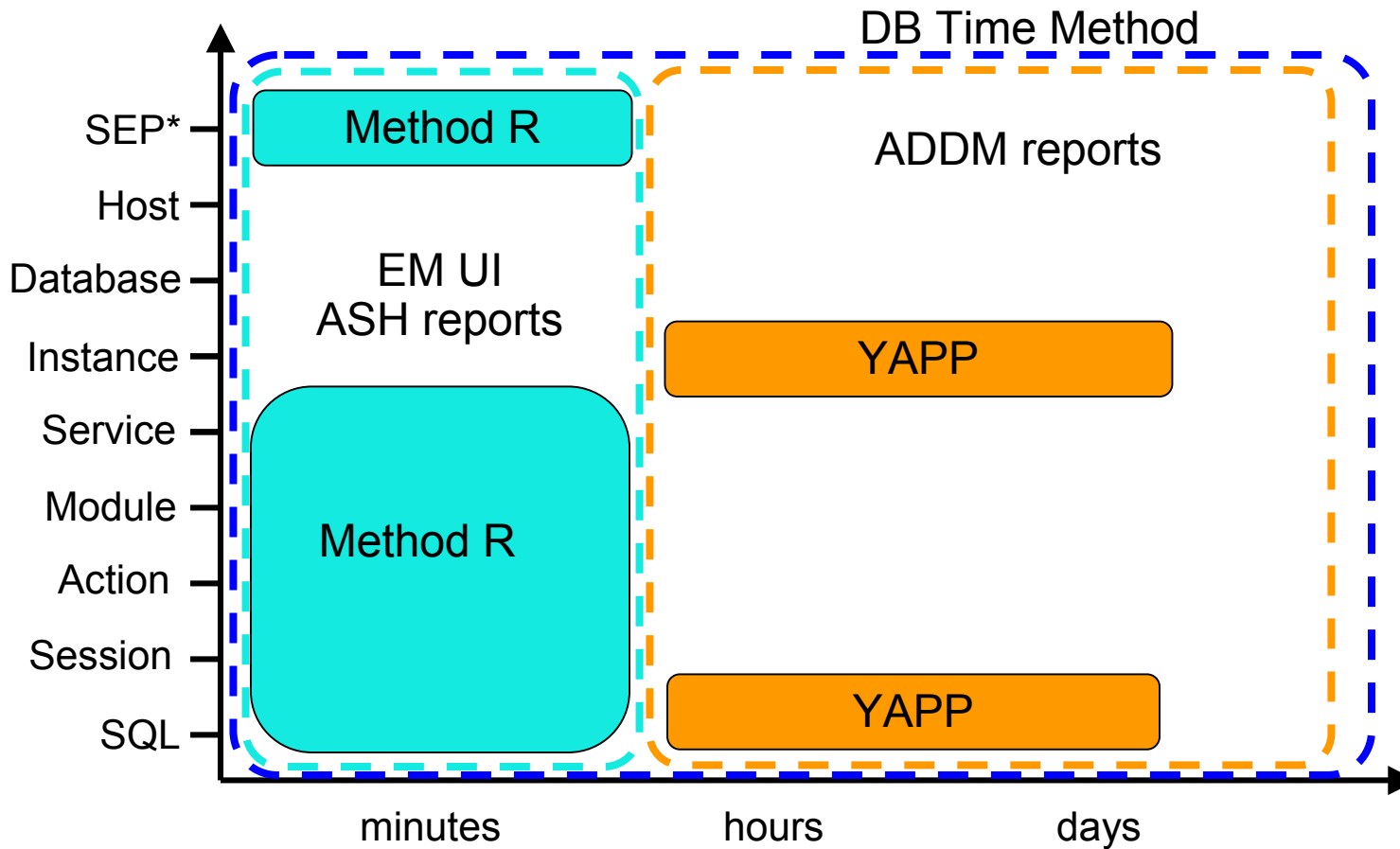


Why Do We Care About Time?

- Human time is critical to the enterprise
- Systems performance affects business goals
 - Human time + technology resource time
- “Time is money”
- Performance improvement means doing things faster

Performance is always and only about time

Scope of time-based methods



*SEP = Somebody Else's Problem



The DB Time method: short course

or

just ask ADDM

Database Time and Average Active Sessions





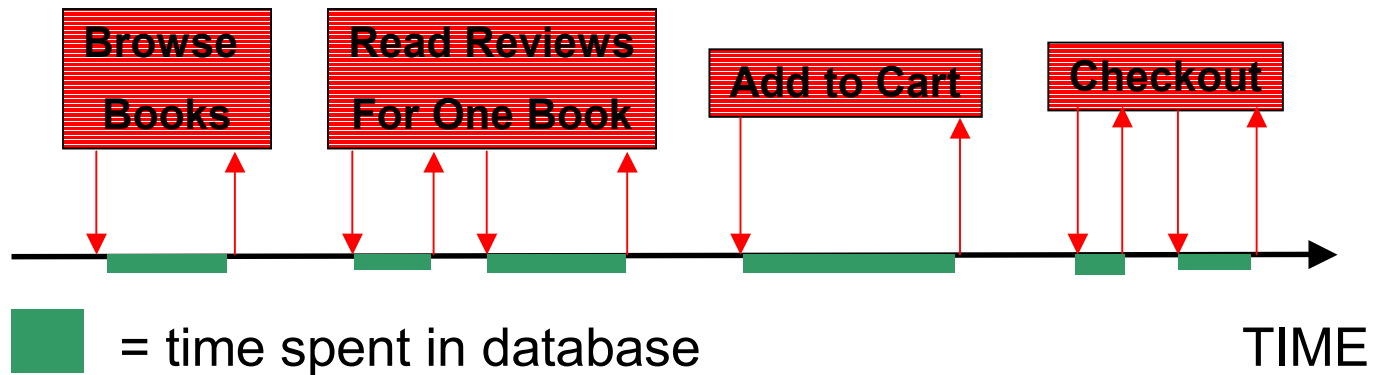
Database time (DB Time)

- Total time in database calls by **foreground sessions**
- Includes **CPU** time, **IO** time and **non-idle wait** time
- DB Time <> response time
- New lingua franca for Oracle performance analysis

Database time is total time spent by user processes either actively working or actively waiting in a database call.

A Single Session

Single session with Database Black Box server



Fundamental concepts

Active Session =

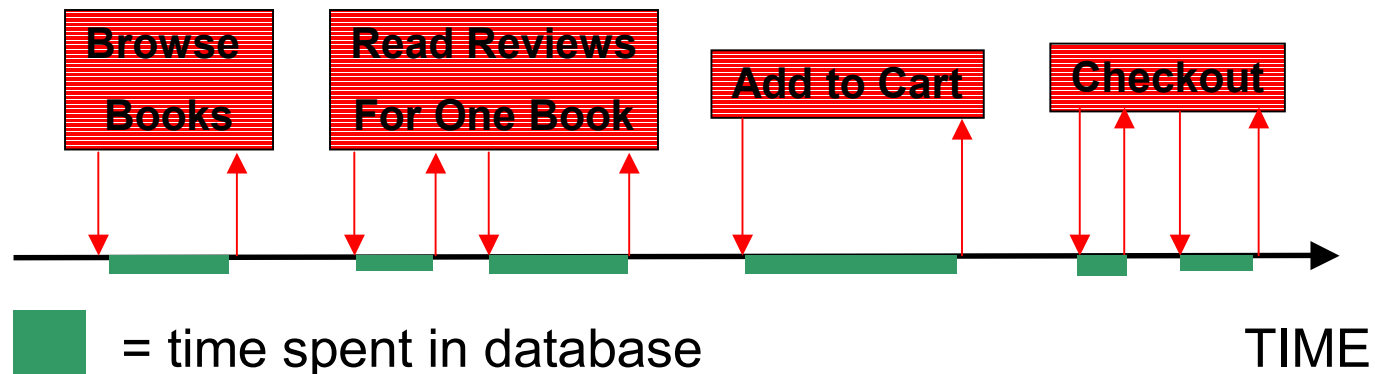
Session currently spending time in a database call

Database Time (DB Time) =

Total time session spent in all database calls

Average Activity of the Session (% Activity) =

The ratio of time active to total wall clock time



ORACLE Enterprise Manager 10g

Grid Control

Home **Targets** Deployments Alerts Compliance Jobs Reports

Hosts | Databases | Application Servers | Web Applications | Groups | **All Targets** | Collaboration Suites

Cluster: dbs232_crs > Cluster Database: BUGAP.US.ORACLE.COM > Top Sessions > Database Instance: BUG1AP_DB8232 > Top Activity > Logged in As JSARICOS

Session Details: 1869 (AFOTHERG)

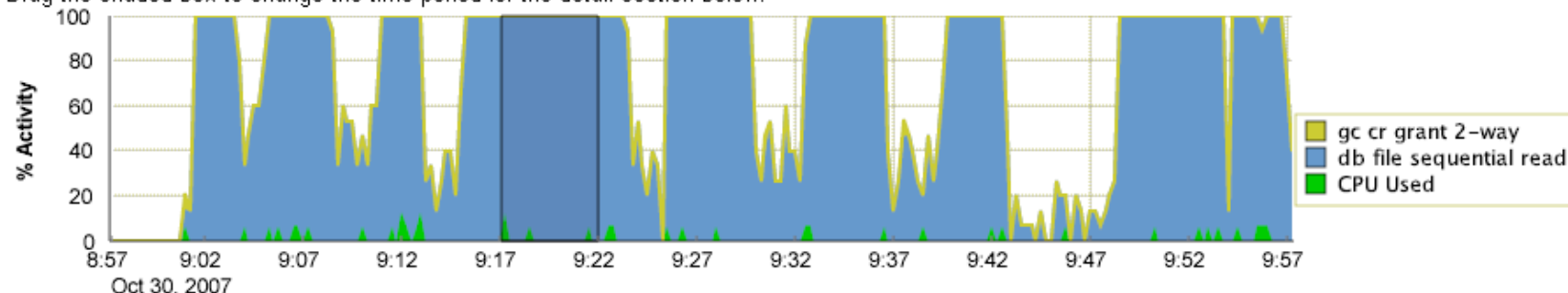
Collected From Target Oct 30, 2007 9:49:37 AM CDT

View Data Real Time: 15 Second Refresh Refresh

Kill Session Enable SQL Trace

General **Activity** Statistics Open Cursors Blocking Tree Wait Event History

Drag the shaded box to change the time period for the detail section below.



Detail for Selected 5 Minute Interval

Start Time Oct 30, 2007 9:17:05 AM View Show Aggregated Data Run ASH Report

Activity (%)	SQL ID	SQL Command	Plan Hash Value	Module	Action	Client ID
100.00	gkmd7xwuz1na0	SELECT	64730335	oraclealan@ap103fam (TNS V1-V3)	AFOTHERG	

General **Activity** Statistics Open Cursors Blocking Tree Wait Event History

Kill Session Enable SQL Trace

Home | **Targets** | Deployments | Alerts | Compliance | Jobs | Reports | Setup | Preferences | Help | Logout

Copyright © 1996, 2007, Oracle. All rights reserved.

Oracle, JD Edwards, PeopleSoft, and Retek are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

[About Oracle Enterprise Manager](#)

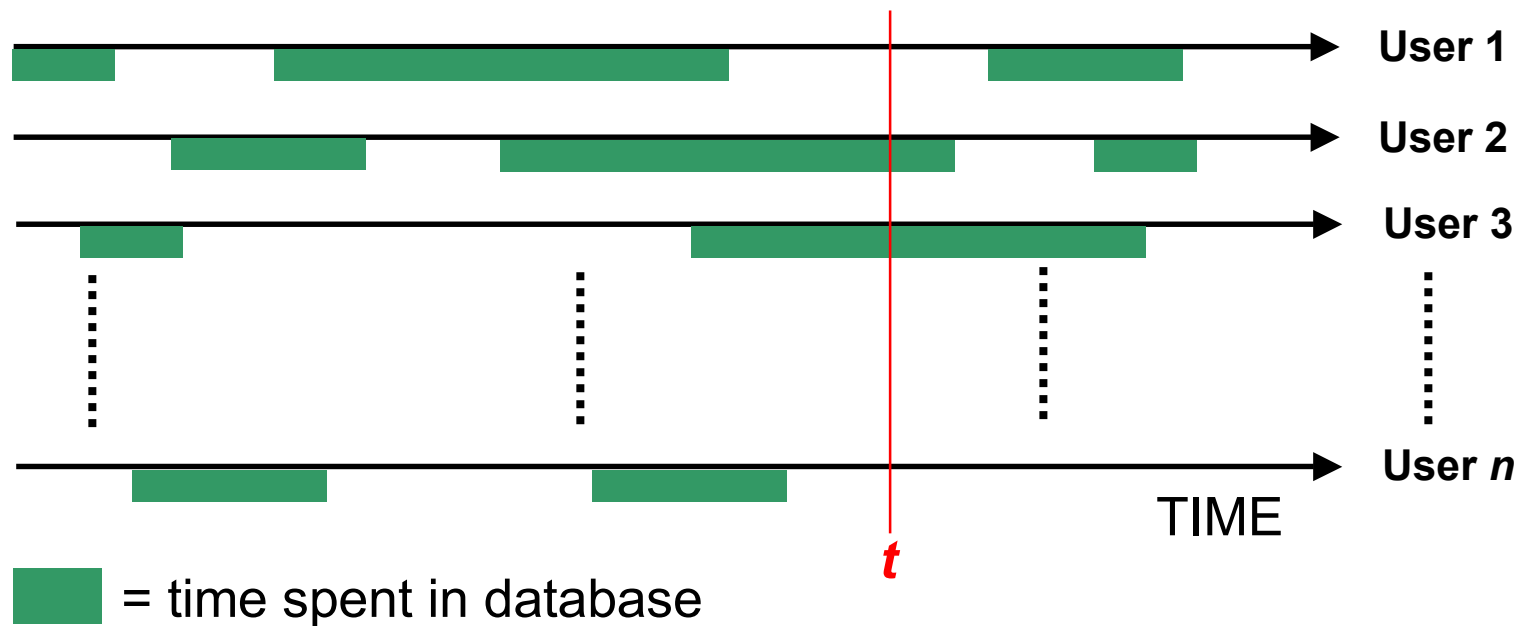


Multiple Sessions

DB Time = Sum of DB Time Over All Sessions

Avg. Active Sessions = Sum of Avg. Activity Over All Sessions

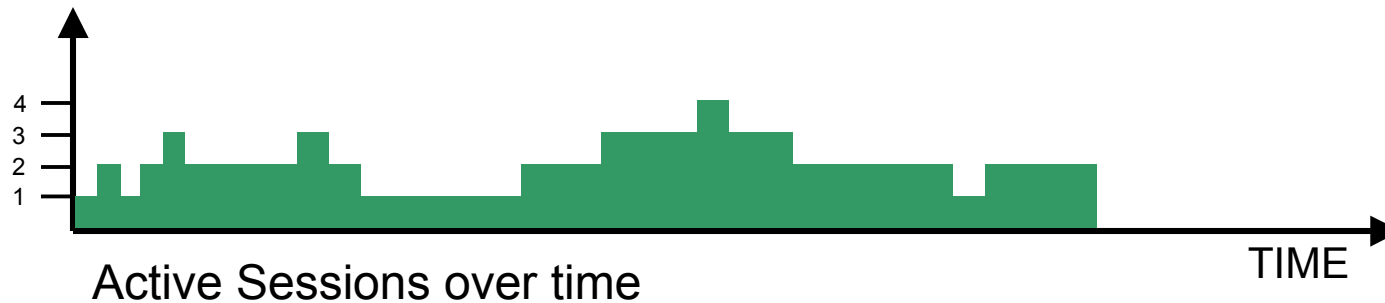
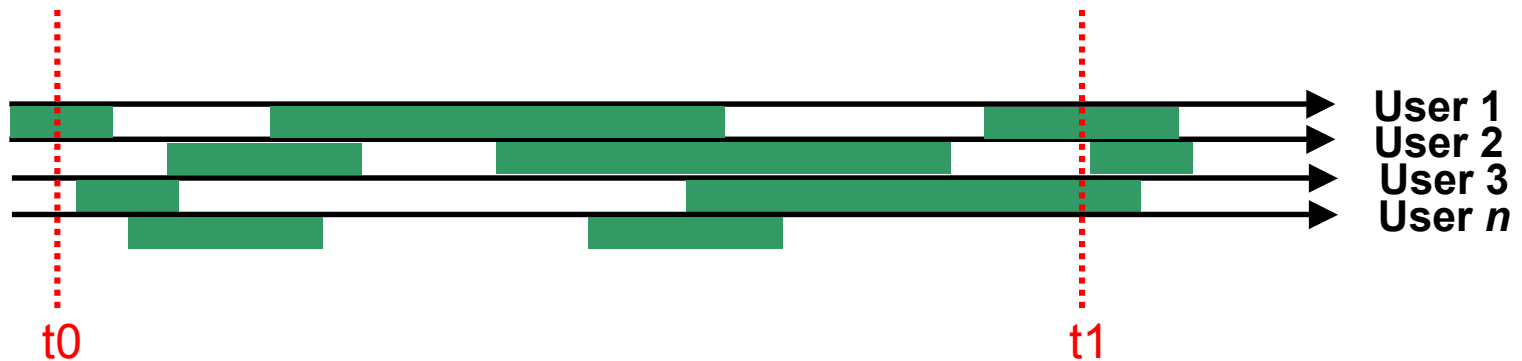
At time t we have 2 active sessions



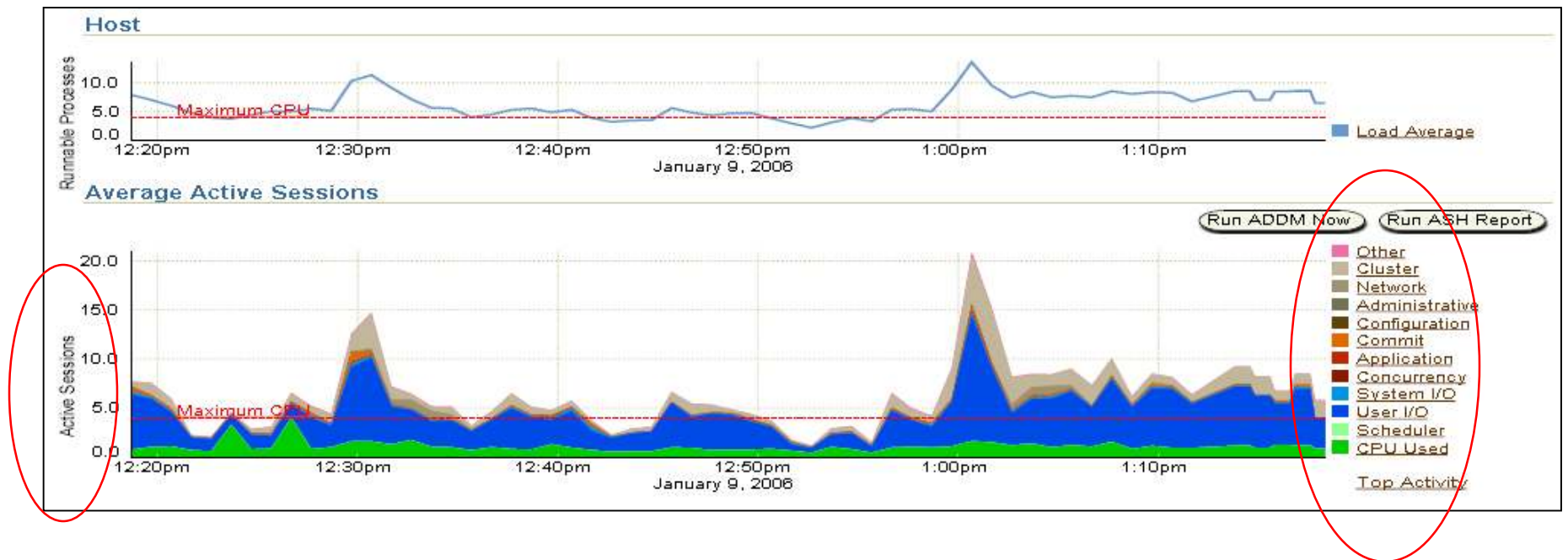


The Basic Concept

$$\text{Avg. Active Sessions} = \frac{\text{Total Database Time}}{\text{Wall Clock (Elapsed) Time}}$$



EM Performance page



- Active Sessions by wait class over time
- Colored area = amount of DB time
- “Click on the big stuff...”



DB Time instrumentation

- New session-level timers at call boundaries
 - Connection management time captured to begin session
 - Some call stack timing captured (e.g. hard parse breakdown)
 - Connection tear-down time captured (?)
- DB time aggregated into system level counters
 - V\$SYS_TIME_MODEL
- More complete wait time instrumentation
 - Call cleanup, e.g. cursor close
 - Many new events
 - Wait event classification

DB Time and System Performance





System load and DB time

- More users
 - => More calls
 - => DB time increases
- Larger transactions
 - => Longer calls
 - => DB time increases

DB time increases as system load increases.



System performance and DB time

- IO performance degrades
 - => IO time increases
 - => DB time increases
- Application performance degrades
 - => Wait time increases
 - => DB time increases

DB time increases when performance degrades.



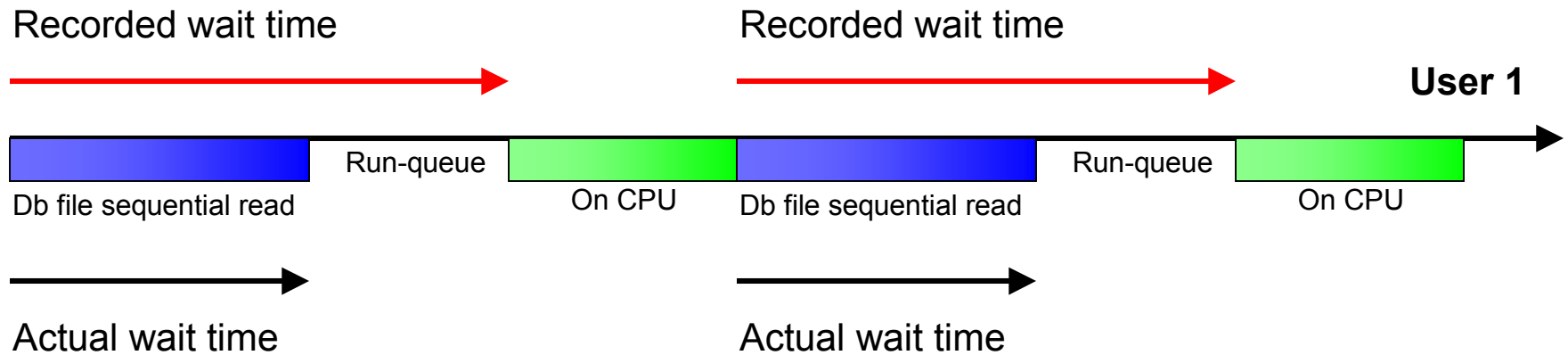
Host performance and DB time

- Host is CPU-bound
 - => foregrounds accumulate active run-queue time
 - => wait event times are artificially inflated
 - => DB time increases

Tune for CPU before waits when CPU constrained



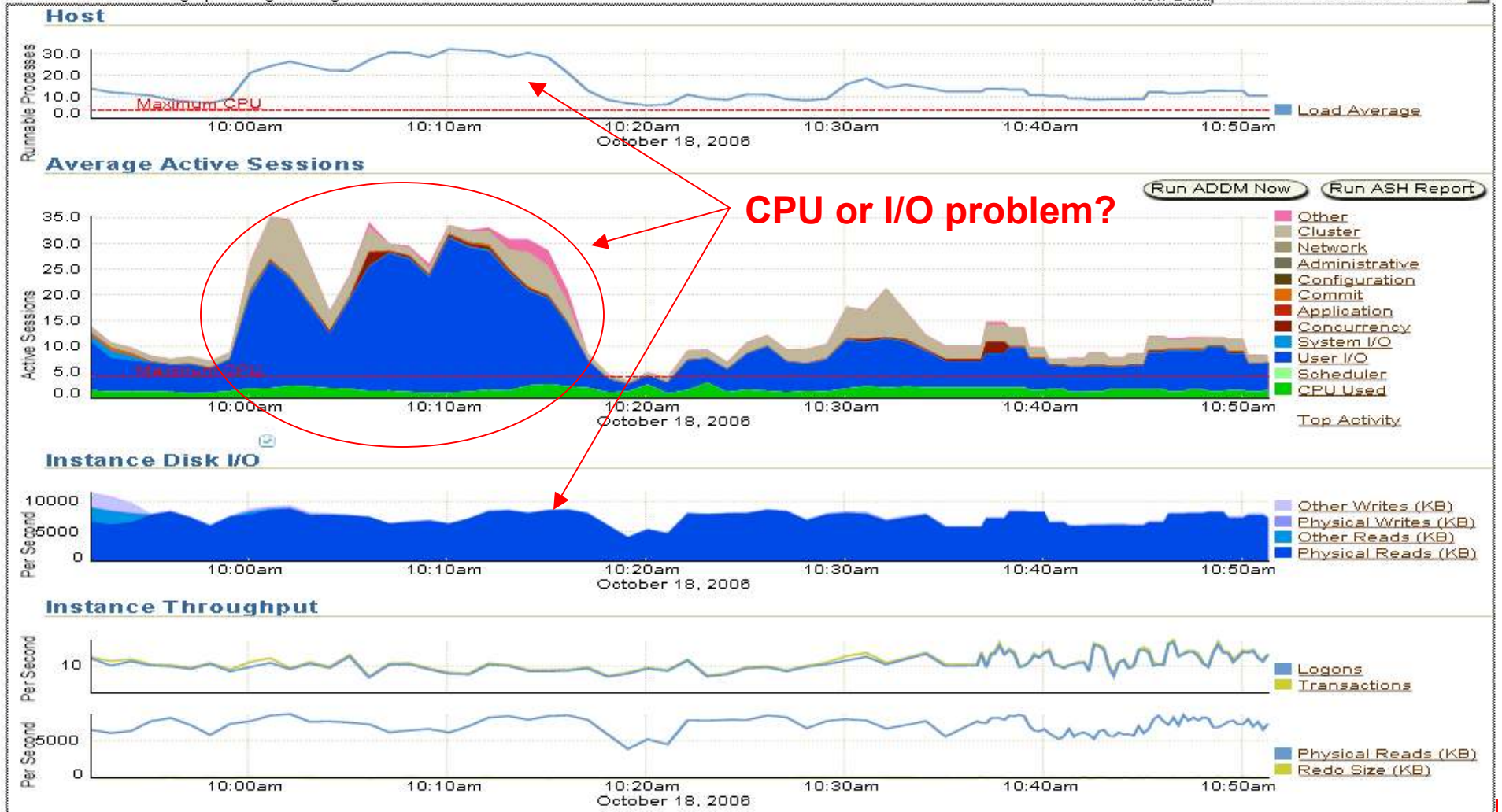
CPU run-queue and DB time



DB time is inflated when host is CPU-bound

Click on an area of a graph or legend to get more detail.

View Data Real Time: 15 Second Refresh



CPU or I/O problem?

Instance Throughput Rate Per Second Per Transaction

Additional Monitoring Links

Top Sessions and Top SQL data from ASH can be found on the Top Activity page.



Where to find DB time?

- **V\$SYS_TIME_MODEL**
 - STAT_NAME = 'DB time'
 - Accumulated value over entire instance
- **V\$WAITCLASSMETRIC_HISTORY**
 - AVERAGE_WAITER_COUNT (Avg Active Sessions)
 - DBTIME_IN_WAIT (percent of DB Time)
- **V\$SYSMETRIC_HISTORY**
 - "Database Time Per Second", "CPU Usage Per Sec"
 - 10g units = centi-secs/sec (100xAvg. Active Sessions)
 - 11g new metric "Average Active Sessions"



Where to find DB time?

- **V\$SQL**
 - ELAPSED_TIME and CPU_TIME
 - Wait class times:
APPLICATION, CONCURRENCY, CLUSTER, USER_IO

- **V\$ACTIVE_SESSION_HISTORY**
 - Samples active sessions every second
 - DB Time = Sample counts



Where is DB time used?

- ADDM
- EM Performance page and drill downs
- ASH report
- AWR and AWR compare periods reports
- SYSMETRICS and Server-generated Alerts

Active Session History





Active sessions

- **Active Foreground** sessions in a database call
 - Either on **CPU**, waiting for **IO**, or **waiting** (not idle)
 - Backgrounds are also interesting

```
select *  
  from V$SESSION S  
 where S.status = 'ACTIVE'  
       and S.type   = 'USER'
```

***Active sessions are foreground sessions
contributing to DB time at any given moment***



Active Session History (ASH)

- Active sessions sampled and persisted in-memory
 - Sampling interval = 1 second
 - V\$ACTIVE_SESSION_HISTORY
 - Foreground and background sessions are sampled
- On-disk persistence (1 in 10 samples)
 - DBA_HIST_ACTIVE_SESS_HISTORY
- ASH is a many-dimensional FACT table
 - Dimensions are V\$SESSION columns
 - Fact is that DB time was accumulating over these dimensions
- ASH is a system-wide trace of what happened

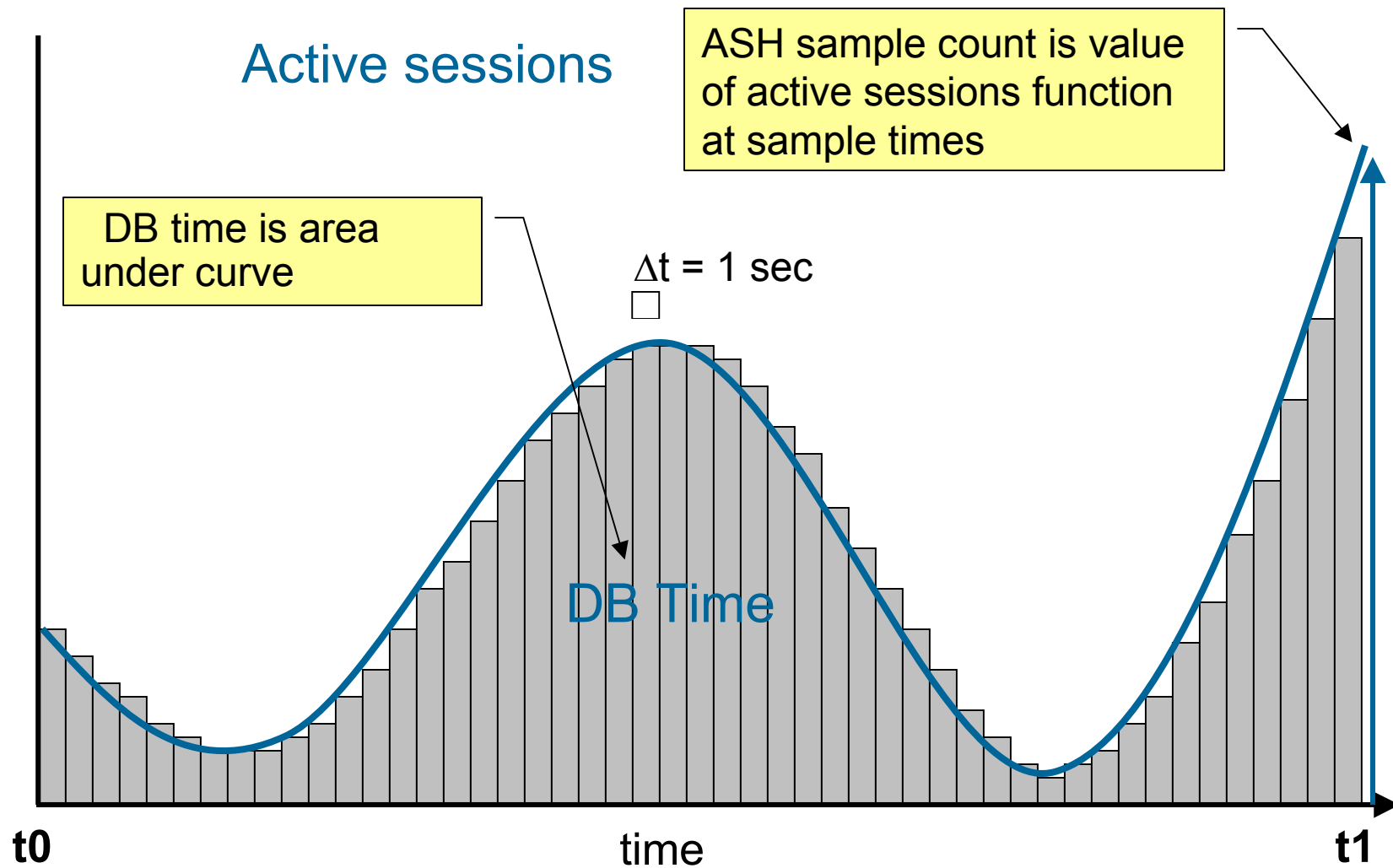


ASH and DB time

- **ASH sample counts = DB Time** in seconds
 - Assumes default 1 second sampling
 - Low sample sizes are less reliable
- Enables DB Time analysis over many dimensions
 - Sqlid, session id, instance, service, module, action

```
select COUNT(1) as dbtime
       ,sqlid
  from v$active_session_history
 where session_type='FOREGROUND'
 group by sqlid
 order by 1;
```

Estimating DB time with ASH



Integral approximation using ASH

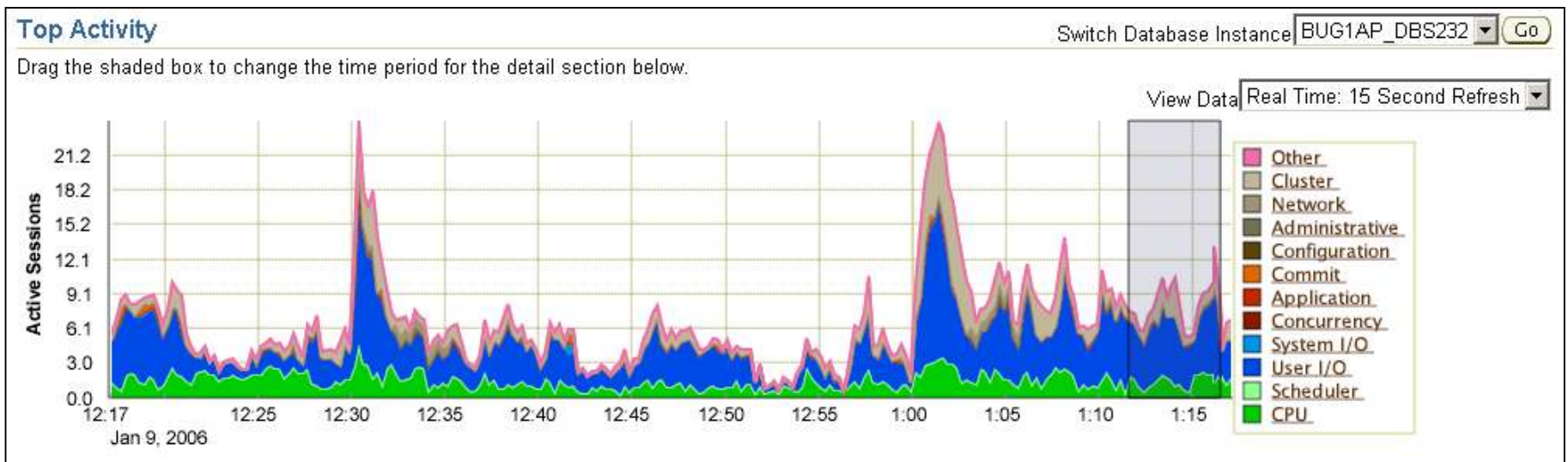
$$\text{DB time} = \int_{t_0}^{t_1} \text{ActiveSessions}(t) dt$$

$$= \lim_{\Delta t \rightarrow 0} \left[\sum_{k=1}^n \text{ActiveSessions}(t_k) * \Delta t \right]$$

$$\approx \sum_{\text{sampletime} \geq t_0}^{\text{sampletime} \leq t_1} \text{ASHsamples} * 1$$

(where $\Delta t = 1$ second)

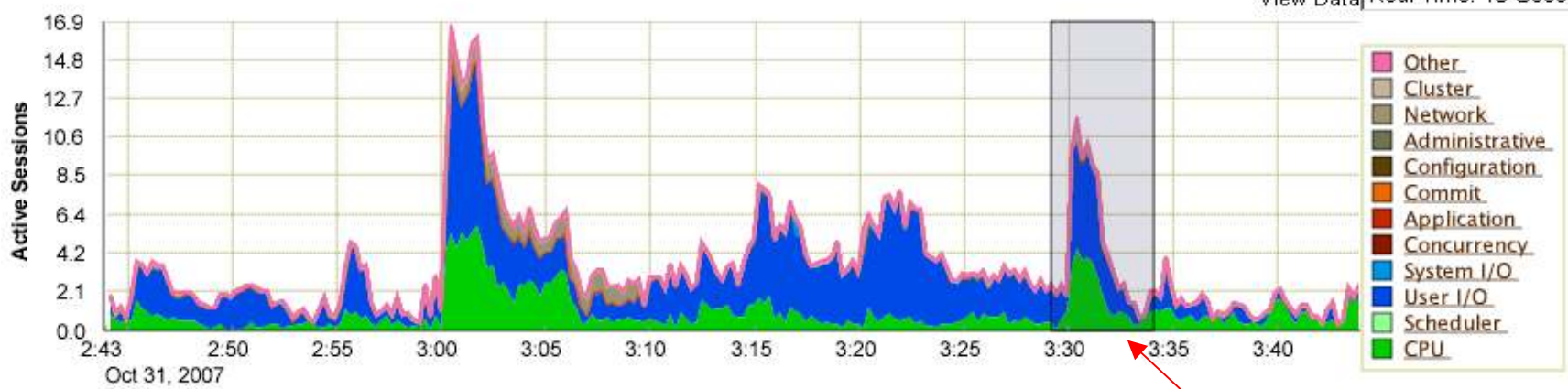
EM Top Activity page



- ASH-estimated DB time by wait class
- Aggregated over 15 second intervals

Drag the shaded box to change the time period for the detail section below.

View Data



Detail for Selected 5 Minute Interval

Start Time **Oct 31, 2007 3:29:12 PM CDT**

DB time

Top SQL

Select All | Select None

Select Activity (%)	SQL ID	SQL Type
<input type="checkbox"/> 12.59	8zn5trv71d4a	SELECT
<input type="checkbox"/> 10.75	6p15v8k8zwuh5	SELECT
<input type="checkbox"/> 10.66	076tygk66k8ha	SELECT
<input type="checkbox"/> 7.30	8f4k3v24y910b	SELECT
<input type="checkbox"/> 4.95	22183tatccs8d	SELECT
<input type="checkbox"/> 4.95	93sgg7vmg35xy	SELECT
<input type="checkbox"/> 4.20	5qvmmr821tbb	SELECT
<input type="checkbox"/> 3.86	9rauu0kprxwuf	SELECT

Top Sessions

View

Activity (%)	Session ID	User Name	Program
<input type="checkbox"/> 9.54	1906	AOLREP	perl@atgebs.us.oracle.com (TNS V1-V3)
<input type="checkbox"/> 9.32	1728	BGRAEF	oracledb92@iasbde.us.oracle.com (TNS V1-V3)
<input type="checkbox"/> 6.38	2251	ARUDAS	JDBC Thin Client
<input type="checkbox"/> 5.72	1682	BUGPATCH	oracle@staip12 (TNS V1-V3)
<input type="checkbox"/> 5.65	1570	MFGOPSTM	
<input type="checkbox"/> 5.36	2047	MFGOPSTM	? @ap615utl (TNS V1-V3)
<input type="checkbox"/> 4.62	1695	TOGEORGE	
<input type="checkbox"/> 3.96	1935	JSARICOS	OMS

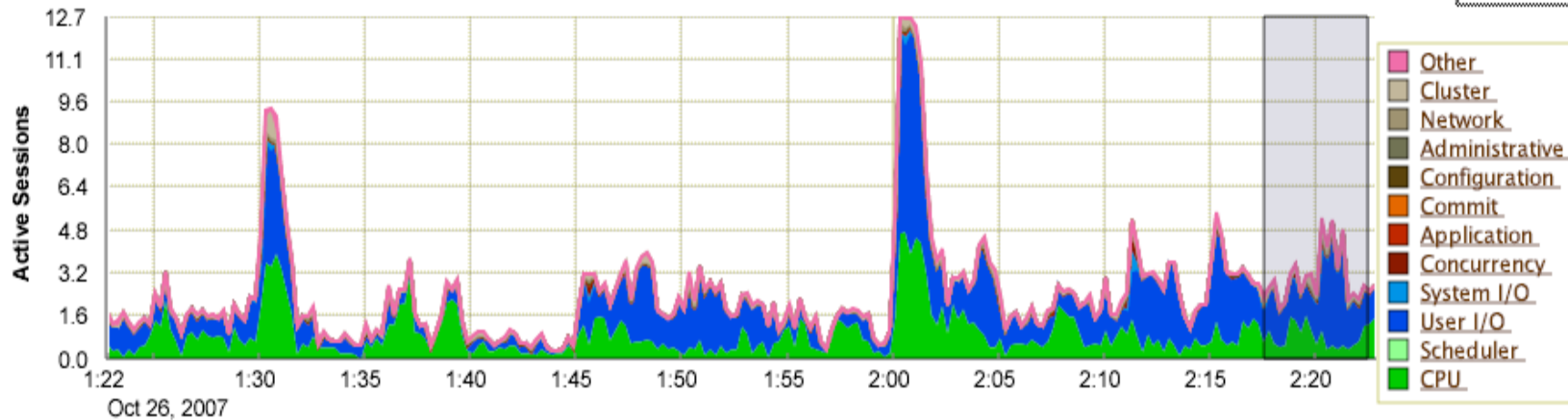
Sampled vs. cumulative DB time

Top Activity

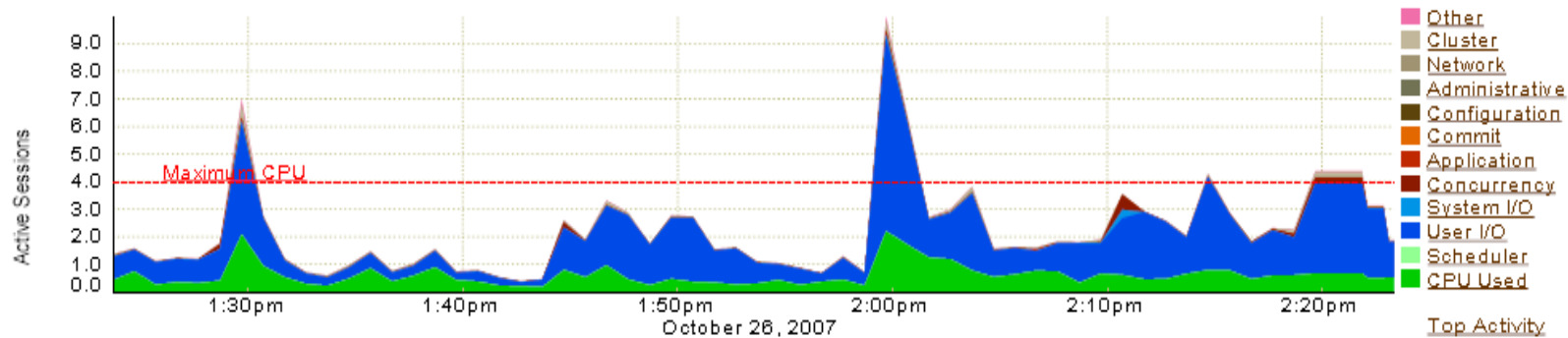
Switch Database Instance

Drag the shaded box to change the time period for the detail section below.

View Data



Average Active Sessions





The DB Time Method





The DB Time Method: context

Two use-cases, one method:

1. Systemic performance tuning

- Improve overall workload throughput or response time
- Best practice: use ADDM

2. Transient performance problem diagnosis

- Confirm and investigate reported performance issues
- Best practice: use EM



The DB Time Method: process

1. Identify performance issue
2. Scope the issue
3. Set goals
4. Investigate DB time distribution
 - Identify the largest potential for improvement
5. Modify system to tune for largest gain
6. Evaluate against goals
 - Repeat from step 4 if goals not met

Performance tuning by removing excess DB time



Identify performance issues

- User reported: complaints about performance
- System monitoring, SYSMETRIC thresholds:
 - Database Time Per Sec (10g)
 - **Average Active Sessions** (11g)
 - Response Time Per Txn (DB time per transaction)
 - SQL Service Response Time (DB time per user call)
- Extending system capacity
 - Don't Kill It With Iron, Resuscitate It With Tuning



Scope performance issues

- Organizational level scoping (business impact)
 - Enterprise
 - Department
 - Application
 - User
- Database level scoping (technology impact)
 - Host
 - Database/Instance
 - Service/Module/Action
 - Session/Client ID
 - SQL ID



Set tuning goals

- Goals should be business-oriented
 - Technology serves the business, not vice versa
- Goals should be quantifiable
 - Business-relevant, database measurable metrics are best
 - Perform X transactions per minute
 - “Run as fast as possible” is NOT a goal (not quantifiable)
- Scope the goals
 - How much of the pain can be relieved?
 - Temporary vs. permanent solutions
 - What are the constraints?



Investigate DB time distribution

- Identify uneven distributions of DB time (skew)
 - => Largest potential improvement within scope
- System scope:
 - Resource limits – is problem outside the DB?
- Application scope:
 - Service, module, action
 - Resource contention (e.g. latches)
 - SQLID, rowsource
- Session scope:
 - Long running SQL
 - Resource contention (e.g. enqueues)



Identify potential solutions

- Session contention issues
 - Kill session
 - Fix application
- SQL issues
 - SQL Tuning Advisor => Indexes, SQL profile
 - Re-write SQL
- Design issues
 - Access Advisor => Indexes, physical layout
- System issues
 - Initialization parameters
 - Add resources



DB time distribution: data sources

- System scope
 - V\$SYS_TIME_MODEL
 - V\$WAITCLASSMETRIC_HISTORY
 - V\$SQL
- Session scope
 - V\$SESS_TIME_MODEL
- All scopes
 - V\$ACTIVE_SESSION_HISTORY
 - DBA_HIST_ACTIVE_SESS_HISTORY



Modify system

- Address the largest DB time issues first
- Stay within scope
 - Don't tweak optimizer parameters before tuning SQL
- Proceed iteratively one fix at a time
 - Concurrent fixes should be orthogonal (if necessary)
- Measure results at each successive step
- Stop when goals are met



Method Summary

- DB time is the fundamental performance metric
- The method allows DB time analysis at many scopes
 - Proper scoping of problems and solutions is critical to success
- DB time based diagnosis removes value judgments
 - Scientific method, not sorcerer's magic
- Performance improvement means doing the same work in less DB Time



The DB Time Method: advantages

- Combines cumulative and sampled DB Time
 - 'Always on' data collection
 - Advance scoping not necessary
- Combines best of current methods
 - Less intrusive, more inclusive
- No requirement to reproduce problem
- Works for concurrency problems such as locking

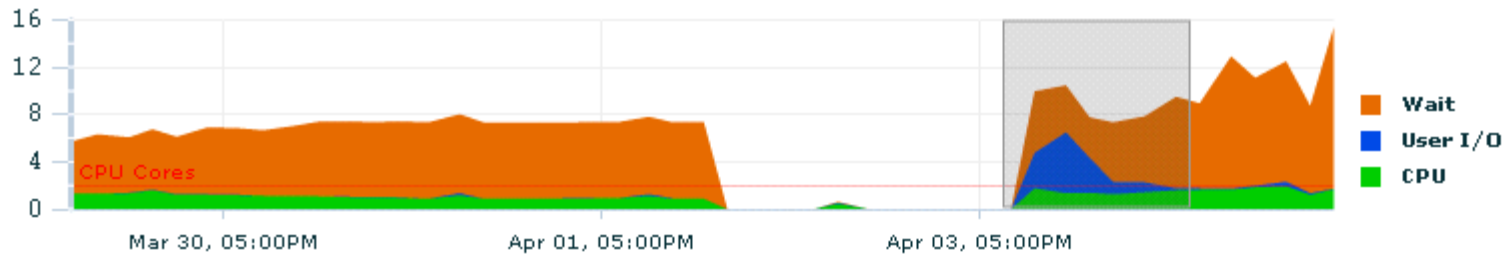


Best practice: use ADDM

- Embedded expert system using the DB time method
 - Identifies root causes behind the symptoms
- Variably scoped:
 - Host to instance to SQL and even database block
 - Scoped to database for RAC (new in 11g)
- Findings prioritized by impact on DB time
 - Finding history allows flexible time scoping
 - Directives can filter findings
- Recommendations by benefit (reduction) to DB time

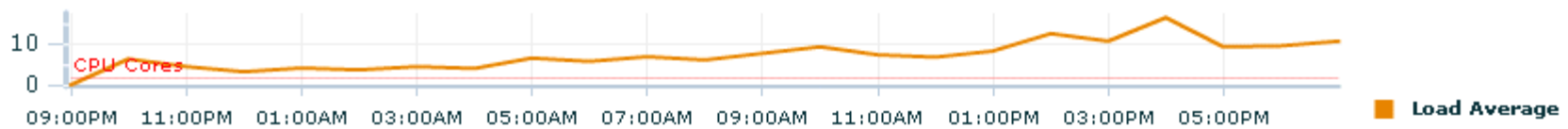
emptarget_emptarget1 (Database Instance)

Average Active Sessions - 7 Day View

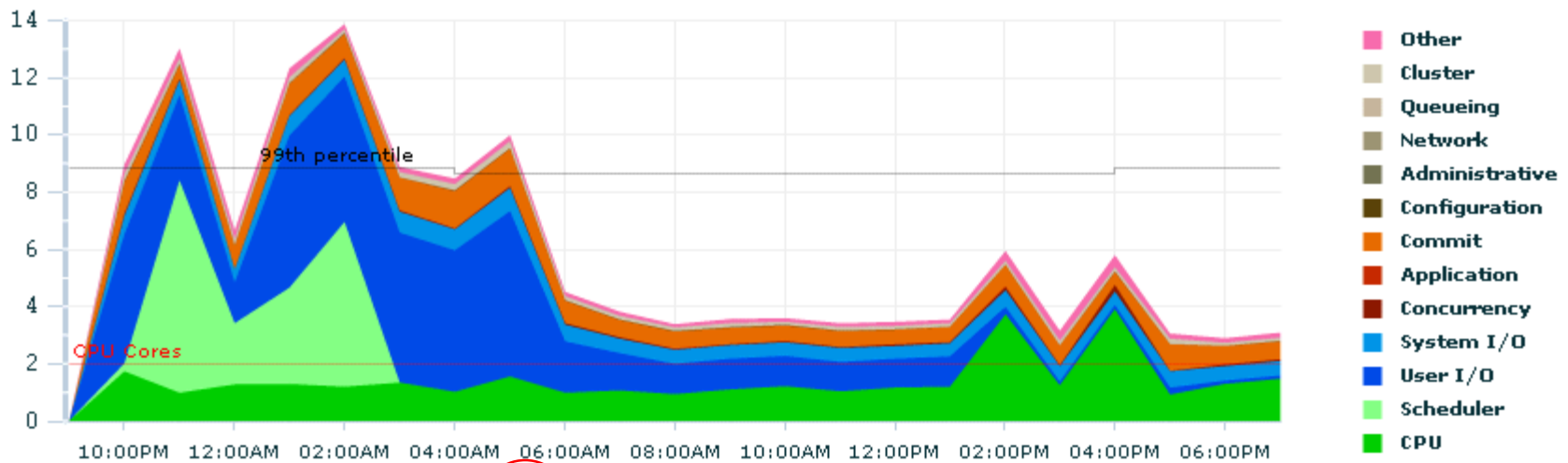


April 2008						
S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

Host: Runnable Processes



Average Active Sessions



Top Activity



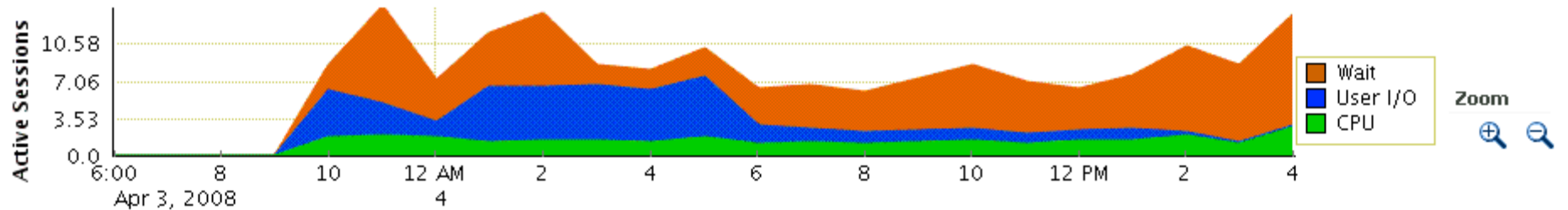
Automatic Database Diagnostic Monitor (ADDM)

Page Refreshed Apr 5, 2008 2:17:40 PM PDT [Refresh](#)

Database Activity

[Run ADDM](#) [Finding History](#)

The icon selected below the graph identifies the ADDM analysis period. Click on a different icon to select a different analysis period.



TIP For an explanation of the icons and symbols used in this page, see the [Icon Key](#)

ADDM Performance Analysis

Task Name ADDM:3132078998_1_1978

[Filters](#) [View Snapshots](#) [View Report](#)


Task Owner	SYS	Average Active Sessions	10.2	Period Start Time	Apr 4, 2008 4:00:31 AM PDT	Period Duration (minutes)	60	Instance	emptarget_emptarget1	Global ADDM Task Name	ADDM:3132078998_1_1978
------------	-----	-------------------------	------	-------------------	----------------------------	---------------------------	----	----------	----------------------	-----------------------	------------------------

Impact (%) ▼	Finding	Occurrences (latest 24 hrs)
47.9	Top SQL by DB Time	24 of 24
39.5	Top SQL By I/O	0 of 24
36.3	Top Segments by I/O	1 of 24
12.9	Commits and Rollbacks	23 of 24
7.6	I/O Throughput	1 of 24
2.5	Undersized PGA	0 of 24

Performance Finding Details: Top SQL by DB Time

Finding: SQL statements consuming significant database time were found. [Finding History](#)

Impact (Active Sessions): 4.03

Impact (%):  52.8

Period Start Time: Apr 4, 2008 12:00:04 PM PDT






Period Duration (minutes): 60.2

Filtered: No [Filters](#)

Recommendations

[Schedule SQL Tuning Advisor](#)

[Select All](#) | [Select None](#) | [Show All Details](#) | [Hide All Details](#)

Select	Details	Category	Benefit (%)
<input type="checkbox"/>	Hide	SQL Tuning	 15
Action	Investigate the SQL statement with SQL_ID "66n44vwsmynr" for possible performance improvements. SQL Text: <code>select /* serial_guys */ p_brand, p_type, p_size, ...</code> SQL ID: 66n44vwsmynr		View Tuning History
Rationale: SQL statement with SQL_ID "66n44vwsmynr" was executed 4 times and had an average elapsed time of 1031 seconds.			
<input checked="" type="checkbox"/>	Hide	SQL Tuning	 13.3
Action	Run SQL Tuning Advisor on the SQL statement with SQL_ID "4scj37xz190kp". SQL Text: <code>select /* big_guys */ /* NO_GBY_PUSHDOWN */ s_name, s_address ...</code> SQL ID: 4scj37xz190kp		View Tuning History Run Advisor Now Filters
<input type="checkbox"/>	Show	SQL Tuning	 10.2
<input type="checkbox"/>	Show	SQL Tuning	 8
<input type="checkbox"/>	Show	SQL Tuning	 6.6

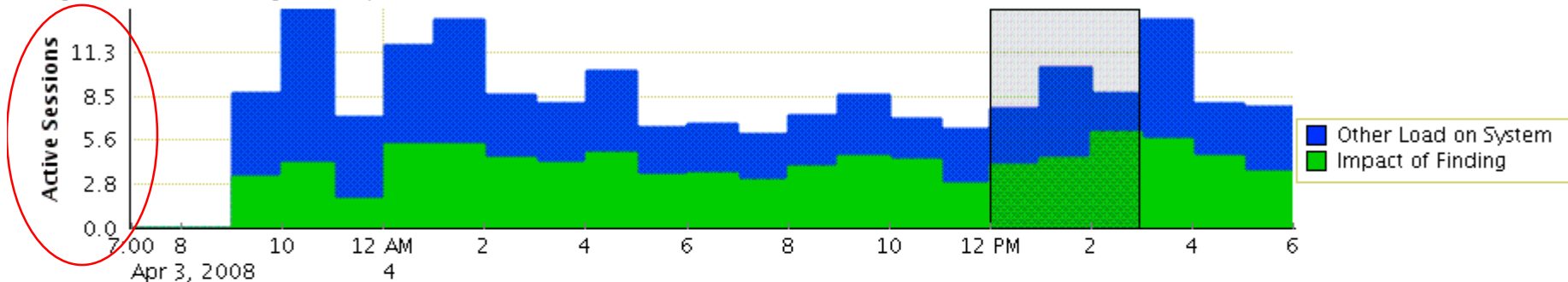
Findings Path

[Expand All](#) | [Collapse All](#)

Finding History: Top SQL by DB Time

View

Drag the shaded box to change the time period for the detail section below.



Detail for Selected 3 Hour Interval

Show All Details | Hide All Details

Details	Finding Details	Impact (Active Sessions)	Start Date
Show	ADDM:3132078998_1_1986	4.03	Apr 4, 2008 12:00:04 PM PDT
Show	ADDM:3132078998_1_1987	4.55	Apr 4, 2008 1:00:18 PM PDT
Hide	ADDM:3132078998_1_1988	6.24	Apr 4, 2008 2:00:45 PM PDT

- Action Investigate the SQL statement with SQL_ID "2a6s3wn0nu91w" for possible performance improvements.
SQL Text `select /* big_guys */ /* pq_distribute(supplier none partition) pq_map(supplie...`
SQL ID 2a6s3wn0nu91w
- Action Investigate the SQL statement with SQL_ID "1pzgsfba2jrm8" for possible performance improvements.
SQL Text `select /* big_guys */ o_year, sum(case when nation='BRAZIL' then volume...`
SQL ID 1pzgsfba2jrm8
- Action Investigate the SQL statement with SQL_ID "dt7umutdm8p67" for possible performance improvements.
SQL Text `select /* big_guys */ supp_nation, cust_nation, year, ...`
SQL ID dt7umutdm8p67
- Action Investigate the SQL statement with SQL_ID "9sqv60uk9hjzw" for possible performance improvements.
SQL Text `select /* big_guys */ o_orderpriority, count(*) as order_count from ...`
SQL ID 9sqv60uk9hjzw
- Action Investigate the SQL statement with SQL_ID "66n44vwsmyknr" for possible performance improvements.



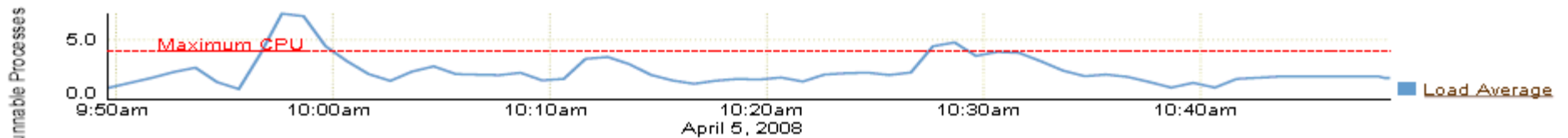
Best practice: use EM

- Transient (sub-hour) or immediate time scope
 - Requires interactivity of UI
- ‘Click on the big stuff’
 - Data visualizations display skew directly
- Takes some expertise to separate symptoms from root causes

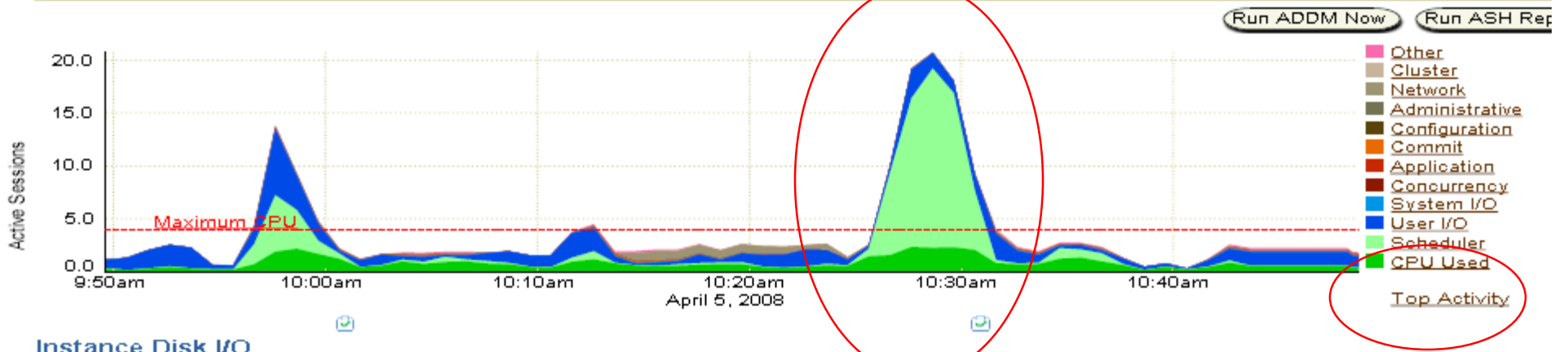
Click on an area of a graph or legend to get more detail.

View Data

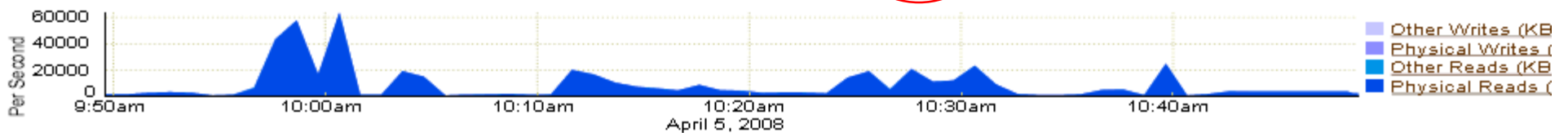
Host



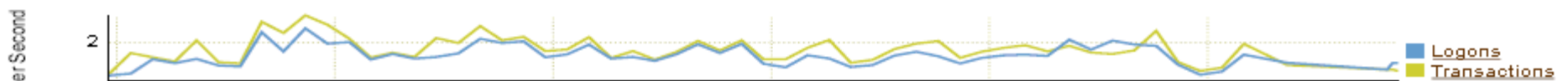
Average Active Sessions



Instance Disk I/O



Instance Throughput

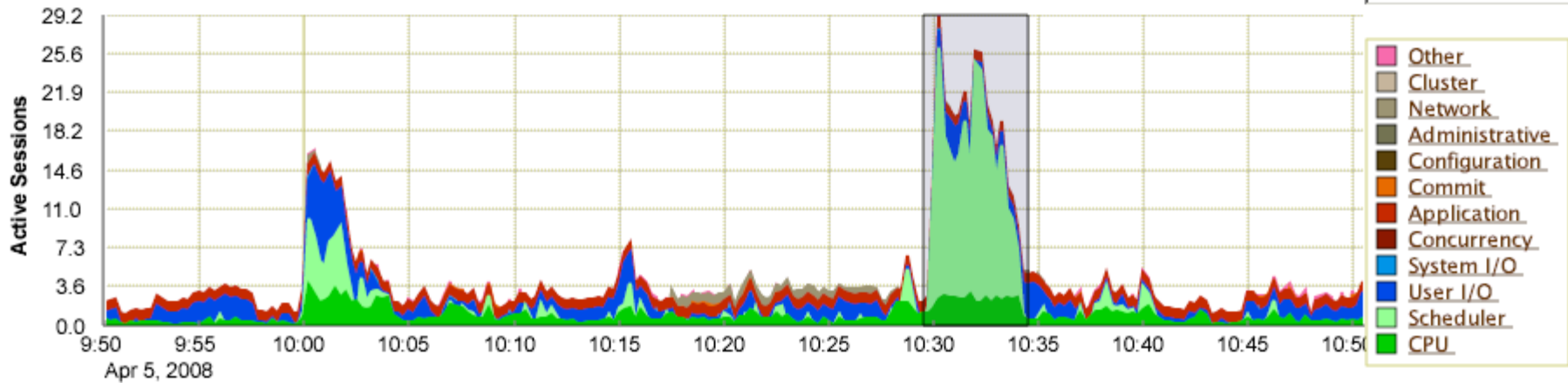


Top Activity

Switch Database Instance B_DBS232

Drag the shaded box to change the time period for the detail section below.

View Data Real Time: 15 Second Refre



Detail for Selected 5 Minute Interval

Start Time Apr 5, 2008 10:29:32 AM CDT

Run ASH Report

Top SQL

Schedule SQL Tuning Advisor Create SQL Tuning Set

Select All | Select None

Select Activity (%)	SQL ID	SQL Type
<input type="checkbox"/> 12.10	bbxb6c4kmgmmg	SELECT
<input type="checkbox"/> 7.19	b2yz2b9vga7h	SELECT
<input type="checkbox"/> 6.60	8zrv5tr71d4a	SELECT
<input type="checkbox"/> 6.30	9c09ntcqunu1u	SELECT
<input type="checkbox"/> 5.82	cn96qsdrmaub	SELECT
<input type="checkbox"/> 4.66	93sgq7vmg35xy	SELECT
<input type="checkbox"/> 4.50	bxygj7qmvrfan	SELECT

Top Sessions

View Top Sessions

Activity (%)	Session ID	User Name	Program
5.90	2170	NKANDALU	oracle@stddr46 (TNS V1-V3)
5.29	1772	AOLREP	perl@atgebs.us.oracle.cc (TNS V1-V3)
4.85	2023	MFGOPSTM	? @ap615utl (TNS V1-V3)
4.66	2228	MOCONNEL	oracle@rmlinxie01 (TNS V1-V3)
4.62	1955	MOCONNEL	oracle@moconnel-lnx (TNS V1-V3)
4.32	2203	MOCONNEL	oracle@moconnel-lnx (TNS V1-V3)

SQL Details: **bbxb6c4kmgmmq**

Switch to SQL ID

View Data

Text

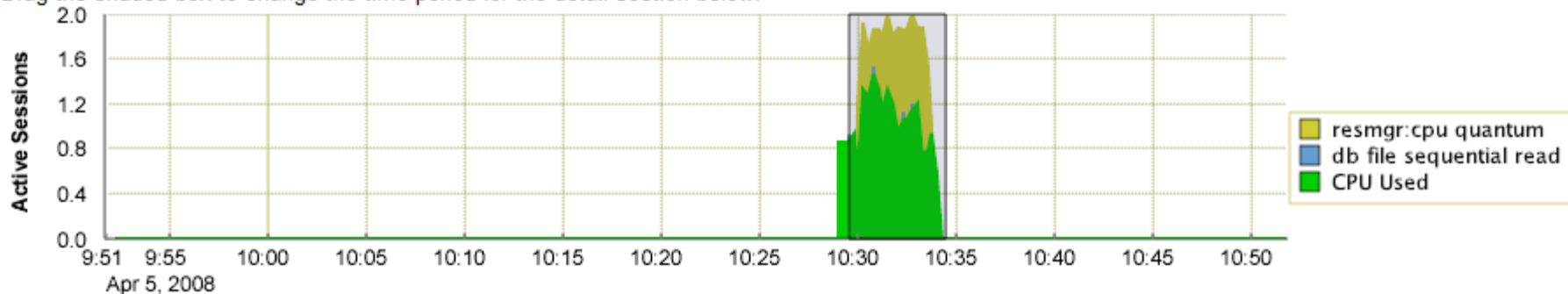
```
SELECT /*+ OPAQUE_TRANSFORM */
"RPTNO", "RPTDATE", "RPTD_BY", "VERSION", "UTILITY_VERSION", "CATEGORY", "STATUS", "SUBJECT", "UPD_BY", "CUSTOMER"
FROM "BG"."RPTHEAD" "H" WHERE "RPTDATE">:1 AND "RPTD_BY"<>'BATCH' AND "CUSTOMER" LIKE '%WPTG%' AND
```

Details

Select the plan hash value to see the details below. Plan Hash Value

Summary

Drag the shaded box to change the time period for the detail section below.



Detail for Selected 5 Minute Interval

Start Time **Apr 5, 2008 10:29:32 AM**

Activity (%)	SID	User	Program	Service	Plan Hash Value
51.89	2228	MOCONNEL	oracle@rmlnxie01 (TNS V1-V3)	boracle.com	301316116
48.11	2203	MOCONNEL	oracle@moconnel-lnx (TNS V1-V3)	boracle.com	301316116

SQL Details: **bbxb6c4kmgmmq**

Switch to SQL ID

View Data

▶ Text

```
SELECT /*+ OPAQUE_TRANSFORM */
"RPTNO", "RPTDATE", "RPTD_BY", "VERSION", "UTILITY_VERSION", "CATEGORY", "STATUS", "SUBJECT", "UPD_BY", "CUSTOMER
FROM "BG"."RPTHEAD" "H" WHERE "RPTDATE">:1 AND "RPTD_BY"<>'BATCH' AND "CUSTOMER" LIKE '%WPTG%' AND "
```

Details

Select the plan hash value to see the details below. Plan Hash Value

[Statistics](#) [Activity](#) **Plan** [Tuning Information](#)

Data Source **Cursor Cache** Capture Time **Apr 5, 2008 10:53:15 AM** Parsing Schema **MOCONNEL** Optimizer Mode **ALL_ROWS**
 View Graph Table

[Expand All](#) | [Collapse All](#)

Operation	Object	Object Type	Order	Rows	Size (KB)	Cost	Time (sec)	CPU Cost
SELECT STATEMENT			12			71,662		
FILTER			11					
TABLE ACCESS BY INDEX ROWID	BG.RPTHEAD	TABLE	9	1	0.172	71,662	557	5,287,109,561
BITMAP CONVERSION TO ROWIDS			8					
BITMAP AND			7					
BITMAP CONVERSION FROM ROWIDS			3					
SORT ORDER BY			2					
INDEX RANGE SCAN	BG.I_RPTHEAD_PRODUCT_ID	INDEX	1			1,074	9	74,441,376
BITMAP CONVERSION FROM ROWIDS			6					
SORT ORDER BY			5					
INDEX RANGE SCAN	BG.I_RPTDATE	INDEX	4			4,205	33	311,071,176
INDEX RANGE SCAN	BG.BG_ACCESS_UNQ	INDEX (UNIQUE)	10	1	0.016	3	1	22,364



ORACLE IS THE INFORMATION COMPANY